

Disasters, Dynamics, and Decisions

Strategic Planning in Stochastic Dynamic Environments

Rabeaeh Kiaghadi

January 2026



Carl von Ossietzky Universität Oldenburg
Fakultät II — Informatik, Wirtschafts- und Rechtswissenschaften
Department für Informatik

Disasters, Dynamics, and Decisions

Strategic Planning in Stochastic Dynamic Environments

Von der Fakultät für Informatik, Wirtschafts- und Rechtswissenschaften
der Carl von Ossietzky Universität Oldenburg
zur Erlangung des Grades und Titels

Doktorin der Ingenieurwissenschaften (Dr.-Ing.)

angenommene Dissertation

von

Rabeaeh Kiaghadi

geboren am 13 July 1987 in Amol, Iran

Gutachter:

Prof. Dr. Martin Fränze
Carl von Ossietzky Universität Oldenburg

Assoc. Prof. Dr. Arnd Hartmanns
University of Twente

Prof. Pedro R. D'Argenio
Universidad Nacional de Córdoba

Tag der Disputation: 16. March 2026

In loving memory of Mom...

for whom this journey began

Abstract

In disaster scenarios, optimal scheduling of rescue operations poses a challenge, as perilous injuries incurred by victims induce an instationary planning problem with individually varied and dynamically changing survival probabilities. Autonomous search-and-rescue (SAR) strategies in such environments require a precise handling of both discrete events and continuously degrading victim health conditions. In this thesis, a new model class, Interactive Markov Chains with Clocks (IMCC), is proposed that merges the concurrency and stochastic behaviors of Interactive Markov Chains with the precise timing mechanisms of Timed Automata. Unlike previous models, IMCCs accommodate continuously varying, clock-dependent transition rates — a critical feature to capture the dynamic, time-inhomogeneous behavior of rescue scenarios.

Building on a rigorous analysis of decidability and computability properties of IMCCs, a heuristic strategy synthesis algorithm has been developed and implemented to systematically construct near-optimal real-time patient visit sequences in a variety of stochastic environments, where optimality refers to maximization of the expected number of rescued victims. More generally, the approach tackles the complexity of time-critical decision-making in continuously time-inhomogeneous environments. The proposed IMCC model and its accompanying algorithms provide rigorous development of autonomous strategies under uncertainty and tight time constraints.

Zusammenfassung

In Katastrophenszenarien stellt die optimale Terminierung von Rettungsmaßnahmen eine besondere Herausforderung dar, da lebensbedrohliche Verletzungen bei Betroffenen ein nichtstationäres Planungsproblem induzieren, in dem sich individuell unterschiedliche und dynamisch veränderliche Überlebenswahrscheinlichkeiten fortlaufend entwickeln. Autonome Such- und Rettungsstrategien (Search and Rescue, SAR) in solchen Umgebungen erfordern eine präzise Behandlung sowohl diskreter Ereignisse als auch kontinuierlich degradierender Gesundheitszustände. In dieser Dissertation wird eine neue Modellklasse, Interactive Markov Chains with Clocks (IMCC), vorgeschlagen, die die Nebenläufigkeit und das stochastische Verhalten interaktiver Markow-Ketten (IMC) mit den präzisen Zeitmechanismen Zeitautomaten (Timed Automata) verbindet. Im Unterschied zu früheren Modellen erlauben IMCCs kontinuierlich variierende, uhrabhängige Übergangsraten – ein entscheidendes Merkmal, um das dynamische, zeitinhomogene Verhalten von Rettungsszenarien adäquat abzubilden.

Aufbauend auf einer Analyse der Entscheidbarkeits- und Berechenbarkeitseigenschaften von IMCCs wurde ein heuristischer Strategiessynthese-Algorithmus entwickelt und implementiert, der systematisch nahezu optimale Echtzeit-Besuchssequenzen von Patientinnen und Patienten in einer Vielzahl stochastischer Umgebungen konstruiert, wobei Optimalität die Maximierung des Erwartungswerts der Anzahl geretteter Personen bezeichnet. Allgemeiner adressiert der Ansatz die Komplexität zeitkritischer Entscheidungsfindung in kontinuierlich zeitinhomogenen Umgebungen. Das vorgeschlagene IMCC-Modell und die begleitenden Algorithmen ermöglichen eine methodisch fundierte Entwicklung autonomer Strategien unter Unsicherheit und strikten Zeitrestriktionen.

Acknowledgement

In life, *time* has often seemed to remain a few steps ahead of us, drawing us along in its wake. We are almost always looking toward what comes next: a few minutes from now, a few days ahead, or even years into the future. We keep an eye on the clock, measure how much time is left, and allow ourselves a brief breath of relief only when we reach a meaningful point along the way, relieved by the quiet sense of *arrival*. Yet no arrival belongs only to the final point. It becomes possible through all the moments that preceded it, and through the people whose presence made those moments bearable, meaningful, and possible. By standing beside us, by supporting us, and even by staying with us through memory, they keep us from falling behind time and life itself. I will remain grateful for that presence, always.

Looking back on my PhD journey, my first and deepest gratitude goes to my supervisor, *Prof. Dr. Martin Fränzle*. From our first interview eight years ago to the final days of my defense, your support gave me a sense of trust and encouragement that stayed with me throughout this work. I am especially grateful for the many rewarding conversations we had about ideas. One may begin with many raw ideas, but it requires a rare combination of knowledge, wisdom, and understanding to shape them with care and guide them toward their strongest possible form. I truly believe that your intellectual insight opened the best path for what I had in mind.

Your precision, concentration, and careful attention to formulas, results, and details were invaluable. More importantly, you taught me how to think through abstraction: how to approach a problem, however complex it may first appear, and bring it to a level at which it can be understood, reasoned about, and solved. This mentorship, however, was never confined to academic guidance alone. In the difficult and unfamiliar circumstances that migration brings into one's life, you were not only a supervisor, but also a genuinely supportive presence. For the confidence to move forward, for the courage to deal with uncertainty, and to trust that I could find my way through, for all of this, I am deeply and sincerely grateful.

I would also like to sincerely thank *Prof. Arnd Hartmanns*, who served both as my second supervisor and as one of the reviewers of this dissertation. His perspective helped sharpen my view of this work, its technical possibilities, and its broader

directions. From the beginning, he was exceptionally welcoming and generous with his time and attention. Through his help with MoDeST, his careful comments on my code, and the valuable opportunities he made possible for me to discuss my work with members of his team, I was able to see my topic from different perspectives and to understand more clearly the range of applications that my model could have. I especially appreciate the care with which you reviewed this dissertation. Your detailed comments, including the subtle points you noticed, were extremely valuable and helped improve the final form of the work.

Special thanks go to *Prof. Pedro D'Argenio*, who kindly agreed to serve as the third reviewer of this dissertation. His genuine interest in my work, the time he took to read and engage with my paper after our first discussion, and his perspective remarks during the defense meant a great deal to me. My gratitude also extends to the other members of examination committee: to *Prof. Dr. Christoph Matheja* for chairing the defense and to *Dr. Eike Möhlmann* for his supportive presence. I also greatly valued the thoughtful feedback and suggestions that *Eike* and my colleagues from the SoCPS group and DLR offered before and during the defense rehearsal.

Time, however, does not always pass with the same pace. In moments of longing, it can become slower, heavier, and more difficult to carry. Distance, too, does not always measure absence. There are people who may be far away, and yet whose presence can still be felt close by. Across these years, the companionship, support, warmth and blessings of my family reached me from far away and remained with me along the way. My beloved *Dad*, *Rahil*, *Eisa* and *Mehdi*, each of you holds a piece of my heart and I cannot thank you enough for your words, your love, and for always making me feel part of the family moments I missed, by calling me or sending me photos whenever you were together.

I was also surrounded, even from afar, by friends whose presence brought laughter, lightness, and joy into these years. My dear friends, especially *Mana*, *Maryam*, and *Nassim*, you stayed awake with me through long nights, listened patiently to my worries, made me laugh when I needed it most, and shared many moments of happiness with me. Your friendship made the difficult days easier, the joyful moments richer, and the distance less heavy.

My heartfelt thanks also go to my parents-in-law, who welcomed me into their family with so much warmth, kindness, and generosity. Throughout these years, their support, encouragement, and affection gave me a sense of comfort and belonging that I will always value. Dear *Brigitte* and *Bernd*, you made every effort to make the distance from my own family feel lighter, and through your presence and care, this place became home.

And finally, my dearest *Stefan*, my love, my husband, you were my closest companion through this journey. You believed in me with a steadiness that often held me together. With patience, tenderness, and a generosity of heart that I will never take for granted, you made room for all the selves I carried through this journey: the intense one, the joyful one, the anxious one, the exhausted one, even the crazy one, or the one who needed reassurance more often than she could admit. You stood beside *all* of me, and helped me bring this long journey, at last, to its end. I have often felt that finding you was the quiet blessing of my doctoral years. I am deeply fortunate that this chapter of my life, and the many more chapters that still lie ahead, are shared with you.

Rabeaeh Kiaghadi

April 2026, Oldenburg

Contents

Disasters — Why Minutes Matter	1
1 Anatomy of a Disaster	3
1.1 Definition and Terminology	3
1.2 Taxonomy	4
1.3 Metrics of Harm	6
1.4 Hazard Profile	8
2 Temporal Anatomy	11
2.1 Rescue Missions and The Shape of Time	11
2.2 Survival Likelihoods and Golden Hour	12
3 Problem in Focus	15
3.1 The Spark Behind the Study	15
3.2 From Chaotic Terrain to a Toy Rescue Scenario	15
3.3 Core Contributions and Thesis Outline	17
Dynamics — Modeling Time-Critical Survivability	21
4 Preliminaries and Pre-analysis	23
4.1 Theoretical Foundations	24
4.1.1 Automata Theory	24
4.1.2 State Transitions Systems	26
4.2 From Prior Models to Building Blocks	27
4.2.1 Timed Automata	27
4.2.2 Interactive Markov Chains	29
4.2.3 Clock Dependent Probabilistic Timed Automata	32
4.3 Decision Problems and Complexity Landscape	33
4.4 Requirements for a Stochastic-Timed Model	35
4.4.1 Phenomenology of The Rescue Scenario	35
4.4.2 Rescue Scenario into State-Transition System	36
4.4.3 Essential Capabilities of The Target Formalism	39
5 Gap Analysis and Rationale	41
5.1 Where Existing Models Fail: A Survey	42

5.2	In The Need of a New Formalism	48
6	Interactive Markov Chain with Clocks (IMCC)	51
6.1	Interactive Markov Chains with Clocks and Clock-dependent Rates	51
6.2	Formal Definition of IMCC: Syntax and Semantics	53
6.2.1	Structure of an IMCC	53
6.2.2	Discrete-Time Semantics of an IMCC	54
6.2.3	Continuous-Time Semantics of an IMCC and Approximability	58
6.3	Expressiveness and Scope	69
6.4	Further Steps to Take	70
7	Decidability Frontiers — What We Can('t) Verify Algorithmically	71
7.1	Decidability in Timed/Probabilistic Models: A Survey	71
7.2	On the Computability of Optimal Strategies	73
7.3	Solution Existence Criteria	74
7.3.1	Strategies in IMCCs	74
7.3.2	(Un)decidability of Strategy Optimality Problems in IMCCs	75
7.4	Implications	79
8	Time-Bounded Problems: Exact vs. Approximate Quantitative Reachability Analysis	81
8.1	Computability over Bounded Horizons	81
8.1.1	With Time Bounds in Place: The Strategy Computation	83
8.2	How Far Are We From The Optimal Strategy	89
8.2.1	Error Calculation Scheme	90
8.2.2	Worst-Case Deviation Over the Time Horizon	94
8.3	Implications	95
9	Model-Checking Tools — From Theory to Practice	97
9.1	Probabilistic Model Checkers: A Survey	97
9.2	Challenges In IMCC Model Checking	100
9.3	Modeling Rescue Scenario via Multi-Agent IMCCs	101
9.3.1	Agent Models In IMCC	102
9.3.2	Product Model In IMCC	104
9.4	IMCC Encodings in PRISM	105
9.4.1	IMCC's Rescue Scenario as a Multi-player Game	106
9.4.2	Limitations & Considerations	107
9.4.3	Single Agent Model in Disaster Responses	108
9.4.4	Multi-Agent Model in Disaster Responses	112
9.4.5	Relations and Formulas	113
9.4.6	Implementation Results	115
9.5	IMCC Encodings in MoDeST	124

9.5.1	IMCC's Rescue Scenario as Stochastic Hybrid Automata	125
9.5.2	Limitations & Considerations	125
9.5.3	Single Agent Model in Disaster Responses	126
9.5.4	Multi-Agent Model in Disaster Responses	129
9.5.5	Implementation Results	129
9.6	From Verification Limits to Practical Planning	130
Decisions — Algorithms that Save the Most Lives		135
10	Rescue Scheduling as an Optimization Problem	137
10.1	Decision-Making under Uncertainties: A Survey	138
10.2	Dynamic Environment and Its Challenges	141
10.3	Choosing a Heuristic: Rationale	143
10.4	Shortlisted Heuristic Algorithm and Final Selection	145
10.4.1	Chosen Method: Genetic Algorithm (GA).	149
11	Strategy of a Savior	153
11.1	Whom to Rescue First?	153
11.1.1	Criteria of a Rescue Mission	154
11.1.2	From Decision Flow to System Architecture	155
11.2	Implementation and Simulation	156
11.2.1	Model Assumption	156
11.2.2	Encoding a Genetic-Algorithm-Based Approach	161
11.2.3	Trade-Offs in Rescue Scheduling	162
11.2.4	Implementation Results	163
11.3	Limitations & Considerations	176
Synopsis — Lessons, Limitations, Horizons		187
12	Recapitulation & Outlook	189
12.1	Back to the Disaster Scene	189
12.2	Synthesis of Contributions	189
12.2.1	From Disaster Response to a Formal Decision Problem	190
12.2.2	A Unified Model of Time-Critical Rescue	191
12.2.3	Theoretical Boundaries and Approximation Guarantees	192
12.2.4	Verification and Strategy Synthesis at the Right Scale	193
12.2.5	Strategic Planner in Dynamic Environments	194
12.2.6	Integrated Contribution	195
12.3	Limitations & Assumptions Revisited	196
12.4	Future Research Avenues	198
12.5	Destination Met, Clock Still Ticking	199

References	201
Acronyms	221

Disasters

Why Minutes Matter

Anatomy of a Disaster

“ *If I had an hour to solve a problem, I'd spend 55 minutes thinking about the problem and 5 minutes thinking about solutions.*

— Albert Einstein

Disaster begins as a breach in expectation: For one person it starts at the moment a routine blood test becomes an incurable diagnosis; for another, with a war that barges uninvited into the quiet of everyday life. A five-year-old girl who loses sight of her mother in a crowded mall feels the same fear that a city might endure when an earthquake unhooks its foundations. The student who fails a once-in-a-lifetime entrance exam can sense a collapse just as real as the engineer who watches a dam split open under rising water. Different stages, different scales, yet all these scenes share a sudden collapse of the story we thought we were living, followed by the sinking sense that whatever comes next, must be lived on unfamiliar ground.

1.1 Definition and Terminology

The diversity of perception makes the word *disaster* more self-evident in everyday speech, yet in order to draw a proper picture of our scenario, it demands a sharper lens. A precise definition leads to fixing the variables and making the optimization goal transparent. The United Nations Office for Disaster Risk Reduction (UNDRR) frames a disaster as:

"A serious disruption of the functioning of a community or a society at any scale due to hazardous events interacting with conditions of exposure, vulnerability and capacity, leading to one or more of the following: human, material, economic and environmental losses and impacts." [Uni22]

In other words, crisis only tips into full-blown disaster when three ingredients lock into place: *hazard*, the potentially destructive event; *exposure*, the people or assets caught in its path; and *vulnerability*, the physical, social, or economic susceptibility of those assets to damage [Ped+09].

Understanding Disaster Components

Hazard. It refers to a natural or human-induced event that has the potential to cause harm to people, infrastructure, or the environment. Examples include earthquakes, floods, landslides, and storms. The intensity, frequency, and duration of a hazard determine its potential impact [Ped19; UN23].

Exposure. It describes the presence of people, assets, or infrastructure in areas prone to hazards. It is a critical determinant of disaster risk, as higher exposure increases the likelihood of damage or loss. For instance, communities located in floodplains or coastal areas are more exposed to water-related hazards [AS16; Pen21].

Vulnerability. This term refers to the susceptibility of individuals, communities, or systems to the adverse effects of a hazard. This concept can be quantified, though it is a highly specialized task, and manifests in various forms including physical, economic, social, ecological, organizational, communicational, and attitudinal vulnerabilities. Although natural hazards like earthquakes and volcanoes cannot be eliminated, their adverse effects can be mitigated by reducing vulnerability [NY16; BB14].

With those three building blocks defined, the logical next step is to give the field some structure. A concise taxonomy lets us tell at a glance which mixtures of hazard, exposure, and vulnerability call for which kind of response. The lens we choose is not merely academic: classifying by origin directs attention to technical countermeasures, while a risk-factor view foregrounds social policy; focusing on temporal profile prioritizes warning systems and surge logistics while scaling by impact guides everything from triage doctrine to interagency coordination.

In short, the taxonomy we adopt sets the coordinates for every subsequent step in problem-solving, from data collection to optimization algorithms and, ultimately, field operations. The following section therefore reviews some of main classification schemes –by origin, spatial scale and impact– and explains why these distinctions matter for any model that tries to schedule rescue operations.

1.2 Taxonomy

Disasters, whether natural or human-made, pose significant threats to human life, infrastructure, and the environment. Understanding the taxonomy of these disasters is crucial for developing effective mitigation strategies and reducing their impact.

Disaster studies offer several, not-mutually-exclusive ways to sort events, each emphasizing a different part of the risk equation [Nea97; FH03; Moh07]. They can be classified by *origin* into Natural (e.g., earthquakes, hurricanes) [GA13], Technological (industrial explosions, dam failures, nuclear accidents) [Jha10] and Socio-natural (hybrid) events, where human activity amplifies a natural trigger (landslides caused by deforestation) [Vij22].

Effective disaster management requires strategic planning that combines preparedness, response, and recovery efforts to minimize harm and impact. For natural disasters, this involves implementing early warning systems, developing comprehensive evacuation plans, and reinforcing infrastructure to withstand potential threats [Zho+18]. For human-made disasters, strategic planning includes enforcing stringent safety regulations, conducting regular maintenance of industrial facilities, and ensuring prompt cleanup efforts to mitigate environmental and health risks [CRJ24]. By integrating these strategies, communities can enhance their resilience and reduce the overall impact of disasters [Uni22].

Disasters can also be categorized based on their *spatial scale*, which determines the scope of their impact and the level of response required. The three primary scales are local, national, and global. Local disasters are confined to a specific geographic area, such as a city, town, or region. These events are typically manageable by local emergency services and resources which may not require external assistance [MS23].

National disasters, on the other hand, affect multiple regions within a country and require national-level coordination, including government agencies and potentially international assistance [CW24]. Scaling up, global disasters have widespread impacts that transcend national boundaries, involving multiple countries or regions as well as organizations like the United Nations or the Red Cross as responders [CW24].

A very crucial aspect of disasters is their *impact*, which can be assessed in terms of human, economic, and environmental factors. These criteria help determine the severity of the disaster and guide response efforts [Yew+19]. While the human impact of a disaster is measured by the number of fatalities, injuries, displacements, and the overall effect on communities, the economic impact refers to the cost of damage to infrastructure, loss of productivity, and the financial burden on affected populations [ŞM22]. When ecosystems, biodiversity, and natural resources, are affected and harmed by any type of disaster, it is considered of environmental impact [NY16; TK21].

Each taxonomic branch implies a different tempo of decision-making and mix of capabilities. Urban earthquakes compress life-saving response window into hours, demanding rapid strategic planning, whereas droughts unfold over months-long livelihood support. The taxonomy therefore acts as a time-code for responders and modelers alike: it tells us which *clock* we are racing against.

1.3 Metrics of Harm

Reliable decision-making hinges on knowing *what* harm we are trying to minimize and *how* fast it is likely to grow. Various frameworks and metrics have been developed across different domains to quantify and assess harm, each with its unique focus and methodology. Researchers and agencies have built a family of indicators that catch different sides of loss.

Understanding the metrics of harm when a hazardous event occurs is one of the requirements for effective risk management and mitigation strategies. These metrics help in determining the impact of hazardous events on people, property, and the environment, and guide decision-making processes for risk reduction and resource allocation. Our focus here is on why each metric of harm matters and how choosing one over another re-orders strategy and timing.

Human-life and health metrics. The most widely reported human-impact figures remain deaths, missing persons, and injuries. The Sendai Framework for Disaster Risk Reduction, a United Nations agreement [Ait15; PP15; Uni25], operationalizes these as per-capita indicators: Target A tracks disaster mortality (deaths and missing persons) per 100000 population, while Target B tracks affected people, including the injured/ill, per 100000 population. This normalization facilitates cross-country comparisons even when absolute numbers diverge sharply.

For longer health impacts, public-health economists prefer *Disability-adjusted life years (DALYs)*: a single indicator combining years lost to premature death with years lived with disability [AH97; Wor25], merging them into one clock of “years of healthy life gone”. Because DALYs convert both these kinds of harm into identical time unit, they allow algorithms to compare, on one scale, an event that kills thousands at once with one that keeps a smaller group sick for decades.

Economic metrics. Direct economic losses, including damage to housing, critical infrastructure, and productive assets, are tracked under Sendai Target C. To compare rich and poor countries fairly, these losses are divided by each nation’s Gross Domestic Product (GDP) [NB11; Uni25]. When economic loss is the dominant harm, response

plans lean toward protecting assets (e.g., shutting an industrial plant down quickly) and lining up insurance or other cost-sharing deals as early as possible.

Social-impact metrics. Counts such as people affected, jobs disrupted, and families forced out of their homes fall under Sendai Target B, which tracks how many lives are shaken even when no-one is killed [Uni22]. To highlight places that were already fragile before any hazard hit, planners use composite scores like the Social Vulnerability Index (SoVI) [Oli09]. SoVI ranks communities by age, income, disability and similar factors; a higher score feeds straight into a higher overall Risk Index in tools such as FEMA's National Risk Index [Fed25]. High social-vulnerability scores argue for strategies that deploy social services and communication assets early, even if physical damage is still moderate.

Environmental and ecosystem metrics. Sendai Target C-6 records cultural-heritage and ecosystem loss [Uni25], while recent UNDRR working papers propose parallel accounts for ecosystem services and carbon impacts [Uni22]). After people are safe, fragile habitats can move to the top of the to-do list. Protecting them often means sealing the spill first by deploying floating oil-containment booms, for example, before any other cleanup or recovery steps. In such cases, stopping the pollution at its edge outranks further life-saving tasks because the official response priorities shift from human safety to stabilizing the scene and cutting environmental damage [Nat25].

These markers are not the whole catalogue of disaster metrics, but the ones most often used nowadays. As technology, industry, and even daily life keep shifting, new hazard scenarios appear, old ones may look different, and the harm measurement will follow the trend. Some of the current metrics will be updated, some may fade away, and brand-new ones will be added. We have highlighted this short set since they speak directly to the kind of scenario that pushes us toward strategic, time-sensitive planning in a crisis. Choosing any one of these measurements is not just for paperwork and statistics purposes; it sets the center of gravity for the whole emergency response.

Metrics that prize lives compress decision time into minutes and push triage and evacuation to the front. Metrics that track economic loss often allow a slightly longer decision window of hours or days for action but demand fine-grained asset maps and constant cost-benefit re-evaluation as situations evolve. Focusing on social-impact shifts attention to equitable resource distribution and community liaison; where adaptation speed depends less on physical logistics and more on how fast information and trust circulate.

In addition, ecological metrics can reorder priorities once immediate human danger falls. On the other hand, quantifying harm involves aggregating individual harm to societal harm, with challenges in ensuring appropriate aggregation methods [BCH23]. The Disaster Severity Scale (DSS), for example, evaluates medical severity in multiple-casualty events, though it requires refinement for better reliability and discrimination [Bay+15].

Every one of these measures, however, hangs on a single prior fact: *the hazard profile*. Unless we know how fast an event is unfolding, how wide its footprint is, and how long the damaging forces will persist, the numbers we attach to people affected, cost estimates, or hectares burnt, turn from fact toward speculation. A fire that races through a factory in twenty minutes calls for a totally different measure than a flood that rises over a week. Emergency-management guides such as FEMA's HAZUS manual make hazard profiling the very first analytical step, because likelihood, magnitude, duration and spatial extent set the baseline for any later loss estimate [Fed19; Fed25; Uni22].

1.4 Hazard Profile

As mentioned before, *Disaster* can be defined through an equation whose variables are: *Hazard*, *Exposure* and *Vulnerability*; yet the equation is incomplete until a time stamp is attached. UNDRR defines a *Hazard* as any natural, technological, or socio-biological process that can cause harm, and its online terminology portal makes clear that hazards unfold simultaneously, cascadingly or cumulatively over time [Uni22]. Crucial is the fact that, the moment we profile a hazard we are not only naming what might happen but how fast, how long, and how often it can strike.

To give practitioners a common reference, UNDRR and the International Science Council published the *Sendai Hazard Definition and Classification Review* in 2020 and a companion set of Hazard Information Profiles (HIPs) that now covers more than 300 specific threats, from ash fall to zero-day cyber exploits [Uni22; Ste+24]. The HIPs catalogue key temporal parameters namely, typical onset speed, duration, recurrence interval, alongside spatial footprint and intensity thresholds. A 2025 update keeps that list current for new and hybrid hazards, making clear that a hazard profile is not static but updated as science and risk landscapes evolve.

The question that arises is: "Why does this matter for modeling of strategic decision?" Every optimization engine, no matter in which domain they operate, needs prior information on how quickly conditions can deteriorate. Sudden-onset hazards (e.g., explosions) demand near-real-time scheduling algorithms and compress decisive action into the order of seconds to minutes, while slow-burn hazards (e.g., droughts)

play out over days while admitting longer cycles, different indicators, and different buffers.

In order to capture a hazard's unfolding dynamics and take uncertainty into account, a model's temporal granularity must track the hazard's own clock; studies in emergency response systems show that when models are run at a coarser or finer time-step than the event itself, their forecasts become less actionable and their error grows sharply [Pet+22]. Classification, thus, matters because it shapes what our models pay attention to. In line with the Sendai Framework's multi-hazard approach, profiles are organized by *origin* (geological, hydrometeorological, biological, technological), by *tempo* (rapid-onset vs. slow-burn), and by *interaction potential* (stand-alone vs. cascading). This makes it possible to plan for contexts where hazards arrive together, trigger one another, or cluster over time where small events compound into large consequences [Uni22].

Thus, knowing the profile, changes how we design both risk models and decisions. UNDRR's technical guidance emphasizes representing hazard-exposure-vulnerability together and capturing temporal as well as spatial dynamics, precisely because risk shifts as conditions evolve. In such domains and scenarios that require time-oriented decision-making, without the right time scale, inference and action degrade [Uni22].

Within the hazard profile, several features are especially decision-shaping for strategic planning in dynamic environments: (i) *onset speed* and *duration* (how quickly conditions turn and how long they hold), (ii) *recurrence* and *seasonality* (how often we should expect a return), (iii) *spatial footprint* and *reach* (what area is at stake and how far cascades might travel), and (iv) *escalation potential* (how likely the situation is to worsen abruptly such as aftershocks, flashover, secondary releases). These are not just random details; they set sensor refresh rates, update intervals for scheduling, and the contingency space we keep in reserve [Uni22].

Pulling these threads together, the scope for our thesis becomes clear. We now have a working catalogue of hazards, a classification logic that respects real-world interactions, the main metrics of harm to evaluate outcomes and the way each profile reshapes our optimization assumptions. But as the earlier sections argued, one element cuts across them all: *Time*. While harm metrics quantify the impacts, temporal constraints ultimately dictate the effectiveness of any response [TS15]. What follows therefore dives into the *temporal anatomy* of disaster and dynamic environments, showing how minute-scale deadlines and the clocks we can and cannot bend, ultimately decide whether a strategy succeeds or fails.

Temporal Anatomy

Time is the quiet enemy in every disaster. Long before we clear the rubble or stop the flames, invisible clocks are already ticking if we think of its temporal anatomy as two nested clocks. The larger clock marks the familiar three-part arc of any crisis: the *pre-disaster* span, when we scan for risk and issue warnings; the *impact & cascade* phase, when the actual event strikes, which can be of the single or multi-hazard type, and its failures ripple outward. Multi-hazard events, such as floods and storms, often have cascading effects, increasing their complexity and impact. The third one is the *post-disaster* stretch, when repair and recovery move at a pace set by the event's length and force.

Inside this broad rhythm, however, ticks a far faster clock tracking the *rescue window*: that stretch in which lives or critical assets can still be saved. Where that window sits depends on the scenario. For a factory fire, the window is measured in the minutes the flames are actually burning; for an earthquake, it begins the instant the shaking stops and runs until the last trapped survivor's odds drop to zero; for a cancer patient it may last months or years, right up to the moment treatment can no longer change the outcome; for a hacked server it extends only while the data remain recoverable.

In short-flash events, this window fits neatly inside the middle phase; in drawn-out crises, it runs well into the so-called recovery stage. The crucial point is that, in this inner zone, distance matters less than duration: two victims may lie ten meters apart, yet the one whose airway is blocked will cross the point of "no return" first. Holding both clocks in view lets emergency-response teams keep the long-range strategy of the three phases while focusing on the minute-scale deadlines that decide *survival*. The outer timeline tells us when to fund sensors, diagnosis, drills, treatments, and rebuilding; the inner one tells us how fast the team must move once the alarm sounds.

2.1 Rescue Missions and The Shape of Time

In an emergency, every key choice comes down to time, but time is not the *same* for everyone on the scene. It runs on three tracks that often pull against one another.

- *Survival time* is the body's own countdown or the pace at which, the endangered system is resisting against a total breakdown. A trapped person can breathe only so long; bleeding, dehydration, or toxin build-up each start a separate timer that we cannot stop, only race. However, the exact curve changes from one patient/system to the next.
- *Hazard time* is the way the danger itself changes. Aftershocks, a flash-over fire, or malware that launches a full wipe, all have schedules of their own. These clocks run outside anyone's control and can suddenly erase the *safe window* that seemed available a moment earlier.
- *Operational time* is the part we can actually stretch or shrink: reaching the scene, cutting through rubble, rebooting a server from backups, or moving a patient into surgery. Better tools, clear routes, and faster decisions lengthen this track; while traffic jams, flat batteries, or slow approvals shorten it.

Rescue mission, whether it is pulling survivors from concrete, stabilizing a cancer patient or locking down a hacked database, succeeds when our operational time keeps ahead of both survival time and any surprise shifts in hazard time. These three timelines might even overlap and greatly impact other ones, when the rates of actions race against each other. However, knowing which part we control and which part controls us, stops us from gambling on minutes that biology, physics or code will never give back, and focuses effort on the minutes we can still win.

2.2 Survival Likelihoods and Golden Hour

Rescue missions, at any scale, are highly time-sensitive and the efficiency of these operations is often measured by the number of lives or other metrics saved within specific time-frames. The "Golden hour" is a commonly cited period during which the likelihood of finding survivors is highest [New+10]. The golden hour is not just arbitrary time-frame but is based on the physiological limits of human survival under various conditions [Blo+99]. The human body can survive for several days without water and weeks without food, but the ability to survive traumatic injuries is much more time-sensitive, and one can die within just a few hours if help is slow. Trauma medicine calls this, *the golden hour*, when fast care has the greatest impact on staying alive [Ala+16].

The application of this critical concept can vary depending on the nature of the disaster and the resources available, since the likelihood of survival once a disaster occurs, is closely tied to the time-frame within which rescue operations are conducted [Wex+14]. Taking a post-earthquake Search and Rescue (SAR) mission

as an example, this period is stretched to "golden 72 hours" based on historical data showing that the majority of survivors are rescued within the first three days following this disaster [MBS06; CZ15].

Data from trauma medicine give us rough "Golden hour" targets, about an hour for heavy bleeding, perhaps two for crush injuries, but the true window is never exact. It slides because many things are still unknown or left out of the calculations. Following the earthquake scenario as an instance, victims differ by age, health, gender, injury mix, and even where they are trapped; two people with the same wound can follow very different survival curves. Head-injury or shock patients who reach care after 60 minutes might face sharply higher death rates, yet the hour rule is less clear for road-traffic trauma, underlining that these numbers are only a guide [Gau+19; New+15]. Uncertainty widens further once mind-set enters the picture. Panic, tunnel vision, or a belief that no help is coming can speed physiological decline, while calm reassurance that rescuers are close often slows it down and increase the likelihood of survival [Lea94].

This shifting golden hour is a glimpse of how much uncertainty hides inside any disaster. In other words, it is just one reminder that even a familiar incident we have previously experienced, can play out along a new path. Disasters behave like living systems: weather shifts, secondary shocks, and human reactions introduce fresh parameters faster than any checklist can predict and track them. Expert teams in disaster-management practice treat these timelines as guides, never guarantees, as they have seen how a wide range of incidents can unravel in ways no single rescue plan can fully absorb.

Thus, the solid ground of this study is built by mapping what we can know and what will always stay beyond reach. Spelling out those unknowns, therefore, becomes a practical step of informing us how wide the problem really spreads. Then, We write down core assumptions, draw the line between operational certainty and environmental uncertainty, and decide which factors we can steer and for which, we have no other options than sole observation.

This discipline does not remove the event's unpredictability, but it shows us how to deal with it, through reusing data from past events where they still fit, updating models when they do not, and steering resources toward the parts of the clock we can still bend. That clear-eyed separation while acknowledging this open-ended side of the problem, is what turns rough outlines into better planning when every minute—or even megabyte—still counts. With this temporal map in hand, we look through that rescue window, wherever the timeline happens to place it, because that is where survival is on the line. Later chapters will model, verify, and optimize rescue strategies that treat seconds as first-class resources rather than incidental costs.

Problem in Focus

” *The greatest challenge to any thinker is stating the problem in a way that will allow a solution.*

— Bertrand Russell

3.1 The Spark Behind the Study

We approached disaster as a class of decision problems rather than a single domain. Whether the objective of rescue is a human life, an alerting infrastructure, or the security of a server system, the label on what we *save* matters less than how we *plan* and execute saving it, "hopefully" in the most efficient way possible with discipline about time, uncertainty, and risk. Our starting point, however, was the most immediate case: *saving lives*. That is the image most people carry when they hear the word "Disaster", and it demands *Strategy*, not just tactics.

The scale of the incident is, however, secondary. The mission may be to save one person or a hundred; extracting trapped wildlife from a wildfire front or delivering a drug schedule that shifts survival odds for oncology patients. The actors may be human responders (firefighters, ICU teams) or a small fleet of robots operating autonomously because conditions are too severe for emergency team. In each instance the work is still rescue, and the constraints are the same: minutes matter; information arrives late or noisy; every second spent for computation or thinking is a second taken from the survival window.

This thesis is motivated by the need for a strategic, transferable way of thinking that holds across those variants: a foundation that lets us be explicit about what we prioritize, when we interrupt, and how we justify that choice, all in real time. That is the through-line that shaped everything that follows.

3.2 From Chaotic Terrain to a Toy Rescue Scenario

To make that ambition operational, we distilled the disaster into a model that preserves what is essential and omits what only obscures action. Rather than

starting with a catalogue of variables, we start with a scene. Suppose a radiological blast has occurred. The environment is immediately perilous as radiation makes prolonged human presence life-threatening. Communication with the outside may be intermittent, yet victims remain inside, and time is the limiting resource. If a human rescue team enters, they become casualties themselves, and we refuse to trade one life for another.

In such settings, effective rescue mission demands an intelligent system that can reduce risk to responders while still executing a coherent mission plan: one that can perceive, decide, and act even when external guidance is delayed or unavailable. This is precisely where autonomous robots step foot in: not as replacements for human judgment, but as agents that can enter, explore, and operate under lethal constraints. We therefore consider a coordinated team of robots tasked with searching for victims and scheduling rescue actions under uncertainty and tight time budgets.

However, the people inside are not identical abstractions but surviving with sharply different trajectories: one is bleeding and losing volume, one has a broken arm, one has a head injury, another is trapped in a sealed room running out of oxygen. The fire from the blast is devouring the remaining air while radiation quietly pushes organs toward failure. The robots only know about those they can actually see or some drones have reported; who else is out there, and where, will be discovered online, in motion.

We have already mentioned that survival is not a constant when a disastrous incident happens. Each person's golden hour is their own, shaped by what really happened to them. These robots act with that information, treating every newly seen patient as someone whose chance of living is degrading over time, and they revise that view as evidence arrives. However, all of this is a pool of probabilities and estimations, not certainties. Once the mission starts, everything changes: hazards shift, patients change state, doors open and close, communications fade in and out. Anyone who has come across, or can accurately imagine, such a scene knows there are too many moving parts to count. But to conclude and declare the problem "unsolvable" only because the level of complexity and uncertainty is too high would be to surrender!

The craft is, then, to preserve the essence while shrinking the problem just enough to make disciplined decisions under pressure. We stay with rescue in a hostile environment and we keep the hard truths on the table: time is limited, the data are partial, and uncertainty is everywhere. And we ask for nothing more than "what the situation truly needs?" which is to, at each moment, choose the action most likely to increase the number of people who can still be saved!

The rescue problem we take on is therefore explicit: given a dynamic hazard field and uncertain travel times, our robotic agents should make decisions at a time in a living disaster, deciding what to do *now* and when to re-plan, so that more people live. Everything that follows serves that single obligation. We formalize rescue as a stateful process with explicit modes (Explore, travel, rescue, survival and etc.) and guarded transitions; we represent harm and urgency through survival functions and their hazard rates so worsening conditions show up directly as steeper decay; and we acknowledge two distinct notions of time: the *global time* or how the world evolves, and *mission time* or how long we spend planning and executing, with the explicit recognition that the latter is paid out of the former. This combination of survival-driven prioritization and explicit architectural policy, is what motivated a new perspective and a foundation we believe generalizes beyond this single sketch.

3.3 Core Contributions and Thesis Outline

Decisions rarely wait for perfect information. Whether a firefighter chooses a hallway while smoke thickens, a trader reacts to a sudden price swing, or an autonomous drone maps debris, each agent faces a flood of partial and quickly outdated data. In such dynamic environments, the world changes even as we observe it and plans can lose relevance in seconds. This thesis asks how to model and support effective decision-making under these conditions in order to provide a strategic planning in real-time. The guiding question that arises is simple: How can an intelligent system—human, machine, or a mixed team—perceive, learn, and act when the situation itself keeps shifting?

Here, we frame the problem as a fast loop of perception, prediction, and action. The loop must compress streaming observations into a concise belief about the current state, forecast near-term changes, and choose actions that balance speed, safety, and cost. Many illustrations in later chapters involve mobile robots, because they give a controlled test-bed and clear performance metrics, yet the proposed framework is deliberately general. The same mathematical tools can assist ambulance dispatchers in traffic, operators of unstable power grids, or supervisors of mixed human-robot workforces. By focusing on underlying principles, this thesis aims for results that transfer across domains.

To reach this goal the thesis blends two lines of work:

- **Formal probabilistic and stochastic modeling and analysis** that track uncertainty in non-stationary states and encodes clocks, time-varying hazards and

observation updates, together with decidability and approximability results for general and bounded fragments of the model.

- **Real-time strategic planning** centered on a modified genetic algorithm that keeps producing usable schedules under tight deadlines, while tolerating streaming uncertainty such as noisy survival curves and partial information.

Taken together, these two threads form a framework that combines time-indexed uncertainty with real-time decision-making under hard latency constraints. In practice, this combination supports usable scheduling under shocks and streaming updates, and provides a formal foundation for optimization in disaster-management scenarios.

Contribution

The primary contribution of this thesis is methodological: we systematically characterize decision-making in highly stochastic, rapidly evolving disaster scenarios. From this characterization, we distill five essential capabilities that a model for such processes should ideally support namely rates being clock-dependent, strictly clock-driven actions, stochastic transitions, concurrent agents and interactive composition (ref. 4.4.3). We then use these capabilities as an explicit comparison lens and examine, across major families of established formalisms, which capabilities are supported and in what form (whether in full or restricted/indirect); the results are summarized in Table 5.1.

This tabular analysis makes clear that, for the particular combination of requirements arising in our setting, the capabilities are currently distributed across different formalisms rather than offered together within a single, explicit framework. This, also, reflects the fact that existing models target different objectives and excel within their respective scopes.

This observation motivates the introduction of a new model class that consolidates these capabilities explicitly within a unified framework. Concretely, we develop a clock-driven, time-critical formalism that goes beyond classical approaches in which time dependence is typically handled parametrically and under restrictive templates; most often simple linear trends or piecewise-constant profiles, if not assumed strictly static. While such linearized or piecewise assumptions are mathematically convenient, they can flatten the richer, non-monotonic dynamics observed in real disasters, such as cascading aftershocks, compounding infrastructure failures, or weather fronts that accelerate non-linearly.

Our new clock-driven model class, on the contrary, lifts this restriction: probabilities evolve through continuously valued rate functions, allowing rather arbitrary temporal dynamics. By coupling the evolution of probabilities directly to real-time progress, the richer expressiveness of this model yields a far more faithful link between elapsed time and strategic planning, opening the door to strategies that keep pace with the unfolding uncertainties of a disaster as it actually evolves.

Building on the modeling innovation of *Interactive Markov Chains with Clocks*, the second strand of the thesis tackles how to act when exact optimality slips beyond reach. We have first proved that, once the full clock-dependent state space is admitted, deciding an optimal rescue schedule turns into an optimization problem whose exact solution is computationally intractable in general setting with no bounds on time or rate figures, and even provably unanswerable for some instances; existing verifiers and probabilistic model checking tools such as MoDeST and PRISM can encode only approximations of our model, forcing severe clock-independence or piece-wise assumptions that undermine fidelity. After experimenting with these tools under such simplifying hypotheses and documenting the blind spots they introduce, a profound modified genetic algorithm explores the remaining combinatorial frontier to tackle planning in such dynamic setting, closing the loop between theory and practice.

The overarching aim of this research is to develop a modeling framework capable of capturing the intricacies of decision-making in systems operating under dynamic, time-critical, and often uncertain conditions. This inquiry is situated at the intersection of formal methods and systems control, drawing on computational modeling, algorithmic synthesis, and the physical constraints imposed by real-world environments.

Thesis Outline

The remainder of this thesis follows the same logic that guides a rescue mission itself: Observe, Understand, Decide, then React. Part I, *Disasters* (Chs. 1–3), gives an overview of what actually matters for decision-making in a real incident. It sets out the dimensions we must account for, and converts the real-world chaos and hazards into a clearly posed scheduling problem by addressing catastrophe, introducing measurable notions of harm and urgency, and extracting from field statistics the empirical survival curves that drive every later equation.

The next part *Dynamics* (Chs. 4–9) lays the analytical foundation. It begins by asking: what kind of mathematical language can even speak about victims whose death

rates rise and fall in real time, and shows why a new formalism –Interactive Markov Chains with Clocks– must be developed and proved sound before any algorithmic planning can be trusted. Once that foundation is secure, the narrative shifts from description to action.

Part III, *Decisions* (Chs. 10–11), accepts those analytical constraints as design input by: first mapping the optimization landscape, then constructing heuristic algorithms that turn limited computational seconds into rescued lives. Simulation experiments on synthetic data track the reliability of the resulting algorithm.

Finally, *Synopsis* (Ch. 12) steps back to form the connection between the analytic and algorithmic threads, and to weigh what has been gained against what remains fragile: practical deployment constraints, ethical trade-offs, and prospects for learning systems that adapt their survival curves from field data. In short, the thesis moves from urgency to abstraction, from abstraction to executable strategy, and closes by threading those pieces back together into a coherent roadmap for time-critical, outcome-maximizing automation.

Dynamics

Modeling Time-Critical Survivability

Preliminaries and Pre-analysis

” *Successful problem solvers are more able to determine the deep structure of a problem before they proceed to match a strategy to it.*

— David Epstein

Disasters never stand still: each minute, a patient’s survival chance slides up or down, a robot hits rubble or finds a shortcut, and new sensor data changes every priority. Planning in that swirl means more than good guesses; it needs a dynamic picture we can test with mathematics. Before we can interrogate mathematics, we must convert the rescue mission scenario into a handful of repeatable frames: identify who and what can change (locations and states), measure how long each element has been in its current condition (keep track of time), and attach to every possible jump in the story a rule that says *when* it may fire and *how likely* it is to happen.

This act of abstraction turns collapsing hallways, dwindling blood pressure and robot behavior into a *state–transition* canvas whose axes are discrete modes and continuous time. Concretely, we will translate the scenario into a finite alphabet of interaction labels that synchronizes agents, per-agents clocks that tick uniformly until a transition resets them, and rate expressions that let hazards accelerate or decelerate as those clocks evolve. Only after this reduction, from live chaos to labeled, clock-driven state changes, we can map the scenario onto the formal machinery, paving the way for the gap analysis that follows.

To carry out this formalization, we need a set of theoretical tools that can express time-sensitive behavior, uncertainty, and interaction. These foundational elements form the backbone of our approach. In what follows, we introduce these building blocks as the necessary ingredients for understanding, structuring, and eventually solving the problem at hand.

4.1 Theoretical Foundations

4.1.1 Automata Theory

We begin with Automata Theory [HU69] which is a foundational area in computer science that deals with the study of abstract machines, known as Automata, which are mathematical models of computation, and the problems they can solve. It provides a framework for modeling computational processes through state-transition system, particularly useful for visualizing and understanding behavior of complex systems. Common types of Automata include finite automata, which are used to model simpler systems, and more complex models like Mealy and Moore machines, which are used in natural computing and informatics.

Automata theory's ability to represent systems as state-transition diagrams makes it an invaluable tool for sketching and analyzing problems, specially in fields like temporal concept analysis, where it provides a clear visual representation of how a system evolves over time. In temporal concept analysis, states and transitions are described as formal concepts of formal contexts, allowing for a structured representation of time-dependent systems [Wol02]. This approach can be used to model complex systems with attributes and transitions. Automata theory also extends to infinite-state systems, where states are viewed as nodes in an infinite tree. Transitions between these states can be simulated by finite-state automata, allowing for the analysis of systems with unbounded information [BG04].

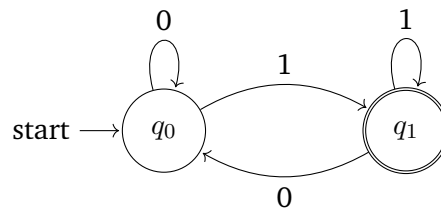


Fig. 4.1: A simple Deterministic Finite Automaton over $\{0, 1\}$ accepting strings that end with 1.

At its core, automata theory spans a spectrum of models for computation over strings. Finite automata capture regular languages and, in practice, power the lexical analysis stage of compilers and many finite-state protocol models [CL89; HMU01]. Turing machines, at the other extreme, formalize general computation and ground the theory of decidability and computational complexity [Mac96]. Between these poles, we distinguish variants that change capabilities or semantics without necessarily extending language-recognition power: Mealy and Moore machines are finite-state transducers that attach outputs to transitions or states; probabilistic automata introduce quantitative branching (with subtle acceptance semantics); and

timed automata enrich finite control with real-valued clocks to recognize timed languages for real-time verification [CL89; AD94].

Automata theory also extends beyond finite-state systems. Infinite-state automata, such as alternating tree automata, allow symbolic reasoning about systems with unbounded information [KV00]. This has enabled the analysis of complex temporal properties and has become central in model checking [SS04]. In temporal concept analysis, automata are combined with formal contexts, providing a structured way to represent states and transitions in time-dependent systems.

Applications are ranging widely including synthetic biology where plant cells have been modeled as finite automata, highlighting their decision-making processes and potential role in bio-inspired computing [KBM12]. In software verification, timed automata provide an efficient basis for model checking real-time systems [Pet99]. In security, Algebraic State-Transition Diagrams and Colored State Charts extend classical automata with algebraic or Petri-net based semantics to represent distributed, concurrent, or adversarial behaviors [Ozi+22].

Despite this versatility, automata face challenges when applied to continuous systems. Traditional discrete models struggle with representing continuous evolution, leading to issues such as state explosion, loss of precision, or undecidability in hybrid contexts. To overcome this, extensions such as hybrid automata, stochastic hybrid systems [Hes06], and approximation techniques have been developed. These frameworks integrate discrete transitions with continuous flows, allowing for the modeling of real-time, distributed, and probabilistic behaviors with greater fidelity.

However, automata theory remains a cornerstone of theoretical computer science and formal methods, offering both conceptual clarity and analytical power. Its integration with temporal logic, hybrid modeling, and probabilistic frameworks continues to push its boundaries, ensuring applicability to the increasingly complex systems encountered in modern computing, robotics, and natural sciences.

Overall, automata theory and its graphical representation through state-transition diagrams provide a rigorous yet intuitive framework for analyzing complex systems. Their strength lies in capturing both the discrete dynamics of system behavior and, in extended models, the continuous evolution of time. This ability to structure states, transitions, and timing makes automata particularly suited for reasoning about scenarios where timing and uncertainty are crucial.

4.1.2 State Transitions Systems

A formal model of our scenario begins with a state –a snapshot that freezes, at one instant, where every robot is, how far each patient’s condition has progressed, and what every local timer reads. State-transition diagrams are a central concept in automata theory, offering a clear graphical representation of states and transitions to analyze system behavior.

These diagrams are particularly effective for representing discrete dynamic behaviors, supporting early-stage system design by clarifying operational modes and predicting behaviors across transitions. Nevertheless, as system complexity increases, state-transition diagrams may become difficult to interpret without tool support, and their discrete abstraction cannot fully capture continuous dynamics. Despite these limitations, ongoing methodological and tool-based developments continue to enhance their relevance for modeling and verifying large-scale and critical systems.

State transition diagram includes a finite set of states, arrows for possible moves, and labels that tell us which action drives each move. This picture (often called a labeled transition system [VB94]) is enough to talk about logic and reachability, but it ignores the two things disasters care about most: **when** a move happens and **how likely** it is to happen.

Example 4.1 (Simple State Transition System) To illustrate the notion of a state-transition system in the most elementary form, consider a rescue robot with three abstract locations:

- *Idle*: the robot is waiting and has not yet made a decision,
- *Move*: the robot is en route to a patient,
- *Rescue*: the robot is performing the rescue mission.

The transitions between these locations are:

$$Idle \xrightarrow{decide} Move, \quad Move \xrightarrow{arrive} Rescue, \quad Rescue \xrightarrow{done} Idle.$$

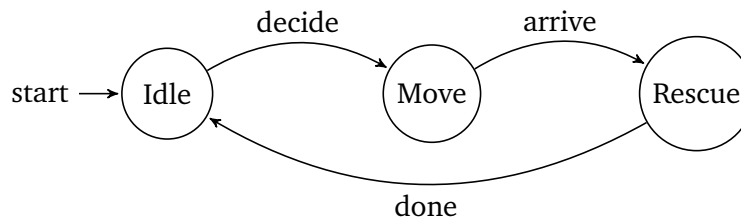


Fig. 4.2: A simple state-transition diagram for a rescue robot.

In our rescue problem, however, such a snapshot must *simultaneously* encode not only discrete locations (e.g. *Idle*, *On-route*, *Performing Rescue mission*, *Being-treated*, *Dead*, and so on) but also continuous time-stamps that measure how long the system has resided in those locations.

4.2 From Prior Models to Building Blocks

This dual encoding of discrete and continuous aspects is exactly what motivates the move from classical state-transition systems to more expressive formalisms capable of faithfully capturing and analyzing the dynamics of rescue scenarios. In order to account for *when* in our scenario, we take a recap to *Timed Automata* model[AD94]. There, every arrow is guarded by clocks and may reset clocks as it fires, giving us a formal handle on strict deadlines and urgent jumps .

Another model class that accounts for *likelihood of an event* by providing a notion for probability together with interactions we need, is *Interactive Markov Chain*: components that race on continuous-time rates yet still synchronize through labels when a joint action is required[Her02]. In disaster scenarios, the occurrence probability of critical events is strongly governed by the temporal dynamics of the situation. Understanding how such probabilities evolve over time is essential for estimating the likely timing of events. Among the available modeling frameworks, the class of *Clock-dependent Probabilistic Timed Automata (Cd-PTA)* offers a suitable abstraction for capturing such time-varying probabilistic behaviors [Spr17], which will be explained in more detail below.

4.2.1 Timed Automata

Definition 4.2 (Timed Automaton) A timed automaton (TA) is a tuple

$$\mathcal{A} = (L, \ell_0, X, \Sigma, E, \text{Inv})$$

where

- L is a finite set of locations, with $\ell_0 \in L$ the initial location;
- X is a finite set of real-valued clocks; a clock valuation is a map $v : X \rightarrow \mathbb{R}_{\geq 0}$, and $\text{ClockVal}(X) := \mathbb{R}_{\geq 0}^X$;
- Σ is a finite set of action labels (optionally including an internal label τ);
- $E \subseteq L \times \Sigma \times \text{CC}(X) \times 2^X \times L$ is a finite set of edges; each edge $(\ell, a, g, R, \ell') \in E$ has source ℓ , target ℓ' , action $a \in \Sigma$, guarded by a set of clock constraints $g \in \text{CC}(X)$, and reset set $R \subseteq X$.

- $\text{Inv} : L \rightarrow \text{CC}(X)$ assigns an invariant to each location, constraining how long the system may remain in ℓ .

The core elements of a Timed Automaton (TA), following Alur and Dill [AD94], are a finite set of discrete *locations* L and a finite set of real-valued *clocks* X . The current readings of all clocks are represented by a *clock valuation*, i.e. a mapping $v : X \rightarrow \mathbb{R}_{\geq 0}$, where we denote the set of all valuations by ClockVal .

Clock valuations. Let X be the set of clocks. A *clock valuation* is a mapping function $v : X \rightarrow \mathbb{R}_{\geq 0}$ assigning a nonnegative real to each clock. For a delay $\delta \geq 0$, time elapse updates a valuation pointwise by $(v + \delta)(x) = v(x) + \delta$ for all $x \in X$. We write $v \models g$ to denote that valuation v satisfies a clock constraint $g \in \text{CC}(X)$.

Clock constraints and Invariants. The behaviour of a TA is governed by *clock constraints*. The constraint language $\text{CC}(X)$ consists of all finite conjunctions of atoms of the form $x \sim k$ (or $x - y \sim k$, without extending expressiveness) where $x, y \in X$, $\sim \in \{<, \leq, =, \geq, >\}$, and $k \in \mathbb{Z}$ (typically $k \geq 0$). For $v \in \text{ClockVal}(X)$ and $g \in \text{CC}(X)$, satisfaction $v \models g$ is defined in the natural way. Each location $\ell \in L$ is equipped with an *invariant* $\text{Inv}(\ell) \in \text{CC}(X)$, which specifies how long the system may remain in that location: a state (ℓ, v) is valid only if $v \models \text{Inv}(\ell)$.

Clock dynamics. All clocks evolve synchronously with time: for each $c \in X$ we have $\frac{dc}{dt} = 1$ with respect to a global time variable t . A reset set $R \subseteq X$ updates a valuation by $v[R := 0]$, which sets all clocks in R to 0 and leaves the rest unchanged.

Transitions. The discrete dynamics is given by the finite relation

$$E \subseteq L \times \Sigma \times \text{CC}(X) \times 2^X \times L,$$

where $(\ell, a, g, R, \ell') \in E$ denotes a transition from ℓ to ℓ' labelled by a , guarded by g , and resetting all clocks in R . A transition is *enabled* in state (ℓ, v) iff $v \models \text{Inv}(\ell)$ and $v \models g$. If it is taken, the successor state is $(\ell', v[R := 0])$, which must satisfy the target invariant $v[R := 0] \models \text{Inv}(\ell')$.

Semantics. The semantics of a TA is the timed transition system whose states are pairs (ℓ, v) with $v \models \text{Inv}(\ell)$, and whose transitions are of two kinds:

1. *Time-elapse:* $(\ell, v) \xrightarrow{\delta} (\ell, v + \delta)$ for $\delta \geq 0$, permitted iff for all $t \in [0, \delta]$ we have $v + t \models \text{Inv}(\ell)$.

2. *Discrete step:* $(\ell, v) \xrightarrow{a} (\ell', v[R := 0])$ whenever $(\ell, a, g, R, \ell') \in E$, $v \models \text{Inv}(\ell)$ and $v \models g$, and $v[R := 0] \models \text{Inv}(\ell')$.

Thus, the dynamics of TA arise from the continuous and synchronous growth of clocks, combined with invariants that restrict residence time in locations and discrete transitions that reset subsets of clocks. Discrete jumps occur at dedicated time instants which are determined, in general nondeterministically, by when the guards become satisfied. This combination provides the expressive power of TAs to capture quantitative timing properties of real-time systems.

4.2.2 Interactive Markov Chains

Definition 4.3 (Interactive Markov Chain (IMC)) An Interactive Markov Chain (IMC) is a tuple

$$\mathcal{I} = (S, \text{Act}_\tau, \rightarrow, \Rightarrow, s_0)$$

where

- S is a finite or countable set of states, with $s_0 \in S$ the initial state;
- Act_τ is a finite set of action labels containing a distinguished internal action τ , and $\text{Act} := \text{Act}_\tau \setminus \{\tau\}$ denotes the set of external actions;
- $\rightarrow \subseteq S \times \text{Act}_\tau \times S$ is the set of interactive transitions; we write $s \xrightarrow{\alpha} s'$ for (s, α, s') ;
- $\Rightarrow \subseteq S \times \mathbb{R}_{>0} \times S$ is the set of Markovian transitions; we write $s \xRightarrow{\lambda} s'$ for (s, λ, s') with rate $\lambda > 0$.

Aggregated rates and embedded probabilities.. For $s, s' \in S$, the rate to move from state s to state s' and the exit rate of state s' , are as follows:

$$R(s, s') := \sum \{ \lambda \mid (s, \lambda, s') \in \Rightarrow \}, \quad E(s) := \sum_{s' \in S} R(s, s').$$

If $E(s) > 0$, the embedded probability is $P(s, s') := \frac{R(s, s')}{E(s)}$.

An Interactive Markov Chain (IMC) features two distinct classes of transitions: *interactive transitions*, which are instantaneous and labelled by actions, and *Markovian transitions*, which are governed by exponential distributions.

Markovian transitions. Formally, an IMC contains a set of Markovian transitions

$$mT \subseteq L \times \mathbb{R}_{>0} \times L,$$

where $(\ell, r, \ell') \in mT$ denotes an autonomous transition from location ℓ to ℓ' with rate $r > 0$. Such a transition induces a stochastic delay: if the system resides in ℓ , the probability that the transition to ℓ' occurs within time $t \in \mathbb{R}_{\geq 0}$ is

$$\Pr[\text{jump to } \ell' \text{ before } t] = 1 - e^{-rt}.$$

The residence time in ℓ is therefore exponentially distributed with parameter $E(\ell) = \sum_{\ell'} R(\ell, \ell')$, where $R(\ell, \ell') = \sum\{r \mid (\ell, r, \ell') \in mT\}$ denotes the aggregated rate to ℓ' .

Interactive transitions. Alongside Markovian dynamics, an IMC also supports interactive, action-labelled transitions $s \xrightarrow{a} s'$ that are taken instantaneously, without delay. These model controllable or synchronising moves between components. A maximal progress assumption usually applies.

Maximal progress. In closed IMCs, interactive transitions (in particular the internal action τ) preempt Markovian delay: while any interactive step is enabled in s , time does not elapse.

Residence time and race policy. Whenever several Markovian transitions are enabled concurrently from the same location ℓ , a *race condition* arises: all exponential timers associated with the enabled transitions start simultaneously, and the first one to expire determines the successor. The probability that the next transition leads to ℓ' is therefore

$$P(\ell, \ell') = \frac{R(\ell, \ell')}{E(\ell)}.$$

Hence the stochastic race between rates induces a discrete probabilistic branching, with branch probabilities proportional to the transition rates.

- If $IT(s) \neq \emptyset$ (*interactive urgency*), then by maximal progress no time elapse is permitted in s ; at least one interactive step must be taken immediately (resolved by a scheduler).
- If $IT(s) = \emptyset$ and $E(s) > 0$ (a *Markovian state*), the residence time in s is exponential:

$$T_s \sim \text{Exp}(E(s)), \quad \Pr[T_s > u] = e^{-E(s)u} \quad (u \geq 0).$$

Upon expiry, the successor state is drawn from the embedded distribution

$$\Pr[s \rightarrow s' \mid \text{leave } s] = P(s, s') := \frac{R(s, s')}{E(s)}.$$

- If $IT(s) = \emptyset$ and $E(s) = 0$, then s is a *deadlock (absorbing) state*: there is no outflow and the process can remain in s indefinitely.

Parallel composition (synchronisation over $A \subseteq \text{Act}$). For two IMCs I_1 and I_2 and a synchronization set $A \subseteq \text{Act}$,

$$I_1 \parallel_A I_2 = (S_1 \times S_2, \text{Act}_{\tau,1} \cup \text{Act}_{\tau,2}, \rightarrow, \Rightarrow, (s_{0,1}, s_{0,2}))$$

where interactive steps synchronize on A and interleave otherwise, and Markovian steps lift independently. This composition is defined by the following SOS rules:

(Sync)

$$\frac{s_1 \xrightarrow{a} s'_1 \quad s_2 \xrightarrow{a} s'_2 \quad a \in A \setminus \{\tau\}}{(s_1, s_2) \xrightarrow{a} (s'_1, s'_2)}$$

(Interleave-L/R)

$$\frac{s_1 \xrightarrow{\alpha} s'_1 \quad \alpha \notin A}{(s_1, s_2) \xrightarrow{\alpha} (s'_1, s_2)} \quad \frac{s_2 \xrightarrow{\alpha} s'_2 \quad \alpha \notin A}{(s_1, s_2) \xrightarrow{\alpha} (s_1, s'_2)}$$

(Markov lift-L/R)

$$\frac{s_1 \Rightarrow^\lambda s'_1}{(s_1, s_2) \Rightarrow^\lambda (s'_1, s_2)} \quad \frac{s_2 \Rightarrow^\mu s'_2}{(s_1, s_2) \Rightarrow^\mu (s_1, s'_2)}$$

Runs and schedulers. A run is an alternating sequence $(s_0, u_1, \alpha_1, s_1, u_2, \alpha_2, s_2, \dots)$, where each $u_i \geq 0$ is time-elapse in a Markovian state and each α_i is an instantaneous interactive action (possibly τ). A *scheduler* (policy) σ resolves interactive nondeterminism by selecting, for any finite history, which enabled interactive transition to take (or, equivalently, a distribution over them).

In summary, IMCs combine action-labelled, nondeterministic interactive steps with rate-labelled, exponentially timed autonomous transitions. The stochastic semantics is induced by the exponential race policy, which resolves simultaneously enabled Markovian transitions into a probability distribution over successors. This dual structure allows IMCs to conservatively extend both Labelled Transition Systems (purely interactive) and Continuous-Time Markov Chains (purely Markovian), thereby enabling compositional modelling of systems that exhibit both nondeterministic interaction and stochastic timing.

4.2.3 Clock Dependent Probabilistic Timed Automata

The clock-dependent probabilistic behavior of a cdPTA extends Probabilistic Timed Automata (PTA) by allowing branch probabilities to depend on the current clock valuation [Spr17]. In a PTA, clocks already determine when probabilistic edges are enabled and which clocks are reset when such an edge fires, but the associated probability distribution is not, in general, a function of the current clock values.

In a cdPTA, the probability of selecting successor locations may itself vary with the clock valuation. Accordingly, instead of a simple transition relation between locations, a cdPTA features a finite set of *probabilistic edges* equipped with clock guards, clock resets, and a probability distribution over targets.

Definition 4.4 (Clock-dependent probabilistic timed automaton) A clock-dependent probabilistic timed automaton (cdPTA) is a tuple

$$\mathcal{P} = (L, \bar{l}, X, \text{Inv}, \text{prob})$$

where:

- L is a finite set of locations with initial location $\bar{l} \in L$;
- X is a finite set of clocks;
- $\text{Inv} : L \rightarrow \text{CC}(X)$ associates an invariant condition from the set of clock constraints with each location;
- $\text{prob} \subseteq L \times \text{CC}(X) \times \mathcal{D}\text{ist}(2^X \times L)$ is a finite set of probabilistic edges.

Clock valuations and constraints. Similar to Timed Automata, a *clock valuation* is a map $v : X \rightarrow \mathbb{R}_{\geq 0}$. For $\delta \geq 0$, time elapse updates v pointwise by $(v + \delta)(x) = v(x) + \delta$ for all $x \in X$. For $R \subseteq X$, the reset $v[R := 0]$ sets clocks in R to 0 and leaves the others unchanged. The constraint language $\text{CC}(X)$ consists of finite conjunctions of atoms $x \sim c$ with $x \in X$, $\sim \in \{<, \leq, =, \geq, >\}$, and $c \in \mathbb{N}$. $v \models g$ when valuation v satisfies guard $g \in \text{CC}(X)$.

Clock-dependent distributions. For a set Q , a *distribution template* $\mathfrak{d} : \mathbb{R}_{\geq 0}^X \rightarrow \text{Dist}(Q)$ maps each valuation to a distribution over Q ; notation $\mathfrak{d}[v]$ indicates the distribution at valuation v . With this convention, $\mathcal{D}\text{ist}(Q)$ denotes the set of such templates.

Probabilistic edges (templates). In a cdPTA, each probabilistic edge $(\ell, g, \mathfrak{p}) \in \text{prob}$ has source $\ell \in L$, guard $g \in \text{CC}(X)$, and a *distribution template* $\mathfrak{p} : \mathbb{R}_{\geq 0}^X \rightarrow \text{Dist}(2^X \times$

L). Given a valuation v , the instantiated distribution $p[v]$ is over pairs (R, ℓ') with $R \subseteq X$ (clocks to reset) and $\ell' \in L$ (target).

Semantics. The semantics of a cdPTA is therefore not simply a set of possible runs or traces, as in TA, but a probability space over runs. The semantic state space is

$$S = \{ (\ell, v) \in L \times \mathbb{R}_{\geq 0}^X \mid v \models \text{Inv}(\ell) \}.$$

There are two kinds of transitions:

- *Time-elapse:* $(\ell, v) \xrightarrow{\delta} (\ell, v + \delta)$ for any $\delta \geq 0$ iff $v + t \models \text{Inv}(\ell)$ for all $t \in [0, \delta]$.
- *Discrete probabilistic step:* if $(\ell, g, p) \in \text{prob}$ and $v \models g$, then for any $(R, \ell') \in 2^X \times L$ such that $v[R := 0] \models \text{Inv}(\ell')$,

$$(\ell, v) \xrightarrow{p} (\ell', v[R := 0]) \quad \text{with probability } p[v]((R, \ell')).$$

Runs and schedulers. A run alternates nondeterministic time elapse (subject to invariants) and discrete probabilistic steps instantiated at the current valuation.

$$(\ell_0, v_0) \xrightarrow{\delta_1} (\ell_0, v_0 + \delta_1) \xrightarrow{p_1} (\ell_1, v_1) \xrightarrow{\delta_2} \dots$$

where each $\delta_i \geq 0$ is a time delay and each p_i is a distribution induced by some enabled edge. Nondeterminism (delay amount and which enabled edge to take) is resolved by a scheduler. There is no probability assigned to delays. The probabilities arise only from the valuation-dependent distributions $p[v]$ on edges.

In summary, cdPTAs generalize TA by replacing nondeterministic discrete branching with probabilistic branching that depends continuously on clock valuations. This extension makes the model expressive enough to capture time-dependent randomized behavior, yet at the price of higher semantic complexity, since system behavior is represented by probability distributions over traces rather than plain sets of traces.

4.3 Decision Problems and Complexity Landscape

Timed models are evaluated by three parameters that govern whether analysis is possible and useful: *Reachability* (can a desired goal be hit/reached), *Decidability* (is there any algorithm that answers the reachability question for the model class?), and *Complexity* (what is the worst-case cost of such an algorithm). The first parameter determines whether quantitative assessments such as probabilities or expected times can be computed soundly (reachability).

In order to obtain an answer to such question, we ask whether verification and synthesis are possible at all (decidability), and whether analyses scale beyond toy examples (complexity), as properties of the reachability problem. Different formalisms couple clocks, probability, and interaction to different degrees and this coupling moves the frontier of what is decidable and how hard it is to compute.

This coupling and different combinations of these variables fundamentally change the nature of the problem: some combinations admit a finite abstraction and hence lead to decidable and algorithmically tractable analysis, while others inevitably push the model beyond decidability or into prohibitively high complexity.

When a finite abstraction exists, the analysis can be reduced to symbolic state exploration. When it does not, the model necessarily relies on numerical kernels in order to compute approximation, estimation, or optimization. Thus, the choice of model commits one to a particular decidability/complexity profile. In safety-critical scheduling (e.g., rescue mission), these parameters decide whether one can compute guaranteed time-bounded success probabilities or only approximate them, which is why the baseline theory is laid out explicitly before requirements.

For Timed Automata (TA), the Alur–Dill construction finitises dense time via regions, yielding decidable location reachability with PSPACE complexity, which is the classic baseline for real-time verification. Beyond control-state reachability, the binary reachability relation over configurations is effectively definable in mixed linear arithmetic (FO over reals with an integer predicate), and this definability yields an EXPSPACE algorithm for the reachability fragment of parametric TCTL together with a NEXPTIME lower bound. These positive results coexist with fragile boundary: a variety of seemingly modest extensions (e.g., minimizing the number of clocks, complementability/determinisation over ω -words, or adding certain additive/diagonal constraints) cross into undecidability or highly undecidable territory [QSW17; CLP20].

Interactive Markov Chains (IMC) interleave instantaneous interactive steps with autonomous exponential delays. Under maximal progress, interactive steps preempt delays and races induce discrete branching proportional to rates. Once a scheduler resolves interactive nondeterminism, several quantitative reachability objectives (e.g., extremal expected time to reach a goal and long-run average time in a set) reduce to stochastic shortest-path problems for which value-iteration or Linear-Programming-based solvers apply, with prototype tool support reported. Time-bounded reachability in closed IMCs admits discretisation-based approximations with controllable error terms that mirror CTMC transient analysis once nondeterminism is fixed [Guc+12; HK09; Her02].

Clock-dependent Probabilistic Timed Automata (cdPTA) allow edge probabilities to depend on the current clock valuation, which breaks the time-invariance that powers standard TA abstractions and yields undecidability of quantitative reachability in general once enough clocks (≥ 3) are present. A tractable case is the one-clock, affine, initialised fragment where the clock-dependent probabilities are affine over clock intervals with natural endpoints and between non-constant edges. The clock must hit an integer (by reset or time progress), enabling an exact finite IMDP abstraction. Thus, qualitative and quantitative reachability are solvable in polynomial time via the IMDP reduction, whereas dropping affine dependence or initialisation breaks the endpoint-extremal property the construction relies on [Spr24; Spr21].

These baselines clarify the trade-space. TA offers a decidable core via finite quotients but faces hardness/undecidability at language level or under certain quantitative extensions. IMC remains numerically tractable once interactive nondeterminism is resolved, with time-bounded reachability available through discretisation with proven error bounds. cdPTA regains decidability only under the tight 1-clock/affine/initialised discipline with an IMDP compilation. For rescue-mission planning with time-varying mortality, what is needed is to encode precise hazard dynamics together with interactive choices and TA-style guards, which standard TA and IMC do not offer jointly and plain cdPTA does not express for rates.

4.4 Requirements for a Stochastic-Timed Model

The previously reminded four fundamental layers –plain transitions, real-time guards, stochastic synchronization, and clock-dependent probabilities– cover nearly all the requirements of our rescue model; the only missing piece is the clock-dependent rates, which we introduce next to lay a firm foundation for the proofs and algorithms in the following chapters.

4.4.1 Phenomenology of The Rescue Scenario

Let us consider our motivation scenario once again. Catastrophe victims do not deteriorate at a uniform, deterministic pace; each patient carries an individual survival curve whose death-rate can accelerate, decelerate, or even oscillate as bleeding, toxicity, or temperature exposure unfold. For example, a bleeding wound may seem not harmful at first, yet rise sharply once blood loss passes a physiological tipping point, whereas gas intoxication risks may fade if ventilation improves. At any given instant rescue robots, victims and hazards co-evolve: travel times are uncertain, medical conditions deteriorate as clocks tick, and robots may work around their local clock, keeping track of their time budget.

Overlaying this medical clock is a swarm of concurrent agents –robots, patients, medics, drones– each following its own motion dynamics, communication delays, and battery limits. Robots leave a staging area, choose the next patient, traverse uncertain travel times, and either reset their internal clocks upon arrival or abort if the patient dies en-route. Intermittent sensor updates from drones or human spotters may instantly reshuffle priorities, forcing the team to re-schedule on the fly. The resulting scenario is a composition of overlapping timelines where every agent’s action can ripple through the survival prospects of every patient.

Crucially, these timelines are punctuated by stochastic and forced changes. Travel delays and treatment durations are variable of time; sudden events such as building collapses, aftershocks, equipment failures, can reset clocks or trigger immediate changes that override previously planned actions. At any decision point, the system’s future expands into a branching tree whose edges carry clock-dependent rates rather than fixed probabilities, reflecting the real-time drift of risk. Field intuition alone cannot expose the compound effect of these factors on the final success metric.

A mathematical model offers a controlled sandbox where strategies can be stress-tested, compared and improved before they are deployed in the real hazardous domain. It also provides the rigor needed for proof-based safety guarantees and for the automated synthesis of policies; capabilities that simulations alone rarely deliver. Formal models like classic Timed Automata, Continuous Time Markov Chains (CTMCs), Interactive Markov Chains (IMCs) and Clock-dependant Probabilistic Timed Automata (Cd-PTA), as mentioned earlier, each capture fragments of this behavior.

4.4.2 Rescue Scenario into State-Transition System

The rescue narrative introduced in Chapter 3.2 can be formalized as a *time - inhomogeneous continuous-time Markov model with interaction labels*, constructed as follows: We model each casualty in a disaster scene as progressing through three possible clinical states. Immediately after the incident, the victim sustains an injury whose severity determines the initial state. It is possible that the individual remains physiologically stable and shows no direct harm, yet the evolving environment keeps the person exposed to future danger.

Another possibility is that the victim is wounded but with a treatable injury; without timely care the condition will deteriorate and eventually become fatal. Also, sometimes the damage is so extensive that no medical intervention can prevent death at which the patient is effectively lost from the outset. Transitions among these states occur as time passes and as medical resources are applied or withheld, capturing

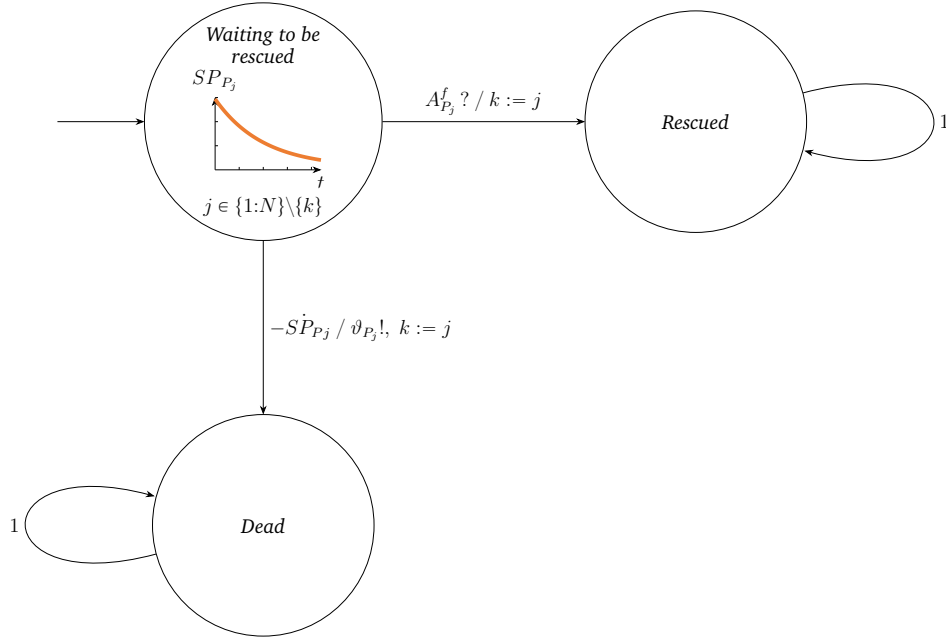
the critical pathways of casualty evolution. Our final formulation of casualty model suggests that a victim can occupy exactly one of three mutually exclusive clinical states:

1. *Alive- waiting to be rescued*: The patient is alive and potentially recoverable but has not yet received treatment from the robotic response team. All victims enter this state at time 0 unless immediate triage identifies them as lost.
2. *Dead/Lost*: Irreversible physiological decline has occurred; no subsequent intervention can restore life. Transition from State 1 to State 2 is triggered when the time-dependent injury progression reaches a critical threshold before assistance arrives.
3. *Rescued*: Definitive medical intervention by the robotic team has been delivered within the allowable time window, stabilizing the patient. Only at this point is the victim counted as saved.

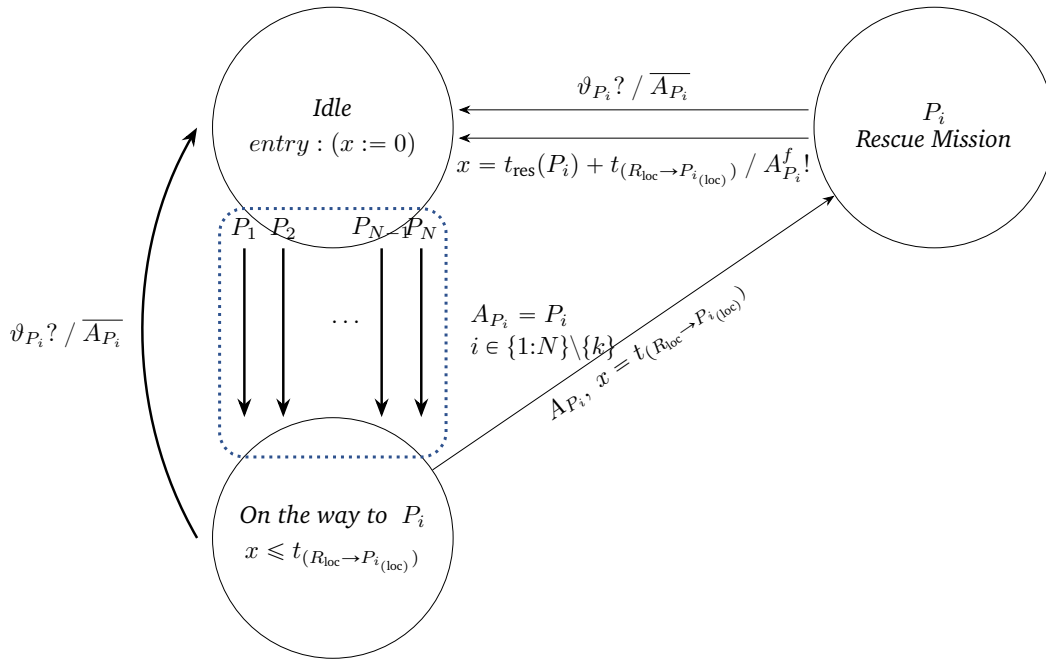
Transitions are therefore State 1 $\rightarrow$ State 3 upon successful treatment, or State 1 $\rightarrow$ State 2 if the physiological deadline is missed. No other transitions are permitted, and *Rescued* and *Dead* are absorbing states. This tri-state abstraction ensures that survivability is strictly contingent on timely robotic intervention, enabling precise quantitative analysis of rescue policies and resource allocation. In patient's model figure 4.3a, therefore, patient can reside in one of these three states. In the first state, patient's health condition is degrading based on a survival probability function $SP_{P_i}(t)$ assigned to each patient, potentially leading to sudden death with a rate of $-\dot{SP}_{P_i}(t)$ at time t ; once the patient has died, it however remains dead.

In each disastrous incident, the number of victims can be, most probably, more than one. Therefore, this patient model refers to any possible patients, and eventually, the overall system model includes a three-state model per patient, each with a different survival function and death rate. Another agent present in the scenario is the robot. Each robot, depending on the number of victims within its range or in the whole area, can occupy different states. All robots in the scenario initiate their operation from the *Idle* state, where the clock is reset. Each robot has a nondeterministic choice among the victims that can be still selected for rescue mission, and this choice can be influenced by the metrics defined for the search-and-rescue task; however, it is not certain.

If this specific robot either selects -or is assigned to- one of the victims/patients (action A_{P_i}), it travels toward that patient to rescue them, with the required time to reach him/her being counted. When the robot reaches the patient, the treatment procedure commences. This procedure may succeed or may fail. In either case, the



(a) Patient state-transition diagram



(b) Robot state-transition diagram

robot returns to the *Idle* state and attempts to select another victim for the next operation. To reduce the number of states robot can occupy, in favor of visibility and simplification, we call them locations instead of states. This way, the robotic agent model constitutes of three locations where each robot is in *Idle*, *On the way to Patient i* , and *Patient i Rescue Mission*.

The interaction between these two models that comprises the product rescue scenario, includes several signals from Patient model to Robot and back. If the patient dies before the rescue mission is done, robot receives a death signal (v_{P_i}) from supervisor (A drone, its own sensor or CCTV), terminates the decision (action $\bar{A}_{P_i}$), updates its location and becomes *Idle* again in order to asses the current situation and make another decision. The same can happen when robot is on its way to the patient. In case of rescue mission being successful, patient will be added to *Final Sequence List*, sends the signal of action for patient i failed ($A_{P_i}^f$) and goes back to *idle* for another decision making process. Each patient that dies or is rescued, will be eliminated from the available patient list.

This translation delivers the executable artifact needed for both model checking and algorithmic planning: every simulation trace corresponds to a path through the stochastic-timed transition system, and every policy intervention can be formalized as a scheduler that resolves its nondeterministic choices.

4.4.3 Essential Capabilities of The Target Formalism

The rescue mission we study unfolds with certain behavioral characteristics while dealing with a lot of uncertainty induced by the nature of the scenario. Although there is a vast variety of scenarios that all can be mapped into a mutual formalization, here, we elaborate around our motivation scenario which eventually can be generalized to one stochastic time-critical system.

Stochastic transitions. Travel durations, treatment time and sudden deaths are random variables, not deterministic constants. In order to capture these parameters, the state graph needs autonomous transitions whose firing times follow probability distributions derived from survival rate functions. Continuous-Time Markov Chains (CTMC) provide the baseline machinery, but their time-invariant rates are too restrictive for rescue domains where hazards accelerate or decelerate over time.

Forced transitions and clock resets. Certain events such as robot arrival, target patient's death, changes in the environment, must fire immediately once a guard condition is met, and they often re-initialize local timers so downstream hazards depend on the time since that event. Timed Automata introduce precisely this notion of urgent

jumps combined with explicit clock-reset sets, giving the model a way to enforce real-time commitments.

Clock-dependent rates. Victim survival rarely decays monotonically and at a constant rate: it may plateau, fluctuate, or drop after a physiological tipping point, and stabilize once treatment starts. Such behavior are most naturally specified by rate expressions that are explicit functions of clock values (e.g., $\lambda(x, y) = \min(0.1x - y^2, 1)$ where x is wait time and y treatment time). Although time-varying rate function induces an associated non-exponential delay distribution, a distribution-based encoding would require repeatedly reconstructing and resampling that distribution whenever clocks are reset or interaction events occur. Existing stochastic-automata style formalisms either use countdown timers, which do not retain the elapsed-time information needed in our model, or admit clocks without supporting the required clock-dependent autonomous timing behaviour.

Concurrent agents. Robots, patients, drones, humans, evolve on separate timelines yet occasionally synchronize. Flattening them into a single global automaton would explode the state-space; instead, the formalism must offer a compositional process in which each agent carries its own clocks and transitions while a product construction preserves probabilistic semantics.

Interactive compositions. A robot's action must coincide with the patient automaton switching from Waiting to Rescued; a drone's "hazard cleared" broadcast should unblock every robot simultaneously. These synchronized moves are neither purely stochastic nor purely deterministic; they are negotiated across agents at the first instant when all guards align. Formal models therefore require labeled interactive transitions whose execution is conditioned on multi-party agreement, preserving both urgency and probabilistic branching.

Together, these five capabilities form the minimal checklist for any framework that claims to represent time-critical, stochastic, multi-agent rescue missions. Later on, in Chapter 5, we will systematically analyze these capabilities and assess existing formalisms against this checklist to reveal whether any of current model classes is suitable for analyzing our rescue scenario or where they fall short, setting the stage for a new, integrated model in Chapter 6.

Gap Analysis and Rationale

In order to capture the essence of time-critical, stochastic, and multi-agent rescue missions, we identified the fundamental capabilities that any target formal model must at least support. These capabilities –ranging from explicit time representation and probabilistic branching to the integration of multi-agent interactions– form the benchmark against which existing modeling frameworks can be evaluated. Anything below that level of abstraction erases the core phenomenon we aim to analyze.

To make this concrete, we recall our running example and notions of time. We distinguish survival time and operational time (with the hazard effect already folded into survival time). At any instant there is a race between them. The instantaneous death rate for a patient naturally depends on two durations: first, the duration d_x since having been struck by the accident and, second, the duration d_y since some robot commenced the rescue treatment on the very patient, if applicable.

Such durations can be measured by two resettable clocks x and y , as known from Timed Automata [AD94], and in each moment of time, the death rate would then be an expression $f(x, y)$ incorporating a non-negative function f that is monotonically increasing in its first and monotonically decreasing in the second argument, plus the two clocks x and y . This is the schematic transition logic behind the state-transition system defined in the previous chapter. For a concrete (though not necessarily medically meaningful) example, consider the rate expression $\min(0.1x - y^2, 1)$ for the momentary death rate.

However, as mentioned, rigorously capturing this natural and intuitively straightforward scheduling problem turned out to require a combination of modelling elements we listed. A wide range of formalisms has been developed within automata theory and stochastic modeling to address parts of this challenge.

Some of them, for instance, replace rates by delay distributions such that the above rate expression $\min(0.1x - y^2, 1)$ could in principle be simulated by (re-)scheduling the death event, whenever one of the clocks x or y is reset, according to the cumulative delay distribution that is the solution to the differential equation

$$P(0) = 1 \wedge \frac{dP}{dt}(\delta) = (1 - P(\delta)) \cdot \min(0.1(x + \delta) - (y + \delta)^2, 1) , \quad (5.1)$$

where x and y refer to the values of the clocks at the moment of (re-)scheduling and δ is the delay measured from that time point.

Nonetheless, the suitability of current models for analyzing large-scale, time-critical rescue missions is not evident without a systematic comparison. The next logical step is to position our problem within the broader research landscape by examining the state of the art in automata-based and stochastic modeling formalisms.

In this chapter, we therefore turn to the literature to investigate whether current models are capable of satisfying this checklist. We systematically review prominent frameworks such as Timed Automata, Probabilistic Timed Automata, Markov Automata, Stochastic Hybrid Automata, and Interactive Markov Chains, among others, and analyze the extent to which they address the requirements of disaster-oriented rescue scenarios.

5.1 Where Existing Models Fail: A Survey

A fundamental model for stochastic dynamical systems is the *Continuous-Time Markov Chain (CTMC)* [Nor98; HH15a]. In the finite-state setting, CTMCs consist of a finite set of states and a matrix defining state-pair dependent, yet otherwise time-invariant transition rates. Due to their time-invariance, transition rates define exponential distributions for the amount of time staying in a discrete state before leaving it via the pertinent transition. Extending Markov Chains (MCs) to decision-making problems under uncertainty, *Markov Decision Processes (MDPs)* [Bel57a; Bel57b; Put05] add discrete decisions to MCs, which are to be resolved by a policy or scheduler maximizing the expected reward attached to runs.

While the aforementioned models are monolithic in that they require a single global state space, *Concurrent Stochastic Games (CSGs)* [Che+13b; BKW18] as well as (continuous-time) *Interactive Markov Chains (IMCs)* [Her02; HK09] aim at compositional agent-based modelling by featuring concurrent agents and therein differentiating explicitly between synchronous interactive and thus controllable transitions and probabilistic or stochastic autonomous transitions featuring transition probabilities or rates. Thus, IMCs come close to our modelling demands, as they allow modelling complex interaction between multiple agents featuring stochastic behaviour.

Their only shortcoming is that autonomous transitions feature location-wise constant rates, such that the catastrophe victims' continuously varying death (or recovery) rates can at best be approximated by time-wise step functions, at the price of accuracy loss and state-space explosion. Continuous-time IMCs have furthermore

been extended with probabilistic branching in transitions, yielding *Markov Automata (MA)* [EHZ10]. MA thus feature both probabilistic discrete branching and stochastic timed behaviour due to transition rates. Yet, as for IMCs, they rely on time-invariant transition rates and probability distributions, and consequently require the same approximation and encoding for our continuously varying death rates.

There are MC variants that address this coding problem by featuring time-inhomogeneous transition rates, especially *Time-Inhomogeneous Continuous-Time Markov Chains (ICTMCs)* and their interactive variant *Inhomogeneous Interactive Markov Chains (I²MCs)* [HKM08]. Delays in these model classes are generally stochastic, and they lack forced transitions controlled by timers, as necessary e.g. for modelling robots' travel times towards patients.

The modelling of strictly constrained timed behaviour is the domain of *Timed Automata (TA)* [AD94], which build upon continuous synchronous evolving clocks which can be queried in transition guards and location invariants, as well as reset independently upon transitions. Probabilistic flavours of Timed Automata, known as *Probabilistic Timed Automata (PTA)* [Spr01; Bea03; Bai+07], combine timed transitions with finite probabilistic branching across their target locations and sometimes with time-invariant transition rates.

In the aforementioned PTA variants, probabilities and, where supported, rates are clock-independent (up to normalisations necessary to recover an overall probabilistic measure [Dav+15]), but *Clock-dependent Probabilistic Timed Automata (cdPTA)* [Spr17] recently added such a feature. They extend PTA by introducing probabilistic branching, where the time-varying branch probabilities depends on the momentary valuations of local clocks. While coming close to our requirements, cdPTA still lack the ability to express stochastic transitions with clock- or time-dependent rates.

Very general frameworks permitting to model such state-dependent time inhomogeneity that follows a continuously time-variant state governed by a defined differential dynamics are *Regenerative Generalized Semi-Markov Processes (RGSM)* [HS87], *Interactive Generalized Semi-Markov Processes (IGSM)* [BG02a], and *Stochastic Hybrid Systems (SHS)* [HLS00; Hes04; LP10] in their numerous flavours. While most of them carry restrictions that prevent an encoding of our problem, like the absence of forced transitions in [HLS00; Hes04], the most general SHS variants, like [LP10], would allow an encoding at the price of translations converting clock constraints into domain boundaries and of limited analytical and computational tools due to their generality.

A comprehensive modelling framework coming close to our demands emerges when GSMP are integrated with TA, yielding *Stochastic Automata (SA)* [DKB97],

which are automata equipped with downward-running timers that can be initialized stochastically. History-dependence of the induced stochastic delays however is confined by the run-down nature of the timers, which consequently can only measure bounded durations and inject them in the thus eventually constant distributions. While this problem is mitigated by *Stochastic Timed Automata (STA)* [HHH14; Har15] by using TA-style forward-running clocks and clock constraints, these lack clock-dependency of the distributions of the delays governing autonomous transitions and as well as clock-dependency in the branch probabilities of autonomous transitions.

Stochastic Hybrid Automata (SHA) [Frä+11; Hah+13; HHH14; HH15a], finally, can be viewed as a generalization of STA by replacing clocks with continuous variables governed by ordinary differential equations or inclusions and by discontinuous updates. SHA thus combine hybrid system behaviour, as in hybrid automata [Alu+92], with stochastic sampling and delays from STA. Whether they can encode our concrete problem hinges on the exact form of sampling expressions supplied. The tool-supported variants we investigated do not seem to be able to generally encode a solution to the differential equation (5.1) as a distribution for next-state sampling, as necessary for determining the delay.

The outcome of this literature survey is synthesized in a comparative table (Table 5.1), which visualizes how the different formalisms align with the essential capabilities identified for disaster-response scenarios.

Tab. 5.1: Formal Models' Comparison in terms of Expressiveness required by Rescue Mission

Available models \ Properties	Time Inhomogeneity		Forced Transitions (Clock-Resets)	Concurrent Agents	Stochastic Transitions	Interactive Composition
	Clock/time-dependent transition rates	Clock/time-dependent transition probabilities				
Timed Automata (TA)	No, TA does not allow clock/time-dependent rates.	No, probabilities are absent without PTA extensions	Yes, transitions are forced by invariants and guards with clock resets	No, Only Concurrent-TA (CTA) which supports parallel composition for modeling concurrent systems	No, inherently lacks stochastic transitions; requires probabilistic branching	Yes, supports parallel composition with synchronization on shared actions
Probabilistic Timed Automata (PTA)	No, probabilities are discrete and not governed by rates	No, probabilities in PTA are constant and not explicitly dependent on clock values or elapsed time	Yes, transitions can be forced using guards and invariants that reset clocks, allowing control over timing constraints.	Yes, concurrent systems can be modeled using parallel composition operators, similar to Timed Automata but extended with probabilities	Yes, probabilistic transitions are enabled by associating probabilities with discrete choices	Yes, supports composition through synchronization of actions, enabling interaction among components.
Continuous Time Markov Chains (CTMC)	No, rates are constant and do not depend on time or clocks	No, probabilities are fixed and time-invariant, as governed by the constant rate matrix.	No, CTMCs do not support forced transitions or clock resets	No, CTMCs do not natively support concurrent composition	Yes, transitions occur probabilistically based on memoryless exponential distributions defined by the rate matrix.	No, standard CTMCs are not inherently interactive.
Interactive Markov Chains (IMC)	No, IMC models do not allow time-dependent rates	No, probabilities in IMC are constant and not time-dependent	No, lacks the concept of clocks, so forced transitions via clock resets are not inherently supported.	Yes, concurrency is modeled via synchronized interactions and interleaving semantics.	Yes, allows probabilistic transitions with memoryless property using exponential distributions for delays	Yes, supports synchronization on shared labels and asynchronous interleaving for independent actions.

Continued on next page

Properties Available models	Time Inhomogeneity		Forced Transitions (Clock-Resets)	Concurrent Agents	Stochastic Transitions	Interactive Composition
	Clock/time-dependent transition rates	Clock/time-dependent transition probabilities				
Time-Inhomog. CTMC (ICTMC) / I^2MC	Yes, supports time-inhomogeneous (clock/time-dependent) rates via rate matrix functions $R(t)$	Yes, transition probabilities are derived directly from rate matrices where time-dependence introduces non-uniform probabilistic branching based on system evolution.	No, does not inherently support clock resets or forced transitions; transitions are stochastic and depend on clock progress without resets	Yes, modeled via parallel agents interacting through synchronization rules, enabling modularity and concurrency	Yes, incorporates time-dependent probabilistic transitions through rate matrices dependent on the system state.	Only in I^2MC , supports synchronization on shared actions and interleaving of independent actions via interactive extensions
Regenerative GSMP (RGSMP)/IGSMP	Yes, transition rates can depend on the elapsed time, with rates dynamically evolving through semi-Markov kernels and regenerative structure	Yes, probabilities are derived dynamically from semi-Markov kernels, reflecting time-dependent branching at each regeneration point	No, lacks forced transitions or explicit clock reset mechanisms	Yes, supports concurrent systems by synchronizing processes on events defined by the regeneration points.	Yes, transitions are governed by semi-Markov processes with distributions over state changes based on elapsed time	Yes, but only in IGSMP which extends RGSMP with synchronization mechanisms, allowing systems to interact and exchange events at regeneration points
Stochastic Automata (SA)	No, rates are determined by downward-running timers, which lack continuous or clock-dependent evolution	No, Limited, probabilities are not explicitly clock-dependent, as timers eventually run out and lead to constant probabilities.	No, transitions rely on stochastic timers rather than deterministic resets	Yes, concurrent components can synchronize their transitions and operate in parallel, enabling modularity and complex systems.	Yes, stochastic delays are governed by probabilistic distributions, allowing modeling of randomness and uncertainty.	Yes, supports interaction through synchronization of transitions
Stochastic Timed Automata (STA)	No, STA does not inherently support clock/time-dependent transition rates	No, probabilities in STA are independent of clock values	Yes, clock resets allow enforcing specific timing constraints on transitions, extending deterministic and stochastic capabilities	Yes, supports modeling of concurrent agents interacting through synchronized actions and clock constraints.	Yes, extends timed automata with stochastic delays governed by probability distributions, enabling randomization in time and transitions.	Yes, supports synchronization of actions among multiple STA components, allowing concurrent systems with modular interaction.

Continued on next page

Properties Available models	Time Inhomogeneity		Forced Transitions (Clock-Resets)	Concurrent Agents	Stochastic Transitions	Interactive Composition
	Clock/time-dependent transition rates	Clock/time-dependent transition probabilities				
Clock-dependent PTA (cdPTA)	No, cdPTA does not support continuous-time stochastic transition rates. It is limited to clock-dependent branching probabilities.	Probabilistic branching only Yes, supports probabilities that directly depend on clock valuations, enabling dynamic modeling of time-sensitive probabilistic transitions.	Yes, transitions are enforced via guards and invariants that reset clocks, enabling precise control over timing behaviors.	Yes, allows modeling of concurrent systems where transitions synchronize across components while respecting clock constraints	Yes, extends PTA with clock-based probabilistic branching, where probabilities are governed by clock valuations at the time of transition.	Yes, supports synchronization of actions across multiple agents. Enables modular and concurrent system modeling with clock-based constraints.
Markov Automata (MA)	No, rates are time-invariant and based on memoryless exponential distributions; they do not vary dynamically with clock values	No, Probabilistic branching only and not dependent on clock values or elapsed time, focusing instead on stochastic delay distributions.	No, transitions are governed by stochastic timing and probabilistic branching but do not inherently include clock resets or time-dependent guards	Yes, supports modular and concurrent behaviors via synchronization, extending IMC capabilities	Yes, incorporates probabilistic branching combined with Markovian transitions using exponential distributions for delays	Yes, allows composition of automata with synchronization on shared actions, enabling concurrent and modular modeling
Stochastic Hybrid Automata (SHA)	(Limited), supports time-dependent rates governed by differential equations or inclusions, allowing dynamic rates based on continuous evolution.	Yes, transition probabilities depend on continuous state variables and time, enabling dynamic probabilistic behaviors based on hybrid states	Yes, clock resets and forced transitions are enabled via guards and invariants on hybrid variables, supporting time-sensitive transitions.	Yes, concurrent hybrid systems can be modeled with synchronized transitions among components	Yes, transitions involve stochastic delays and probabilistic branching, combining hybrid dynamics with random behavior	Yes, supports modular and concurrent modeling of systems through synchronization on shared actions

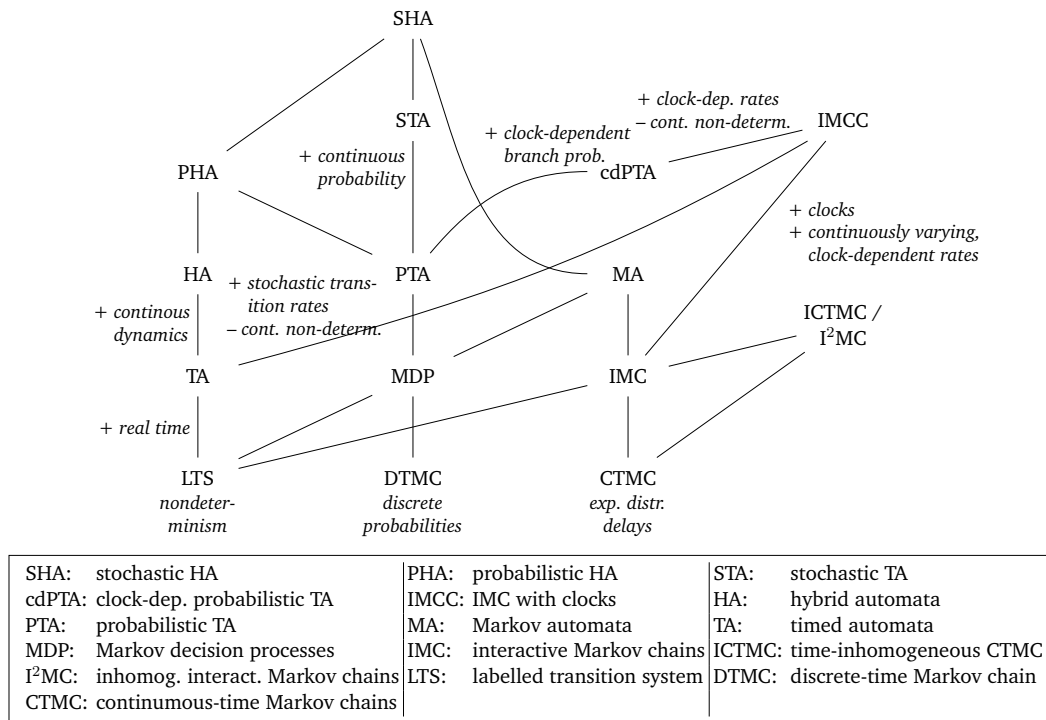


Fig. 5.1: Hierarchy of a subset of the modeling frameworks related to our IMCCs, ordered by expressiveness (based on a similar hierarchy taken from [Har15]).

5.2 In The Need of a New Formalism

Despite the breadth of formalisms developed in automata theory and stochastic modeling, the survey in this chapter together with the tabular overview highlight at a glance which models provide partial coverage of the requirements and, more importantly, reveals that none of the existing approaches explicitly support the full set of features and requirements posed by disaster-response scenarios in a unified manner. Some excel at representing strict timing or capturing probabilistic dynamics, and yet others at modeling continuous flows or multi-agent interactions. However, each of these frameworks falls short when confronted with the need to integrate all these aspects simultaneously in order to provide a faithful and analyzable representation of large-scale rescue missions.

This mismatch reflects not a flaw in the existing models, but rather their historical development. Each was designed with a particular class of systems in mind – communication protocols, embedded controllers, stochastic processes– where certain aspects were paramount while others could be abstracted away. In contrast, disaster-response missions combine urgency of time, stochastic degradation, concurrent agents, uncertainty and dynamic interactions. An attempt of eliminating any of these features puts the essence of the proposed problem in jeopardy of loss, making them a boundary case that exposes the limitations of existing approaches.

Consequently, there is a clear need for a new formalism that not only synthesizes features of existing models but also provides a conceptual framework tailored to time-critical, stochastic, multi-agent rescue scenarios. Lacking a matching model, we here set out to define a model that combines exactly

1. clocks in the style of Timed Automata [AD94] and forced transitions governed by clocks,
2. autonomous transitions that fire stochastically with clock-dependent rates, i.e. where the rates change continuously as the clocks evolve, and
3. concurrent agents.

The semantics we provide effectively is composed (stochastic selection of delay time and branch alternative –in the case of competing autonomous transitions– go hand in hand, with the branch probabilities in our case being determined by the rate relations applying at the time of the transition) in the sense of [WRÁ23], but also compositional (giving semantics to individual agents rather than just for flattened-out product automata). The combination of composed and compositional semantics induces semantic intricacies in the setting of time-inhomogeneity, which we resolve by a limit construction.

The model we introduce in the next chapter occupies a distinct position in the landscape of formalisms: it extends the expressive power of traditional approaches while remaining analyzable, and it explicitly addresses the gaps identified in our checklist. Figure 5.1 situates our model (to be introduced in the next chapter) within the hierarchy of related Automata Zoo [HH15a], clarifying both its lineage and its unique contribution.

Interactive Markov Chain with Clocks (IMCC)

6.1 Interactive Markov Chains with Clocks and Clock-dependent Rates

Rescue operations in hazardous environments exhibit two main intertwined phenomena: (i) *Synchronous actions* which e.g. a robot and a victim jointly initiating treatment, whose occurrence is governed by urgency rather than by probability, and (ii) *Time-inhomogeneous hazards* where e.g. the victim's physiological state deteriorates at an increasing rate the longer treatment is withheld.

Therefore, as extensively argued in previous chapters, modeling and solving the rescue problem requires a semantic model that reconciles clock-controlled timed behavior, autonomous stochastic transitions with time-dependent rates, and interaction.

Timed Automata (TA) model dense-time enabling conditions through real-valued clocks, yet remain purely nondeterministic. Classical Interactive Markov Chains (IMCs) capture synchronous actions by combining urgent synchronization with memoryless exponential delays, but their rates are location-constant. Extensions such as Stochastic Timed Automata (STA) and Stochastic Automata (SA) introduce probabilistic timing, but either restrict rates to piece-wise constants or replace clocks with one-shot timers, which precludes modeling continuously evolving hazard functions.

We therefore suggest Interactive Markov Chains with Clocks (IMCC) as a new model class which lift these restrictions by allowing clock-dependent rates of autonomous transitions, thereby subsuming all five requirements that rescue mission imposes (Sec. 4.4), within a single, compositional framework. They offer

1. autonomous stochastic transitions with time-variant rates, where the rates are given by expressions over clocks, thus mapping clock valuations to clock-dependent rates that change continuously as the clocks evolve,

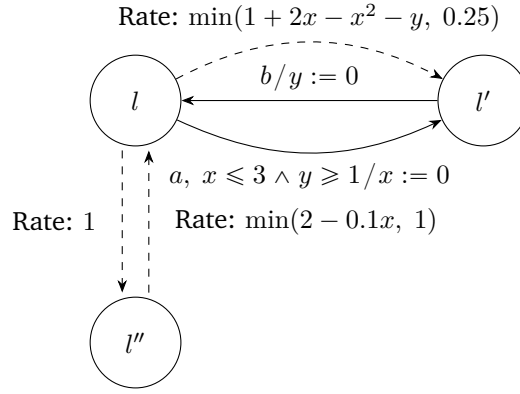


Fig. 6.1: An example of an Interactive Markov Agent with Clocks.

2. interactive transitions that synchronize on interaction symbols and are governed by clock conditions controlling their enabling,
3. real-valued clocks evolving over time and potentially being reset by transitions.

IMCCs therefore unify Interactive Markov Chains (IMCs) [Her02], timed automata (TA) [AD94], and some aspects of probabilistic TA with clock-dependent probabilities (cd-PTA) [Spr17]. They generalize both IMCs and a subset of TA with urgent transitions, as IMCs feature the two types of transitions mentioned above, albeit in a time-invariant form where both the rates of autonomous transitions and the enabling of interactive transitions are only location-dependent, while TA support clock-dependent transitions. IMCCs also generalize Stochastic Timed Automata [HHH14], which permit time-inhomogeneous yet not clock-dependent rates of autonomous transitions, and Stochastic Automata [DKB97], which employ downward-running timers instead of clocks and thus can express eventually constant clock dependencies only.

As in IMCs, IMCCs comprise a set of concurrent agents. An example of an IMCC agent is given in Fig. 6.1. In this IMCC agent, the location l can be left towards location l' in two ways: either via the interactive transition labelled with ‘ a ’ or via an autonomous transition featuring a rate depending on the clocks x and y . The interactive transition can only be taken when the clock x features a value of at most 3, the clock y is at least 1, and if all interaction partners agree on “ a ”; it is then taken as soon as it is enabled by all partners (i.e., a maximum progress semantics is adopted).

The autonomous transition fires with a rate $\min(1 + 2x - x^2 - y, 0.25)$ to l' (dashed line) depending on the current values of both clock x and y , which continuously change while residing in location l , thus inducing a corresponding continuous change in the autonomous firing rate. The total rate for autonomously leaving l at any time is $\min(1 + 2x - x^2 - y, 0.25) + 1$. When an autonomous transition from l happens,

the likelihood of ending in l' or l'' , resp., is given by the relation between the current values of the clock-dependent expressions $\min(1 + 2x - x^2 - y, 0.25)$ and 1.

In following, we formally specify IMCCs syntax (constructs and well-formed components), then its operational semantics, and finally the compositional construction. These definitions establish the mathematical groundwork on which later chapters build the reachability, decidability and approximability analysis.

6.2 Formal Definition of IMCC: Syntax and Semantics

6.2.1 Structure of an IMCC

Formally, an IMCC is a finite set $\{A_1, \dots, A_n\}$ of agents plus a finite alphabet Σ for interaction labels and a finite set C of globally visible clocks. Each agent $A_i = (L_i, ls_i, \Sigma_i, aT_i, iT_i, R_i, cond_i, r_i)$ comprises

1. a finite set L_i of locations that the agent can reside in, plus a dedicated start location $ls_i \in L_i$,
2. a local interaction alphabet $\Sigma_i \subseteq \Sigma$ naming the labels from the joint alphabet Σ that the particular agent synchronizes on,
3. a (possibly empty) finite set $aT_i \subseteq L_i \times L_i$ of autonomous transitions that do not loop immediately, i.e. satisfy $(l, l') \in aT_i \implies l \neq l'$,
4. a (possibly empty) finite set $iT_i \subseteq L_i \times \Sigma_i \times L_i$ of interactive transitions, where we require determinism: $(l, a, l') \wedge (l, a, l'') \implies l' = l''$,
5. a mapping $R_i : iT_i \rightarrow 2^{C_i}$ denoting the clocks to be reset on interactive transitions,
6. a mapping $cond_i : iT_i \rightarrow 2^{ClockVal}$, where $ClockVal = \{v \mid v : C \rightarrow \mathbb{R}_{\geq 0}\}$, stating the guard conditions for interactive transitions and satisfying that for each $(l, a, l') \in iT_i$, the guard $cond_i(l, a, l')$ is non-empty, closed, and *non-punctual* in that the closure of its interior $cl(int(cond_i(l, a, l')))$ equals $cond_i(l, a, l')$,
7. a Lipschitz-continuous function $r_i : aT_i \rightarrow ClockVal \rightarrow \mathbb{R}_{\geq \underline{r}}$, where $\underline{r} \in \mathbb{R}_{>0}$ and $\mathbb{R}_{\geq \underline{r}} = \{x \in \mathbb{R} \mid x \geq \underline{r}\}$, providing clock-dependent transition rates for the autonomous transitions.

Note that there can be multiple interactive transitions (with different labels) between the same pair of locations, yet only a single autonomous one. To the latter, a clock-dependent rate is associated by the mapping r_i . Note that the rates are uniformly bounded from below by $\underline{r} > 0$, yet not from above.

We say that an IMCC is *well-formed* if all intersections of guard conditions pertaining to the same interaction label in different agents are non-punctual, i.e. iff $\text{int}(\text{cond} \cap \text{cond}') \neq \emptyset$ for any $a \in \Sigma$, any agents $A_i \neq A_j$ with $a \in \Sigma_i \cap \Sigma_j$, and any $\text{cond} = \text{cond}_i(l_i, a, l'_i)$ and $\text{cond}' = \text{cond}_j(l_j, a, l'_j)$ for some $l_i, l'_i \in L_i$ and $l_j, l'_j \in L_j$. We will in the remainder assume that all IMCC we deal with are well-formed and will not mention this side condition explicitly. The insistence on non-punctuality in a setting of urgency of interactive transitions, which we enforce, may seem redundant or inconsistent, but actually is necessary for approximability results based on a weak form of time discretization, which we also employ.

6.2.2 Discrete-Time Semantics of an IMCC

IMCCs are continuous-time dynamical systems reconciling forced clock-dependent behaviour with rate-based autonomous transition dynamics and probabilistic branching happening in time instants when autonomous transitions are taken. To unify these stochastic and probabilistic elements in a single semantic model, we take a detour and first define a family of approximations that operate over continuous time, yet take autonomous transitions only at discrete time instants $t \in \Delta\mathbb{N}$, where $\Delta \in \mathbb{R}_{>0}$ is a time granularity defining the particular approximation and $\Delta\mathbb{N}$ signifies non-negative integer multiples of Δ . These approximations allow us to replace the time-dependent stochastic autonomous behaviour by probabilistic branching happening at times $\Delta\mathbb{N}$, thus constituting a Probabilistic Timed Automaton [Spr01]. The continuous-time stochastic behaviour is recovered from these Δ -approximations by building the limit for $\Delta \rightarrow 0$.

Timed probabilistic approximation of agent semantics.. We start by defining an approximating probabilistic semantics for an individual agent A_i , which essentially constitutes a Probabilistic Timed Automaton (PTA, [Spr01]), yet is stated here in the form of a transition system rather than a reduction to PTA. Therefore, let $\Delta \in \mathbb{R}_{>0}$ be a time granule. We then define three different transition relations: the relation $\xrightarrow{a, \Delta}_i$ for instantaneous interaction on the symbol $a \in \Sigma_i$, the relation $\xrightarrow{\delta, \Delta}_i^p$ for time passage with a duration $\delta \in \mathbb{R}_{>0}$ and a probability p , and the relation $\xrightarrow{\tau, \Delta}_i^p$ for instantaneous autonomous action occurring with probability p , with the latter happening at time points $t \in \Delta\mathbb{N}$ only. These transitions range over states from $L_i \times \text{ClockVal} \times \mathbb{R}_{\geq 0}$, with a triple $(l, v, t) \in L_i \times \text{ClockVal} \times \mathbb{R}_{\geq 0}$ representing that at absolute time t the agent resides in the location l and the (global) clock valuation is v . The three transition relations are defined as follows.

Given a location $l \in L_i$ and a clock valuation $v \in \text{ClockVal}$ plus a current time $t \in \Delta\mathbb{N}$,

- (l, v) can at time t engage into an instantaneous interactive transition labelled by $a \in \Sigma_i$ and leading to location $l' \in L_i$ and clock valuation $v' \in \text{ClockVal}$, iff the three conditions

1. $(l, a, l') \in iT_i$,
2. $v \in \text{cond}_i(l, l')$, and
3. $v' = v \oplus R_i(l, l') := 0$,

$$\text{where } (v \oplus R := 0)(c) = \begin{cases} v(c) & \text{iff } c \notin R, \\ 0 & \text{iff } c \in R, \end{cases}$$

hold. We denote this fact by $(l, v, t) \xrightarrow{a, \Delta}_i (l', v', t)$. Note that time does not change as the transition is instantaneous, but that clock valuations may and i.g. will change due to clock resets. Please do also note that these forced (by clock progress) transitions are allowed to happen any time, also between time instants $\Delta\mathbb{N}$.

- while staying in location l , (l, v, t) lets time pass by $\delta \in \mathbb{R}_{>0}$ with probability $p \in]0, 1[$ and then reaches state $(l, v + \delta, t + \delta) \in L_i \times \text{ClockVal} \times \mathbb{R}_{>0}$, denoted by $(l, v, t) \xrightarrow{\delta, \Delta^p}_i (l, v + \delta, t + \delta)$, iff $p = x(\delta)$ for the solution $x : \mathbb{R}_{\geq 0} \rightarrow [0, 1]$ of the initial value problem

1. $x(0) = 1$ and
2. $\frac{dx}{dt}(t) = \left(- \sum_{l'' \mid (l, l'') \in aT} r(l, l'')(v + t) \right) \cdot x(t)$ for all $t \in \mathbb{R}_{\geq 0}$,

where $\forall c \in C : (v + t)(c) = v(c) + t$.

Note that x is bounded by $[0, 1]$ and strictly monotonically falling due to the rates being strictly positive, with 0 being approached only asymptotically. Consequently $p = x(\delta) \in]0, 1[$. Please do also note that time passage transitions are allowed to start and end any time, also between time instants $\Delta\mathbb{N}$, in order to permit forced transitions any time.

- (l, v, t) can autonomously branch to state $(l', v, t) \in L_i \times \text{ClockVal} \times \mathbb{R}_{\geq 0}$ with probability $p \in]0, 1[$, denoted $(l, v, t) \xrightarrow{\tau, \Delta^p}_i (l', v, t)$, iff the following three conditions hold:

1. $t \in \Delta\mathbb{N}$, i.e. branching is synchronized to the time discretization $\Delta\mathbb{N}$,
2. $(l, l') \in aT_i$,
3. $p = \frac{r(l, l')(v)}{\sum_{l'' \mid (l, l'') \in aT} r(l, l'')(v)} \cdot (1 - x(\Delta))$, where x is the initial value problem from above.

Note that $p > 0$ as $r_i(l, l'') \geq \underline{r} > 0$. Furthermore, $p \leq x(\Delta) \leq 1 - \Delta\underline{r}$, as $\sum_{l'' \mid (l, l'') \in aT} r(l, l'')(v + t) \geq r(l, l')(v + t) \geq \underline{r}$.

In condition 3, the factor $1 - x(\Delta)$, representing the likelihood of taking the autonomous jump in the next Δ time units, recovers a composed semantics [WW22] while permitting stuttering within delay transitions for compositionality.

Timed probabilistic approximation of IMCC semantics. For an IMCC system $\{A_1, \dots, A_n\}$, interactive transitions take preference over passage of time and autonomous transitions, i.e., we adopt a maximum progress assumption. We therefore define the above three transition relations on tuples of agent locations plus a clock valuation and current time $t \in \mathbb{R}_{\geq 0}$ as follows: For $(l_1, \dots, l_n) \in L_1 \times \dots \times L_n$ and $v \in \text{ClockVal}$ and $t \in \mathbb{R}_{\geq 0}$, we define

- $((l_1, \dots, l_n), v, t) \xrightarrow{a, \Delta}_{A_1 \parallel \dots \parallel A_n} ((l'_1, \dots, l'_n), v', t)$ iff
 1. $l_i = l'_i$ for all $i \in \{1, \dots, n\}$ satisfying $a \notin \Sigma_i$,
 2. $(l_i, v, t) \xrightarrow{a, \Delta}_i (l'_i, v \oplus R_i(l_i, l'_i) := 0, t)$ for all $i \in \{1, \dots, n\}$ satisfying $a \in \Sigma_i$,
 3. $v' = v \oplus \left(\bigcup_{i|a \in \Sigma_i} R_i(l_i, l'_i) \right) := 0$.

Thus, all partners having the symbol a in their interaction alphabet have to agree on a transition labelled a . Their clock resets are then collected and combined.

- $((l_1, \dots, l_n), v, t) \xrightarrow{\delta, \Delta^p}_{A_1 \parallel \dots \parallel A_n} ((l_1, \dots, l_n), v + \delta, t + \delta)$ iff
 1. $(l_i, v, t) \xrightarrow{\delta, \Delta^p}_i (l_i, v + \delta, t + \delta)$ for all $i \in \{1, \dots, n\}$,
 2. $p = \prod_{i=1}^n p_i$, and
 3. for all interaction labels $a \in \Sigma$, no a transition ever is enabled in any state $((l_1, \dots, l_n), v + \varepsilon, t + \varepsilon)$ with $0 \leq \varepsilon \leq \delta$.

Thus, time passage is only admitted as long as no interactive transition is enabled, which otherwise takes precedence (maximum progress assumption). The likelihood of an overall idling for δ time units is the product of the individual agents' likelihoods for idling δ time.

- $((l_1, \dots, l_n), v) \xrightarrow{\tau, \Delta^p}_{A_1 \parallel \dots \parallel A_n} ((l'_1, \dots, l'_n), v')$ iff there is $j \in \{1, \dots, n\}$ such that
 1. $l_i = l'_i$ for all $i \in \{1, \dots, n\}$ satisfying $i \neq j$,
 2. $(l_j, v, t) \xrightarrow{\delta, \Delta^p}_j (l'_j, v, t)$,
 3. for no $a \in \Sigma$, an a transition is enabled in $((l_1, \dots, l_n), v)$.

Thus, autonomous transitions happen independently. They can happen any time when no interactive transition is enabled, which otherwise takes precedence. Note that autonomous transitions of IMCCs inherit their synchronisation to time instants $t \in \Delta\mathbb{N}$ from the agent semantics.

We will in the sequel also write $(\vec{l}, v) \xrightarrow{a, \Delta^1}_{A_1 \parallel \dots \parallel A_n} (\vec{l}', v')$ for the non-probabilistic interactive transition $(\vec{l}, v) \xrightarrow{a, \Delta}_{A_1 \parallel \dots \parallel A_n} (\vec{l}', v')$ to unify notation and thereby simplify subsequent definitions.

A (finite) Δ -run of a system is a finite alternating sequence

$$\pi = \langle (\vec{l}_0, v_0, t_0), p_1, act_1, (\vec{l}_1, v_1, t_1), \dots, (\vec{l}_{k-1}, v_{k-1}, t_{k-1}), p_k, act_k, (\vec{l}_k, v_k, t_k) \rangle$$

of states $(\vec{l}_j, v_j, t_j) \in (\prod_{i=1}^n L_i) \times \text{ClockVal} \times \mathbb{R}_{\geq 0}$, probabilities $p_j \in]0, 1]$, and actions $act_j \in \Sigma \cup \mathbb{R}_{>0} \cup \{\tau\}$ such that any two successive states are connected by the respective transition (mixed from any of the three types of transitions), i.e. such that $\forall j \in \{1, \dots, k\} : (\vec{l}_{j-1}, v_{j-1}) \xrightarrow{act_j, \Delta^{p_j}}_{A_1 \parallel \dots \parallel A_n} (\vec{l}_j, v_j)$. We call the Δ -run π *anchored* if $\vec{l}_0 = (ls_1, \dots, ls_n)$, $t_0 = 0$ and $\forall c \in C : v_0(c) = 0$. We call the Δ -run π *stutter-free* if it does not contain successive time-passage transitions, i.e. iff it satisfies $\forall j \in \{1, \dots, k-1\} : act_j \in \mathbb{R}_{>0} \implies act_{j+1} \notin \mathbb{R}_{>0}$.

The *probability of a Δ -run* (whether it is stutter-free, i.e. does not contain successive time-passage transitions, or not) is given by the product of all the probabilities attached to its individual time-passing and autonomous transitions, i.e. $P(\pi) = \prod_{j=1}^k p_j$, with this definition being possible due to our convention of assigning probability 1 to interactive transitions. The *duration of a Δ -run* is given by the sum of all the delays attached to its individual time-passing transitions, i.e. $\text{dur}(\pi) = \sum_{j \in \{j=1, \dots, k \mid act_j \in \mathbb{R}_{>0}\}} act_j = t_k - t_0$. The *trace* $tr \in \Sigma^*$ associated to a Δ -run π is the sequence of interaction labels $\sigma_i \in \Sigma$ occurring on its interactive transitions; we then also say that the Δ -run π is labelled with tr . The *extended trace* $tr \in (\Sigma \cup \{\tau\})^*$ associated to a Δ -run π is the sequence of labels $\sigma_i \in \Sigma \cup \{\tau\}$ occurring on its interactive and autonomous transitions.

We call a state $(\vec{l}', v', t')$ Δ -reachable from a state $(\vec{l}, v, t)$ iff there is a Δ -run π starting in $(\vec{l}, v, t)$ and ending in $(\vec{l}', v', t')$. We call a pair $(\vec{l}', v')$ of a location $\vec{l}'$ and a clock valuation v' Δ -reachable iff there is an anchored Δ -run reaching $(\vec{l}', v', t')$ for some $t' \in \mathbb{R}_{\geq 0}$. We call $(\vec{l}', v')$ Δ -reachable in d time units (or Δ -reachable from a state $(\vec{l}, v, t)$ in d time units) iff there is an anchored Δ -run π reaching $(\vec{l}', v', t')$ with $t' \leq d$ (or reaching $(\vec{l}', v', t')$ with $t' - t \leq d$ from $(\vec{l}, v, t)$, resp.).

Lemma 6.1 (Stutter invariance) $(\vec{l}, v, t) \xrightarrow{\delta, \Delta^p}_{A_1 \parallel \dots \parallel A_m} (\vec{l}, v + \delta, t + \delta)$ iff for each $\delta_1, \delta_2 \in \mathbb{R}_{>0}$ such that $\delta = \delta_1 + \delta_2$ there are $p_1, p_2 \in]0, 1[$ with $p = p_1 p_2$ such that $(\vec{l}, v, t) \xrightarrow{\delta_1, \Delta^{p_1}}_{A_1 \parallel \dots \parallel A_m} (\vec{l}, v + \delta_1, t + \delta_1)$ and $(\vec{l}, v + \delta_1, t + \delta_1) \xrightarrow{\delta_2, \Delta^{p_2}}_{A_1 \parallel \dots \parallel A_m} (\vec{l}, v + \delta, t + \delta)$.

Proof. The statement follows immediately from the structure of the differential equation underlying time passage, plus associativity of multiplication. $\square$

Due to Lemma 6.1, we can w.l.o.g. concentrate on stutter-free runs in the remainder. In our application we furthermore only encounter IMCCs which are strongly time-progressive in the following sense:

Definition 6.2 (Time-progressive IMCC) An IMCC is called *time-progressive* iff each location cycle in its transition graph contains at least one interactive transition which requires at least one clock to be larger than a strictly positive constant $c > 0$,

and that clock is reset along that same cycle. We call an IMCC c -progressive iff $c > 0$ is a joint constant of the above type for all cycles. Each strongly time-progressive IMCC is c -progressive for some $c > 0$ and vice versa.

An interesting consequence of c -progressivity is that time-bounded paths can be enumerated.

Lemma 6.3 *Let the IMCC be c -progressive. There are finitely many extended traces corresponding to Δ -runs of duration d originating from any state $(\vec{l}, v, t)$. Their number is bounded by $(|\Sigma| + 1)^{\lceil \frac{d}{c} \rceil (|L| - 1)}$, where $L = \prod_{i=1}^n L_i$ is the product location space of the IMCC.*

Proof. The length of each location cycle in the location graph is bounded by $|L|$, and c -progressivity enforces any cycle to take at least $c > 0$ time units to complete. Hence, at most $\lceil \frac{d}{c} \rceil (|L| - 1)$ transitions can occur in d time units. As the cardinality of the extended label set $\Sigma \cup \{\tau\}$ is $|\Sigma| + 1$, the result follows. $\square$

6.2.3 Continuous-Time Semantics of an IMCC and Approximability

Our problems of interest are to compute the probability of Δ -reaching a designated location or a designated set of locations within a given time bound. This notion however only makes sense if any non-determinism in the IMCC, i.e. any concurrent availability of different interactive actions, is resolved (e.g., by a policy/strategy). We call an IMCC *deterministic* iff for all its states $(\vec{l}, v, t) \in L \times \text{ClockVal} \times \mathbb{R}_{\geq 0}$ there is at most one interactive transition label a and one target location $\vec{l}'$ having its guard condition $\bigcap_{i|a \in \Sigma_i} \text{cond}_i(\vec{l}_i, a, \vec{l}'_i)$ being satisfied.

Definition 6.4 (Time-bounded Δ -reachability) *Given a deterministic IMCC I , a target location $\vec{t}l \in L$, and a time bound $d \in \mathbb{R}$, the time-bounded Δ -reachability problem is to determine the cumulative probability $P_{\Delta}^d(\vec{t}l)$ of Δ -reaching the location $\vec{t}l$ via any anchored Δ -run of duration strictly less than d . This definition straightforwardly generalizes to sets $T \subseteq L$ of target locations, where $P_{\Delta}^d(T) = \sum_{\vec{t}l \in T} P_{\Delta}^d(\vec{t}l)$.*

The previous lemma shows that this problem is well-defined for c -progressive IMCCs and implies that it is computationally solvable :

Theorem 6.5 *Let the IMCC I be deterministic and c -progressive. Given a target location $\vec{t}l \in L$ and a time bound $d \in \mathbb{R}$, the Δ -reach probability $P_{\Delta}^d(\vec{t}l)$ is uniquely*

defined. $P_{\Delta}^d(\vec{tl})$ can be approximated computationally to arbitrary precision if the guard conditions, $cond$, are described in a decidable logic and if the rate assignments can be computed to arbitrary precision.

Proof sketch. According to Lemma 6.3, there are only finitely many extended traces corresponding to anchored Δ -runs of duration at most d . Each of these extended traces in turn corresponds to a finite number of Δ -runs possibly underlying it, as

1. there is only a finite number of autonomous transition alternatives that each τ transition appearing in the extended trace can map to,
2. each of these autonomous transitions can only happen at a finite number of different time instants, as autonomous transitions generally can only happen at times in $\Delta\mathbb{N} \cap [0, d]$,
3. interactive transitions are urgent, and thus happen either at the first time instant where their clock-related guard conditions are met or immediately after an autonomous transition reaching their source location, whichever happens earlier.¹

As each of these finitely many Δ -runs has a defined probability assigned by the IMCC semantics, $P_{\Delta}^d(\vec{tl})$ is also well-defined as the sum of these run probabilities.

In order to approximate $P_{\Delta}^d(\vec{tl})$ computationally, we need to

1. compute ϵ -accurate approximations to the time points at which guard conditions become true² due to time progress,
2. compute approximate solutions $x(\Delta)$ to the initial value problems of the form $x(0) = 1 \wedge \frac{dx(t)}{dt} = -r(v + t)x(t)$, with r being Lipschitz (with known constant) and itself approximable to arbitrary precision, given an approximation of v .

If the guard conditions are formulated in a decidable logic, the first problem can be solved by bifurcation search (though there clearly are better means for specific choices of logics). The second problem is a standard problem of computational analysis that can be solved effectively by type-2 computability [Wei00] (and can be approximated to reasonable degree using standard floating-point computer arithmetic). □

¹Note that interactive transitions may depend on prior autonomous transitions, which may be needed to reach their source location, and thus are not fully determined by clock progress and interactions only. With finitely many options for prior autonomous actions and their timing, the alternatives however can be enumerated.

²Note that due to urgency, we are only interested in the point of time when the guard becomes true, as the transition then is taken immediately.

We furthermore remark that IMCCs are closed under parallel composition in the sense that for each IMCC comprising a parallel composition $A_1 \parallel \dots \parallel A_n$ of agents, there is an equivalent IMCC with just one parallel component A :

Lemma 6.6 (Product IMCC) *Consider a family of concurrent interacting agents $A_i = (L_i, \Sigma_i, aT_i, iT_i, R_i, \text{cond}_i, r_i)$, with $i = \{1, \dots, n\}$. Define its product agent $A = (L, \Sigma, aT, iT, R, \text{cond}, r)$ by*

- $L = \prod_{i=1}^n L_i$,
- $\Sigma = \bigcup_{i=1}^n \Sigma_i$,
- $aT = \{((l_1, \dots, l_n), (l'_1, \dots, l'_n)) \in L \times L \mid \exists j \in \{1, \dots, n\} : (l_j, l'_j) \in aT_i \wedge \forall j' \in \{1, \dots, n\} : (j \neq j' \implies l_{j'} = l'_{j'})\}$, and we call an autonomous transition t in A that changes the j -th state an A_j -transition and denote its underlying transition in A_j by t_{A_j} ,
- $iT = \{((l_1, \dots, l_n), a, (l'_1, \dots, l'_n)) \in L \times \Sigma \times L \mid \exists j \in \{1, \dots, n\} : (a \Sigma_j \implies (l_j, a, l'_j) \in iT_i \wedge (a \notin \Sigma_j \implies l_j = l'_j))\}$, and we call an interactive transition in A that is labelled with A an a -transition and denote its underlying transition in A_j by t_{A_j} ,
- $R(t) = R_i(t_{A_i})$ for all A_i -transitions t and $R(t) = \bigcup_{j \in \{1, \dots, n\}, a \in \Sigma_j} R_j(t_{A_j})$ if t is an a -transition,
- $\text{cond}(\vec{l}, a, \vec{l}') = \bigcap_{j \in \{1, \dots, n\}, a \in \Sigma_j} \text{cond}_j(l_j, a, l'_j)$ f.e. transition $(\vec{l}, a, \vec{l}') \in iT$,
- $r((l_1, \dots, l_n), (l_1, \dots, l_{i-1}, l'_i, l_{i+1}, \dots, l_n)) = r_i(l_i, l'_i)$.

Then the product IMCC (A) consisting of the single agent A only is equivalent to the IMCC $(A_1 \parallel \dots \parallel A_n)$ in the following sense: Let

$$\pi = \langle (\vec{l}_0, v_0, t_0), p_1, \text{act}_1, (\vec{l}_1, v_1, t_1), \dots, (\vec{l}_{k-1}, v_{k-1}, t_{k-1}), p_k, \text{act}_k, (\vec{l}_n, v_k, t_k) \rangle$$

and

$$\pi' = \langle ((\vec{l}_0), v_0, t_0), p_1, \text{act}_1, ((\vec{l}_1), v_1, t_1), \dots, ((\vec{l}_{k-1}), v_{k-1}, t_{k-1}), p_k, \text{act}_k, ((\vec{l}_n), v_k, t_k) \rangle .$$

Then π is a Δ -run of the IMCC $(A_1 \parallel \dots \parallel A_n)$ iff π' is a Δ -run of the IMCC (A).

Proof. Immediate from the definitions by one-to-one comparison of the transitions.

□

This closure under parallel composition and the resulting ‘flattening’ operation will allow us to lift results from related computational models, like Stochastic Hybrid Systems, to IMCCs.

Recovering the continuous-time semantics

The intuitive idea for establishing a continuous-time semantics, given its Δ -granular approximations, is to define the continuous-time semantics as the limit of the approximations for $\Delta \rightarrow 0$.

Definition 6.7 (Time-bounded continuous reachability) *Given a deterministic IMCC I , a target location $\vec{tl} \in L$, and a time bound $d \in \mathbb{R}$, the time-bounded reachability problem is to determine the probability $P^d(\vec{tl})$ of reaching the location $\vec{tl}$, where $P^d(\vec{tl}) = \lim_{\Delta \rightarrow 0} P_{\Delta}^d(\vec{tl})$.*

This definition does however require existence of the limit $\lim_{\Delta \rightarrow 0} P_{\Delta}^d(\vec{tl})$, which is not immediately obvious.

We start from showing a special case implying the convergence for a certain geometric sequence of granularities Δ , namely the sequence $2^{-n}\Delta$ for arbitrary base $\Delta > 0$, followed by the approximation band for general case.

Lemma 6.8 (Approximation) *Let $lc \in \mathbb{R}_{\geq 0}$ be a common Lipschitz constant for all the rate assignments $r(\vec{l}, \vec{l}')$, where lc is determined w.r.t. the L_{∞} norm on $ClockVal$, ie. using the norm $\|v - v'\|_{\infty} = \max_{c \in C} |v(c) - v'(c)|$. Let $\Delta = 2\Delta' \in \mathbb{R}_{> 0}$ and $v, v' \in ClockVal$ with $\|v - v'\|_{\infty} \leq \varepsilon$. Then for any state $(\vec{l}, v, t) \in L \times ClockVal \times \mathbb{R}_{\geq 0}$ and any duration $\delta \in \mathbb{R}_{\geq 0}$, the following properties hold:*

1. *The probability p of letting time pass by δ from $(\vec{l}, v, t) \xrightarrow{\delta, \Delta}_{A_1 \dots \| A_m}^p$ is independent of Δ , i.e. agrees for granularity $\Delta = 2\Delta'$ and Δ' .*
2. *The probability p of adopting some autonomous transition within δ time from $(\vec{l}, v, t) \xrightarrow{\tau, \Delta}_{A_1 \dots \| A_m}^p$ is independent of Δ , i.e. agrees for granularity $\Delta = 2\Delta'$ and Δ' .*
3. *The probability p of adopting a specific autonomous transition $(\vec{l}, \vec{l}')$ within δ time from $(\vec{l}, v, t) \xrightarrow{\tau, \Delta}_{A_1 \dots \| A_m}^p$ is changing by at most $\pm \frac{lc\Delta}{2\varepsilon}$ when exchanging granularity Δ for Δ' .*

Proof. The first property is immediate, as the definition of time passage in the Δ -granular semantics makes no references to the granularity Δ . The second property follows from the fact that the probability p_s of waiting till time $s \in \Delta\mathbb{N}$ with $s \geq t$

and then adopting a specific autonomous transition $s = (\vec{l}, \vec{l}')$ in the Δ -granular semantics is

$$p_s = x(s) \frac{r(\vec{l}, \vec{l}')(v + s - t)}{\sum_{\vec{l}''; (\vec{l}, \vec{l}'') \in aT} r(\vec{l}, \vec{l}'')(v + s - t)} \cdot (1 - y(\Delta)),$$

where x and y are the solution of the initial value problems

$$\begin{aligned} x(0) = y(0) &= 1, \\ \frac{dx}{dt}(s) &= - \left(\sum_{\vec{l}''; (\vec{l}, \vec{l}'') \in aT} r(\vec{l}, \vec{l}'')(v + s - t) \right) x, \\ \frac{dy}{dt}(s') &= - \left(\sum_{\vec{l}''; (\vec{l}, \vec{l}'') \in aT} r(\vec{l}, \vec{l}'')(v + s' - s) \right) y. \end{aligned}$$

Consequently, the probability of adopting any autonomous transition in $\{(\vec{l}, \vec{l}') \in aT\}$ is

$$p = x(s)(1 - y(\Delta)).$$

Now observe that the Δ -granular semantics does one transition in the time interval $[s, s + \Delta[$ while the Δ' -granular does two in the same time interval, namely one at time s and one at time $s + \Delta' = s + \frac{\Delta}{2}$. The latter two have the probabilities $p'_1 = x(s)(1 - y(\Delta'))$ and $p'_2 = x(s + \Delta')(1 - y'(\Delta'))$, resp., where y' is the solution of the initial value problem

$$\begin{aligned} y'(0) &= 1, \\ \frac{dy'}{dt}(s') &= - \left(\sum_{\vec{l}''; (\vec{l}, \vec{l}'') \in aT} r(\vec{l}, \vec{l}'')(v + s' - (s + \Delta')) \right) y'. \end{aligned}$$

But then

$$\begin{aligned} p'_1 + p'_2 &= x(s)(1 - y(\Delta')) + x(s + \Delta')(1 - y'(\Delta')) \\ &= x(s)(1 - y(\Delta')) + x(s)y(\Delta')(1 - y'(\Delta')) \\ &= x(s)(1 - y(\Delta')y'(\Delta')) \\ &= x(s)(1 - y(2\Delta')) \\ &= x(s)(1 - y(\Delta)) = p \end{aligned}$$

since $x(s + \Delta') = x(s)y(\Delta')$ and $y(\Delta')y'(\Delta') = y(2\Delta')$. As this holds for each time interval $[\Delta n, \Delta(n + 1)[\subseteq [t, t + \delta]$, the second claim follows.

For the third property observe that a specific autonomous transition $(\vec{l}, \vec{l}')$ within

the Δ -granular semantics within time $[\Delta n, \Delta(n+1)[$ gets a share of the above probability p of

$$s = \frac{r(\vec{l}, \vec{l}')(v + n\Delta - t)}{\sum_{\vec{l}'': (\vec{l}, \vec{l}'') \in aT} r(\vec{l}, \vec{l}'')(v + n\Delta - t)}$$

of the total probability of all the autonomous transition at time Δn , while in the Δ' -granular semantics it gets the two shares

$$\begin{aligned} s'_1 &= \frac{r(\vec{l}, \vec{l}')(v + n\Delta - t)}{\sum_{\vec{l}'': (\vec{l}, \vec{l}'') \in aT} r(\vec{l}, \vec{l}'')(v + n\Delta - t)} = s \\ s'_2 &= \frac{r(\vec{l}, \vec{l}')(v + n\Delta + \Delta' - t)}{\sum_{\vec{l}'': (\vec{l}, \vec{l}'') \in aT} r(\vec{l}, \vec{l}'')(v + n\Delta + \Delta' - t)} \end{aligned}$$

of p'_1 and p'_2 , resp. As nominator and denominator of s'_2 are at least $\underline{r} > 0$ and due to the Lipschitz continuity of r with Lipschitz constant lc , we have that $s'_2 \in s'_1 \pm \frac{lc\Delta'}{\underline{r}} = s'_1 \pm \frac{lc\Delta}{2\underline{r}}$, as $\|(v + n\Delta - t) - (v + n\Delta + \Delta' - t)\|_\infty = \Delta'$. From this, the property follows by $p'_1 + p'_2 = p$. $\square$

Note that the probabilities of all autonomous transitions are thus homogeneously multiplied by a factor of $1 \pm \frac{lc\Delta}{2\underline{r}}$.

Lemma 6.9 (Local multiplicative stability) *Let I be a deterministic, progressive IMCC with horizon ($d > 0$). Assume each autonomous rate $r(\vec{\ell}, \vec{\ell}') : [0, M]^k \rightarrow \mathbb{R}_{\geq 0}$ is Lipschitz continuous with lc in L_∞ clock norm, and whenever an autonomous transition race/ branching is enabled at a $(\vec{\ell}, v)$, we have $\hat{r}_{\vec{\ell}}(v) := \sum_{\vec{\ell}'} r(\vec{\ell}, \vec{\ell}')(v) \geq \underline{r} > 0$. Let $\Delta = 2\Delta' > 0$ be sufficiently small. For an arbitrary time point $\delta \geq 0$, and each outgoing branch j of the autonomous race enabled at $(\vec{\ell}, v)$, let P_j^Δ and $P_j^{\Delta'}$ denote the branch allocation probabilities induced over the time windows $[\delta, \delta + \Delta]$ and $[\delta, \delta + \Delta']$, respectively. Then, uniformly in δ , the following bound holds:*

$$\frac{P_j^{\Delta'}}{P_j^\Delta} \in \left[1 - \frac{\alpha\Delta}{2\beta}, 1 + \frac{\alpha\Delta}{2\beta}\right] \quad \text{with} \quad \alpha := \frac{lc}{\underline{r}} \quad \& \quad \beta := \min_j \frac{r_{jmin}}{\hat{r}_{max}} > 0$$

Moreover, the ratio bound implies an absolute bound:

$$|P_j^{\Delta'} - P_j^\Delta| \leq \frac{\alpha\Delta}{2\beta} P_j^\Delta \leq \frac{\alpha\Delta}{2\beta} p_j^\Delta,$$

since $P_j^\Delta = s_j^\Delta p_j^\Delta \leq p_j^\Delta$.

and the bound is independent of the absolute time position δ .

Proof. Lemma 6.8 yields the *absolute* perturbation $|s_j^{\Delta'} - s_j^{\Delta}| \leq \frac{\alpha\Delta}{2}$. The probability allocation to branch j in this slice is: $P_j^{\Delta} = s_j^{\Delta} \cdot p_j^{\Delta}$, and $P_j^{\Delta'} = (s_1^{\Delta'})_j \cdot (p_1^{\Delta'})_j + (s_2^{\Delta'})_j \cdot (p_2^{\Delta'})_j$. Since $s_1^{\Delta'} = s^{\Delta}$ and $p_1^{\Delta'} + p_2^{\Delta'} = p^{\Delta}$, we have:

$$P_j^{\Delta'} = s_j^{\Delta} \cdot p_j^{\Delta} + ((s_2^{\Delta'})_j - s_j^{\Delta})(p_2^{\Delta'})_j$$

Therefore,

$$|P_j^{\Delta'} - P_j^{\Delta}| \leq ((s_2^{\Delta'})_j - s_j^{\Delta})(p_2^{\Delta'})_j \leq \frac{\alpha\Delta}{2} (p_2^{\Delta'})_j \leq \frac{\alpha\Delta}{2} p_j^{\Delta}.$$

Consequently,

$$\left| \frac{P_j^{\Delta'}}{P_j^{\Delta}} - 1 \right| \leq \frac{\frac{\alpha\Delta}{2} p_j^{\Delta}}{P_j^{\Delta}} = \frac{\frac{\alpha\Delta}{2} p_j^{\Delta}}{s_j^{\Delta} p_j^{\Delta}} = \frac{\alpha\Delta}{2} \frac{1}{s_j^{\Delta}}$$

Now, considering a lower bound on the share of an available autonomous transition $s_j^{\Delta} = \frac{r_j(v)}{r_{\ell}(v)}$, as well as Lipschitz continuity holding a rate lower bound and guard stability which prevents rate blow up, we assume there exists $\beta > 0$ such that

$$s_j^{\Delta} \geq \beta \quad \text{for all enabled } j \text{ and all admissible } v.$$

Therefore, we conclude:

$$\left| \frac{P_j^{\Delta'}}{P_j^{\Delta}} - 1 \right| \leq \frac{\alpha\Delta}{2\beta}$$

and the statement of the lemma, therefore, holds. $\square$

Theorem 6.10 (Global multiplicative bound on approximation) *Under the hypotheses of the lemma 6.9, let $N_d < \infty$ be the maximum number of autonomous transitions taken on any extended trace $\mathcal{T}$ of duration $< d$. Then for every strategy π ,*

$$\frac{P_{\Delta'}^d(\vec{tl} \mid \pi)}{P_{\Delta}^d(\vec{tl} \mid \pi)} \in \left[\left(1 - \frac{\alpha\Delta}{2\beta}\right)^{N_d}, \left(1 + \frac{\alpha\Delta}{2\beta}\right)^{N_d} \right]$$

and taking the supremum over strategies,

$$\frac{\sup_{\pi}(P_{\Delta'}^d)}{\sup_{\pi}(P_{\Delta}^d)} \in \left[\left(1 - \frac{\alpha\Delta}{2\beta}\right)^{N_d}, \left(1 + \frac{\alpha\Delta}{2\beta}\right)^{N_d} \right]$$

Proof. Let $\mathcal{T}_d^{\pi}$ be the set of all extended traces of duration $< d$ that reach ℓ_{target} under a realized strategy π . Along that trace, at most, N_d autonomous race points exist due to progressiveness and bounded time, and urgency rules out additional autonomous branching within finite time, thus, $\mathcal{T}_d^{\pi}$ is finite.

For each $\mathcal{T} \in \mathcal{T}_d^\pi$, the probability of $\mathcal{T}$ under the Δ -semantic can be written as a product of local factors along $\mathcal{T}$:

$$\Pr_{\Delta}(\mathcal{T}) = \prod_{i=1}^{m(\mathcal{T})} s_i^{\Delta} p_i^{\Delta}$$

where each $s_i^{\Delta} \in [0, 1]$ is the share of each branch at split points. This share can either be

- The probability of an autonomous transition taken at a particular split point (depending on the local rates and Δ granularity), or
- A forced/urgent transition being taken, which fires deterministically when enabled and whether it is taken does not depend on Δ -refinement while only the exact time it has fired is dependent on Δ , therefore, contributes 1 to the factor

This argument hinges on the fact that our particular type of Δ -approximate semantics does not discretize time, but only the time points of τ transitions, such that other transitions are not affected up to their interference with τ transitions. The latter remains controlled due to the non-punctuality of guards.

By definition, the total probability of reaching a target location ℓ_{target} under a realized strategy π is the sum over all traces:

$$P_{\Delta}^d(\vec{tl} \mid \pi) = \sum_{\mathcal{T} \in \mathcal{T}_d^\pi} \Pr_{\Delta}(\mathcal{T})$$

According to lemma 6.9, local multiplicative stability holds for $\Delta = 2\Delta'$ and any autonomous transition share s_i^{Δ} coming from a branching point along $\mathcal{T}$: $|s_j^{\Delta'} - s_j^{\Delta}| \leq \frac{\alpha\Delta}{2}$. Therefore,

$$\frac{P_i^{\Delta'}}{P_i^{\Delta}} \in \left[1 - \frac{\alpha\Delta}{2\beta}, 1 + \frac{\alpha\Delta}{2\beta}\right], \quad \text{with} \quad \alpha := \frac{lc}{\underline{r}} \quad \& \quad \beta := \min \frac{r_{jmin}}{\hat{r}_{max}} > 0$$

For a forced/urgent transition share, $s_i^{\Delta} = s_i^{\Delta'} = 1$, so this bound still trivially holds. Let trace $\mathcal{T}$ have $m(\mathcal{T})$ transition branching, of which at most N_d are autonomous transitions (the rest are forced transitions with share=1). Multiplying the ratio of autonomous transition probability shares along the trace yields

$$\frac{\Pr_{\Delta'}(\mathcal{T})}{\Pr_{\Delta}(\mathcal{T})} = \prod_{i=1}^{m(\mathcal{T})} \frac{P_i^{\Delta'}}{P_i^{\Delta}} \in \left[\left(1 - \frac{\alpha\Delta}{2\beta}\right)^{N_d}, \left(1 + \frac{\alpha\Delta}{2\beta}\right)^{N_d}\right]$$

because each trace has at most N_d autonomous transition split points and all shares are ≥ 0 . Equivalently,

$$(1 - \frac{\alpha\Delta}{2\beta})^{N_d} \Pr_{\Delta}(\mathcal{T}) \leq \Pr_{\Delta'}(\mathcal{T}) \leq (1 + \frac{\alpha\Delta}{2\beta})^{N_d} \Pr_{\Delta}(\mathcal{T})$$

A sum over all traces $\mathcal{T} \in \mathcal{T}_d^{\pi}$ and pulling the common factors $1 \pm \frac{\alpha\Delta}{2\beta}$ in front of sum since all terms are nonnegative, result in:

$$(1 - \frac{\alpha\Delta}{2\beta})^{N_d} \sum_{\mathcal{T}} \Pr_{\Delta}(\mathcal{T}) \leq \sum_{\mathcal{T}} \Pr_{\Delta'}(\mathcal{T}) \leq (1 + \frac{\alpha\Delta}{2\beta})^{N_d} \sum_{\mathcal{T}} \Pr_{\Delta}(\mathcal{T})$$

By definition, sums indicate the total reach probabilities, therefore, we obtain

$$(1 - \frac{\alpha\Delta}{2\beta})^{N_d} P_{\Delta}^d(\vec{t}l \mid \pi) \leq P_{\Delta'}^d(\vec{t}l \mid \pi) \leq (1 + \frac{\alpha\Delta}{2\beta})^{N_d} P_{\Delta}^d(\vec{t}l \mid \pi)$$

Since the multiplicative factors are nonnegative and independent of π , we can take supremum $\sup_{\pi}$ on both sides preserving the inequalities:

$$(1 - \frac{\alpha\Delta}{2\beta})^{N_d} \sup_{\pi} P_{\Delta}^d \leq \sup_{\pi} P_{\Delta'}^d \leq (1 + \frac{\alpha\Delta}{2\beta})^{N_d} \sup_{\pi} P_{\Delta}^d$$

If $\sup_{\pi} P_{\Delta}^d(\vec{t}l \mid \pi) > 0$, dividing by it yields the stated two-sided multiplicative bound for the optimal value. $\square$

Corollary 6.11 (Dyadic Cauchy Convergence) *For a realized strategy π , horizon $d > 0$ and target $\ell_{\vec{t}l}$, let $\Delta_m := \frac{\Delta}{2^m}$ for $m \geq 0$. Then the total reach probability $P_{\Delta_m}^d|_{m \geq 0}$ is Cauchy sequence $\mathbb{R}$ and converges to a limit $\lim_{m \rightarrow \infty} P_{\Delta_m}^d(\vec{t}l)$ underlying Def. 6.7. Moreover, for every $\varepsilon \in (0, 1)$ there exists m_{ε} such that for all integers $m' \geq m \geq m_{\varepsilon}$,*

$$|P_{\Delta_{m'}}^d(\vec{t}l \mid \pi) - P_{\Delta_m}^d(\vec{t}l \mid \pi)| \leq \varepsilon.$$

In particular, it suffices to choose any m large enough such that

$$\frac{\alpha\Delta_m}{2\beta} < 1 \quad \text{and} \quad e N_d \frac{\alpha}{\beta} \Delta_m \leq \varepsilon,$$

where $\alpha := \frac{lc}{r}$ and $\beta := \min_{\vec{r}_{max}} r_{jmin}^{\vec{r}_{jmin}} > 0$.

Proof. By applying Theorem 6.10 with $\Delta = \Delta_m$ and $\Delta' = \Delta_{m+1} = \frac{\Delta_m}{2}$, for all sufficiently large m we have $\frac{\alpha\Delta_m}{2\beta} < 1$, hence, we obtain the two-sided bound:

$$\left(1 - \frac{\alpha\Delta_m}{2\beta}\right)^{N_d} P_{\Delta_m}^d(\vec{t}l \mid \pi) \leq P_{\Delta_{m+1}}^d(\vec{t}l \mid \pi) \leq \left(1 + \frac{\alpha\Delta_m}{2\beta}\right)^{N_d} P_{\Delta_m}^d(\vec{t}l \mid \pi).$$

Since $P_{\Delta_m}^d(\vec{t}l \mid \pi) \in [0, 1]$, subtracting $P_{\Delta_m}^d(\vec{t}l \mid \pi)$ from the upper inequality and using $P_{\Delta_m}^d(\vec{t}l \mid \pi) \leq 1$ yields

$$P_{\Delta_{m+1}}^d(\vec{t}l \mid \pi) - P_{\Delta_m}^d(\vec{t}l \mid \pi) \leq \left(\left(1 + \frac{\alpha\Delta_m}{2\beta}\right)^{N_d} - 1 \right) P_{\Delta_m}^d(\vec{t}l \mid \pi) \leq \left(1 + \frac{\alpha\Delta_m}{2\beta}\right)^{N_d} - 1.$$

Similarly, subtracting $P_{\Delta_{m+1}}^d(\vec{t}l \mid \pi)$ from $P_{\Delta_m}^d(\vec{t}l \mid \pi)$ and using $P_{\Delta_m}^d(\vec{t}l \mid \pi) \leq 1$ results in

$$P_{\Delta_m}^d(\vec{t}l \mid \pi) - P_{\Delta_{m+1}}^d(\vec{t}l \mid \pi) \leq \left(1 - \left(1 - \frac{\alpha\Delta_m}{2\beta}\right)^{N_d}\right) P_{\Delta_m}^d(\vec{t}l \mid \pi) \leq 1 - \left(1 - \frac{\alpha\Delta_m}{2\beta}\right)^{N_d}.$$

Combining both gives the absolute adjacent bound

$$|P_{\Delta_{m+1}}^d(\vec{t}l \mid \pi) - P_{\Delta_m}^d(\vec{t}l \mid \pi)| \leq \max\left\{\left(1 + \frac{\alpha\Delta_m}{2\beta}\right)^{N_d} - 1, 1 - \left(1 - \frac{\alpha\Delta_m}{2\beta}\right)^{N_d}\right\}.$$

For $\frac{\alpha\Delta_m}{2\beta} \in (0, 1)$ we have

$$1 - \left(1 - \frac{\alpha\Delta_m}{2\beta}\right)^{N_d} \leq \left(1 + \frac{\alpha\Delta_m}{2\beta}\right)^{N_d} - 1,$$

Therefore,

$$|P_{\Delta_{m+1}}^d(\vec{t}l \mid \pi) - P_{\Delta_m}^d(\vec{t}l \mid \pi)| \leq \left(1 + \frac{\alpha\Delta_m}{2\beta}\right)^{N_d} - 1.$$

Using $(1 + x)^{N_d} \leq \exp(N_d x)$ for $x \geq 0$ and $\exp(x) - 1 \leq x \exp(x)$ for $x \geq 0$, with $x = N_d \frac{\alpha\Delta_m}{2\beta}$, we obtain

$$\left(1 + \frac{\alpha\Delta_m}{2\beta}\right)^{N_d} - 1 \leq \exp\left(N_d \frac{\alpha\Delta_m}{2\beta}\right) - 1 \leq N_d \frac{\alpha\Delta_m}{2\beta} \exp\left(N_d \frac{\alpha\Delta_m}{2\beta}\right).$$

For all sufficiently large m we also have $N_d \frac{\alpha\Delta_m}{2\beta} \leq 1$, hence $\exp\left(N_d \frac{\alpha\Delta_m}{2\beta}\right) \leq e$ and thus

$$|P_{\Delta_{m+1}}^d(\vec{t}l \mid \pi) - P_{\Delta_m}^d(\vec{t}l \mid \pi)| \leq e N_d \frac{\alpha\Delta_m}{2\beta}.$$

Now we fix $m' \geq m$ in the sequence where the previous bound holds. By the triangle inequality,

$$|P_{\Delta_{m'}}^d(\vec{t}l \mid \pi) - P_{\Delta_m}^d(\vec{t}l \mid \pi)| \leq \sum_{k=m}^{m'-1} |P_{\Delta_{k+1}}^d(\vec{t}l \mid \pi) - P_{\Delta_k}^d(\vec{t}l \mid \pi)| \leq \sum_{k=m}^{m'-1} e N_d \frac{\alpha\Delta_k}{2\beta}.$$

Since $\Delta_k = \frac{\Delta}{2^k}$, we have $\sum_{k=m}^{\infty} \Delta_k = \sum_{k=m}^{\infty} 2^{-k} \Delta = \Delta_m \sum_{k=0}^{\infty} 2^{-k} = 2\Delta_m$, so

$$|P_{\Delta_{m'}}^d(\vec{t}l \mid \pi) - P_{\Delta_m}^d(\vec{t}l \mid \pi)| \leq e N_d \frac{\alpha}{2\beta} \sum_{k=m}^{\infty} \Delta_k = e N_d \frac{\alpha}{\beta} \Delta_m \leq \varepsilon.$$

This proves the Cauchy property and yields the stated sufficient condition for $|P_{\Delta_m}^d(\vec{t}l \mid \pi) - P_{\Delta_m}^d(\vec{t}l \mid \pi)| \leq \varepsilon$ and therefore converges in $\mathbb{R}$. We denote its limit by

$$P_{Dyadic}^d(\vec{t}l \mid \pi) := \lim_{m \rightarrow \infty} P_{\Delta_m}^d(\vec{t}l \mid \pi)$$

□

From dyadic convergence to arbitrary granularities. At this point we have established convergence along the nested dyadic refinement $\Delta_m = \Delta/2^m$. Extending this to $\lim_{\Delta \rightarrow 0} P_{\Delta}^d(\vec{t}l \mid \pi)$ for arbitrary sequence of $\Delta \rightarrow 0$ requires an additional cross-granularity stability argument with proof sketch below, which we develop in the future.

Proof sketch. For a deterministic, c -progressive IMCC I , a horizon $d > 0$, a target $\vec{t}l \in L$, and a realized strategy π , the dyadic analysis established earlier yields stability under refinement $\Delta \rightarrow \Delta/2$ and implies that the dyadic sequence $\Delta_m = \Delta/2^m$ induces a Cauchy sequence $P_{\Delta_m}^d(\vec{t}l \mid \pi)_{m \geq 0}$, with its limit denoted by $P_{Dyadic}^d(\vec{t}l \mid \pi)$. To extend this convergence to arbitrary granularities, we introduce lower and upper envelope Δ -semantics. For a fixed $\Delta > 0$, we partition $[0, d)$ into slices $\Theta_k^\Delta = [k\Delta, (k+1)\Delta)$, and for each slice start (ℓ, v) and each enabled autonomous branch j , we define its rate envelopes

$$r_{j,k}^{-\Delta}(\ell, v) = \inf_{t \in \Theta_k^\Delta} r_j(\ell, v + t), \quad r_{j,k}^{+\Delta}(\ell, v) = \sup_{t \in \Theta_k^\Delta} r_j(\ell, v + t),$$

where $\hat{r}_k^{\pm\Delta} = \sum_j r_{j,k}^{\pm\Delta}$. These envelopes induce conservative one-slice probability race kernels via

$$p_k^{-\Delta}(\text{no-jump} \mid \ell, v) = \exp(-\Delta \hat{r}_k^{+\Delta}), \quad p_k^{+\Delta}(\text{no-jump} \mid \ell, v) = \exp(-\Delta \hat{r}_k^{-\Delta}),$$

with corresponding branch shares $s_{j,k}^{-\Delta} = \frac{r_{j,k}^{-\Delta}}{\hat{r}_k^{+\Delta}}$ and $s_{j,k}^{+\Delta} = \frac{r_{j,k}^{+\Delta}}{\hat{r}_k^{-\Delta}}$. Replacing the original race resolution on each slice by these kernels yields lower and upper envelope reach probabilities $P_{\Delta}^{d-}(\vec{t}l \mid \pi)$ and $P_{\Delta}^{d+}(\vec{t}l \mid \pi)$. Assuming that the original Δ -semantics computes its one-slice probabilities from representative values lying between the slice infimum and supremum, we obtain the uniform sandwich

$$P_{\Delta}^{d-}(\vec{t}l \mid \pi) \leq P_{\Delta}^d(\vec{t}l \mid \pi) \leq P_{\Delta}^{d+}(\vec{t}l \mid \pi) \quad \text{for all } \Delta > 0.$$

Envelope semantics are refinement-monotone: under subdivision of slices, lower envelopes become less pessimistic and upper envelopes become less optimistic. Hence, along any nested refinement chain, in particular the dyadic one, P_{Δ}^{d-} is monotone non-decreasing and P_{Δ}^{d+} is monotone non-increasing, both bounded in $[0, 1]$. By Lipschitz continuity of the rates, the inner-slice variation is $O(\Delta)$, which

implies that the envelope gap satisfies $0 \leq P_{\Delta}^{d+}(\vec{t}l \mid \pi) - P_{\Delta}^{d-}(\vec{t}l \mid \pi) \leq C \Delta$ for a constant C depending only on the model parameters and d . Thus, the two envelope limits coincide as $\Delta \rightarrow 0$. Finally, for an arbitrary sequence $\Delta_n \rightarrow 0$, the sandwich inequality together with the vanishing envelope gap yields $\lim_{n \rightarrow \infty} P_{\Delta_n}^d(\vec{t}l \mid \pi) = P^d(\vec{t}l \mid \pi)$, showing that the continuous-time reach probability is well defined and independent of the chosen refinement schedule.

6.3 Expressiveness and Scope

The integration of stochastic dynamics with temporal constraints is central to modeling and optimizing complex real-world systems. From autonomous rescue operations to resource allocation and decision-making under uncertainty, it is often inevitable to face the challenge of probabilistic reasoning with timing constraints. Traditional frameworks like Interactive Markov Chains (IMCs) [Her02; HK09] and Timed Automata (TAs) [AD94] have been instrumental in addressing these aspects individually. However, many scenarios demand a unification of these approaches to model systems where timing and stochasticity are entangled.

Interactive Markov Chains with Clocks (IMCCs, [FKK24]) address this need by extending the versatility of existing models by incorporating rate-based Markovian transitions with rates that can change continuously as time and in consequence the clocks evolve. IMCCs achieve this by means of rate expressions formed over the clocks such that the rates of stochastic transitions are a function of the continuously evolving clocks. Pairing these with urgent transitions enabled by clock constraints (and featuring induced clock resets), IMCCs permit a more detailed capture of time-varying probabilities than related models.

The most relevant applications of IMCCs appear in the domain of time-critical decision-making, such as optimizing rescue missions in catastrophic scenarios, where the survival probability of victims diminishes over time, governed by dynamically evolving fatality rates. Robots deployed in such missions must not only decide the order of rescues but also adapt their strategies dynamically based on the progression of time and the probabilistic outcomes of previous decisions. By reflecting both temporal logic (through guards and clock resets) and stochastic transitions (through clock-dependent rates), IMCCs make it feasible to model and analyze such highly complex and dynamic scenarios with greater accuracy.

6.4 Further Steps to Take

While IMCCs provide a relevant expressiveness that significantly extends or complements that of many related model classes (cf. Fig. 5.1), their computational properties in terms of feasibility of effective strategy synthesis and decidability of analytical questions concerning their dynamics hitherto remained active topics of investigation.

A case in point is to decide the existence of strategies that guarantee certain reach probabilities or other expected values (e.g., the expected number of patients saved) or to decide whether a given strategy is optimal in the sense that it maximizes (or minimizes) such expected values. A related, but simpler problem is to decide whether the suggested strategy is at least ε -optimal, i.e. generates expectations that are at most ε smaller (or larger, resp.) than the optimum, for a given $\varepsilon > 0$.

Understanding whether such problems can be decided or respective strategies computed, and if so under which conditions, is crucial for advancing practical solutions as well as for shedding light on the modeling power of IMCCs. Following chapters aim to fill this gap by presenting (un)decidability results for optimal reachability in IMCCs along with the implications that stem from them.

Decidability Frontiers — What We Can('t) Verify Algorithmically

7.1 Decidability in Timed/Probabilistic Models: A Survey

A fundamental concept in theoretical computer science is *decidability* that concerns the existence of algorithms capable of resolving a given problem within finite time [Tur37]. Our attempt to establish and precisely delineate decidability of reachability optimality in IMCCs arises from the essential question driving the broader research on real-time systems: in many application domains, one needs to decide whether optimality of a strategy exists or whether a crucial location can be reached with a probability exceeding some threshold.

In order to analyze the decidability boundaries of diverse models of computation, a variety of methods have been developed, many of which rely on constructing finite abstractions as in region and zone graphs to prove decidability. For certain subclasses, logical methods exist to reduce problems to well-known decidable theories, namely Presburger arithmetic [BC96], real algebraic geometry and o-minimal theories, which enable reasoning about systems with continuous dynamics [Cas+08], particularly in hybrid and parametric systems. Undecidability results, on the other hand, are often obtained via reductions of the undecidable problems of Diophantine equations [Ani85], halting problems of counter machines [Min66], or Skolem-related problems [XDI03; She+05; ABK22].

Depending on the underlying mathematical or formal model and the properties of interest, decidability can vary. In Timed Automata, for instance, reachability of a certain location (potentially also within specific time constraints) is a problem of interest that has been proven to be decidable by a finite-state time-abstract bisimulation [AD94]. In Priced Timed Automata [Beh+01], cost-optimal reachability is computable for a single cost variable as well as when a second cost constraint is added, but in general undecidable for Pareto objectives in Multi-Priced Timed Automata [Frä+22; SSW24]. Each such extension of Timed Automata with features such as costs [BLR04], parameters [ALR22], or stochasticity and probabilistic choices [NPS13] does, however, track the borderline of computability. For example, reachability in more expressive extension models such as Probabilistic Timed Automata

with Clock-dependent Probabilities (cd-PTA) [Spr17] is undecidable i.g., though subclasses like the one-clock case remain solvable [Spr24]. Related undecidability results have also been shown in models such as Probabilistic Finite Automata with exponential ambiguity, where even qualitative reachability becomes Skolem-hard [BS23]. As shown in studies on decidability of optimal reachability in Weighted Timed Games –a generalization of Priced Timed Automata with game-theoretic interaction–, the value problem is undecidable in general and not even approximable in the unbounded-horizon setting [GO24]. However, certain structural assumptions such as divergence (disallowing near-zero values of cycle weights) [BMR17], almost-divergence [BMR21], or the one-clock setting [MPR25] regain decidability.

Combining concurrency with continuous-time stochastic behavior, Interactive Markov Chains (IMCs) unify nondeterminism and probabilistic choices with Markovian transition timing. The decidability results however, depend crucially on whether the model is closed or interacts with an unresolved nondeterministic environment. For closed IMCs [HKK13], several quantitative problems are tractable, e.g. time-bounded reachability can be approximated by discretization-based procedures with explicit error bounds [ZN10; HH15b], while objectives such as extremal expected time to reach target states and long-run average time spent in a set admit reductions to stochastic shortest-path problems and standard numerical solution methods [Guc+12]. When arbitrary unknown environments come into play, being modeled by non-determinism, the problem becomes undecidable in general through a reduction to stochastic partial-observation games, though approximation schemes for the bounded horizon version have been devised [HKK13; Brá+12].

With the development of Markov Automata [EHZ10] as a generalization of IMCs by incorporating probabilistic branching, these quantitative objectives carry over to a richer model class. In particular, expected time objectives were again reduced to Stochastic Shortest Path problems, long-run average objectives to long-run ratio MDPs (Markov Decision Processes [Bel57a; Bel57b; Put05]), and timed reachability remained approximation-based through discretization [Guc+14]. However, these decidability and approximation outcomes are again obtained under technical assumptions such as non-Zenoness and finite-state behavior.

It should be noted that our setting does not currently allow arbitrary non-deterministic environments. By imposing maximal-progress priorities, our semantics eliminates competition between interactive, delay, and autonomous transitions, so that urgency-induced behavior is no longer nondeterministic. Residual nondeterminism may still remain among simultaneously enabled interactive alternatives to be resolved by a scheduler choice, rather than environmental adversarial behavior.

By incorporating clock-dependent stochastic rates into the framework of IMCs and clock-dependent transition guards as in TA, we have enhanced the ability to model real-world in the domains of Safety-critical and Cyber-physical systems where time-varying failure rate as well as decaying and aging performance is crucial to be considered. As the level of expressiveness has grown significantly, one is left with wonder of whether our IMCC model class inherits the quantitative properties from its two ancestors. This raises deeper questions of which classes of performance measures remain undecidable and what consideration has to be made toward computability of IMCCs. It is, thus, crucial to take the step toward bridging the gap between qualitative correctness and quantitative performance assurance in modeling timed and probabilistic systems.

7.2 On the Computability of Optimal Strategies

Models combining probabilistic or stochastic behavior and timed-automaton-style clock-controlled behavior have long been studied, among them Probabilistic Timed Automata [Spr01; Spr17] and Stochastic Hybrid Systems [HLS00; Hes04; Car+12]. The Interactive Markov Chain with Clocks (IMCC, [FKK24]) studied here are a particularly expressive variation of this theme, as they feature clock-dependent and thus continuously varying rates of autonomous transitions. Given their expressiveness, their computability properties are of interest.

As the IMCC model was motivated by complex scheduling problems, especially those in rescue missions where robots must prioritize victims under time-varying and uncertain circumstances, questions of computing optimal strategies or deciding strategy optimality are of primary interest. Optimality herein means maximizing the expectation of a random variable that represents a reward, like maximizing the expected number of patients rescued or, equivalently, minimizing expected casualties. The weaker related notion of ε -optimality in turn signifies strategies that realize an expected reward that is at most ε worse than the optimum for a given $\varepsilon > 0$.

The central questions of computability induced by this setting are the following:

1. Can an optimal strategy be computed effectively if such a strategy exists¹?
2. Can the supremum of the expected rewards achieved by strategies be computed?
3. Given a strategy π , is optimality of π decidable?

¹Note that in general, there need not be an optimal strategy, as the set of expected reward values realized by different strategies may be open. An example would be staying in a state where waiting incurs positive reward, but that reward is lost or decreased when a clock-guarded transition takes place at some future time t , which can be avoided by preempting the waiting strictly before t via a controllable action a . To maximize reward, a should be played close to yet strictly before t .

4. Given $\varepsilon > 0$, can an ε -optimal strategy be computed effectively?
5. Given $\varepsilon > 0$, can the supremum of the expected rewards achieved by strategies be computed approximately up to an accuracy of ε , i.e. up to a tolerance of at most $\pm \frac{\varepsilon}{2}$?
6. Given a strategy π and $\varepsilon > 0$, is ε -optimality of π decidable?
7. Given a target performance θ , can we compute a strategy that provides an expected reward of at least θ if such a strategy exists?
8. Given a target performance θ , can we decide whether a strategy that provides an expected reward of at least θ exists?

We will answer them in both the setting of unbounded time, where the time horizon is infinite and a strategy may run for an unbounded duration, and of bounded time, where a fixed maximum duration delimiting the temporal horizon is given which applies to all runs of interest. Note that the latter bound on the duration of the runs of the IMCC also implies a uniform upper bound on all clocks, as all clocks start at 0 such that they cannot acquire larger values than the duration of the run.

7.3 Solution Existence Criteria

7.3.1 Strategies in IMCCs

The focus of this section is optimal control of IMCCs. We therefore partition the alphabet Σ into a set $\Sigma_C \subseteq \Sigma$ of controllable actions, which a strategy or policy can suppress or activate whenever they are enabled in the IMCC, and its complement $\Sigma \setminus \Sigma_C$ of uncontrollable actions that occur urgently whenever enabled in the IMCC. Formally, a memoryless strategy π then is defined as a function $\pi : L \times \text{ClockVal} \rightarrow (\Sigma_C \cup \{\perp\})$, where the extra symbol $\perp$ denotes the option of not adopting any controllable action at the time. For technical reasons, we will insist on strategies also being non-punctual in that for every action $a \in \Sigma_C$, the topological closure of the interior of $\pi^{-1}(a)$ equals $\pi^{-1}(a)$.

A strategy acts on the IMCC by further restricting the guard conditions of controllable actions. In case that an uncontrollable action and a controllable one are concurrently enabled, we posit that the uncontrollable one takes priority. Likewise, we assume a fixed priority between concurrently enabled uncontrollable actions. Then the IMCC I can under strategy π engage into the following Δ transitions:

- $(l, v, t) \xrightarrow{a, \Delta}_\pi (l', v', t)$ iff $a \in \Sigma \setminus \Sigma_C$ and $(l, v, t) \xrightarrow{a, \Delta}_A (l', v', t)$ and there is no (l'', v'', t) and no $a' \in \Sigma \setminus \Sigma_C$ of higher priority such that $(l, v, t) \xrightarrow{a', \Delta}_A (l'', v'', t)$, i.e. uncontrollable actions are inherited;

- $(l, v, t) \xrightarrow{a, \Delta}_\pi (l', v', t)$ iff $a \in \Sigma_C$ and $\pi(l, v) = a \wedge (l, v, t) \xrightarrow{a, \Delta}_A (l', v', t) \wedge \exists k \in \mathbb{N} : t = k\Delta$ and there is no (l'', v'', t) and no $a' \in \Sigma \setminus \Sigma_C$ such that $(l, v, t) \xrightarrow{a', \Delta}_A (l'', v'', t)$, i.e. controllable actions have to be enabled by both the IMCC and the strategy and they occur at $k\Delta$ time instants;
- $(l, v, t) \xrightarrow{\delta, \Delta}^p_\pi (l, v + \delta, t + \delta)$ iff $(l, v, t) \xrightarrow{\delta, \Delta}^p_A (l, v + \delta, t + \delta)$, i.e. time passage behavior is inherited;
- $(l, v, t) \xrightarrow{\tau, \Delta}^p_\pi (l', v, t)$ iff $(l, v, t) \xrightarrow{\tau, \Delta}^p_A (l', v, t)$, i.e. autonomous transitions are inherited from the underlying IMCC.

Again, the full continuous-time behavior is then recovered as the limit of the Δ -dynamics for $\Delta \rightarrow 0$. As previously shown that this limit is well-defined due to the requirement of non-punctual guard conditions, which eventually (for small enough Δ) stabilizes guard conditions and thus enabledness of runs against changes in the discretization Δ .

7.3.2 (Un)decidability of Strategy Optimality Problems in IMCCs

We first address the case of temporally unbounded runs, for which we show that all the above problems are not computable even if we restrict ourselves to the simplest random variable, namely reaching a designated target location in the IMCC. Encoding a reach of the target location as value 1 and else granting the reward 0, the corresponding expectation is nothing else than the reach probability of the target location, which already turns out to give rise to non-computable instances of the above six problems.

We start our investigation with the last of the eight problems above, showing that it is undecidable i.g. whether a strategy exists that realizes a probability of reaching the target state that exceeds a desired target probability θ . We therefore present a reduction from the Diophantine equation problem ‘Does a given polynomial $P(\mathbf{x}) \in \mathbb{Z}[x_1, \dots, x_n]$ have an integer root?’ to a location reachability problem in an IMCC.

Lemma 7.1 (Undecidability of reach probabilities exceeding a threshold) *Given an IMCC I , a location l in the IMCC I , and a target probability of $\theta \in (0, 1]$, it is undecidable i.g. whether a strategy over I exists that reaches l with probability $p \geq \theta$.*

Proof. We employ a reduction from the undecidable problem of determining the existence of integer solutions to arbitrary Diophantine equations, i.e build upon Matiyasevich’s resolution [Mat96] of Hilbert’s 10th problem [Hil00]. Therefore let $P(x_1, \dots, x_n)$ be an arbitrary multivariate polynomial with integer coefficients.

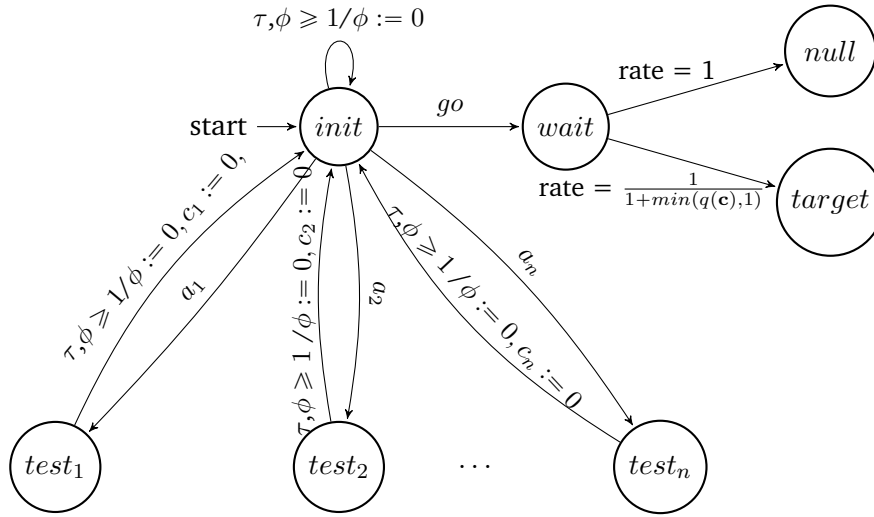


Fig. 7.1: An IMCC encoding a Diophantine equation problem

Our objective is to construct an IMCC such that the existence of a timed sequence leading to the target state with long-run probability $\theta = \frac{1}{2}$ precisely corresponds to the existence of integer solutions to the equation $P(\mathbf{x}) = 0$.

Initially, we transform the given polynomial P into a non-negative polynomial P_{nn} by squaring it, yielding $P_{nn}(x_1, \dots, x_n) = (P(x_1, \dots, x_n))^2$. This transformation preserves the set of integer roots of the original equation while ensuring that P_{nn} is non-negative for all arguments. Subsequently, we substitute each variable x_i in P_{nn} with the expression $c_0 - c_i$, resulting in a new polynomial $q(c_0, c_1, \dots, c_n) = P_{nn}(c_0 - c_1, \dots, c_0 - c_n)$. The polynomial q retains the property of having an integer root if and only if there exists an integer root of $P(\mathbf{x}) = 0$.

Due to the specific structure of q , which incorporates only differences of variables $c_0 - c_i$ which can be realized by clocks c_0 and c_i as these keep their difference even when time evolves, the reduction of the Diophantine equation problem $q(c_0, c_1, \dots, c_n) = 0$ into an IMCC optimal reachability probability problem can be achieved as follows: We construct an IMCC I_q with

- a finite set of pairwise distinct locations $L = \{init\} \cup \{test_1, \dots, test_n\} \cup \{wait\} \cup \{target, null\}$ and
- a finite set of pairwise distinct clocks $C = \{c_0, c_1, \dots, c_n\} \cup \{\phi\}$

and transitions as shown in Fig.7.1. The target location we are interested on is *target*.

It should be noted that c_0 is never reset and for each $i = 1, \dots, n$, the clock constraints on the transitions from locations $test_i$ back to *init* enforce by means of the auxiliary

clock ϕ that $(c_0 - c_i)$ is set to an integer x_i ; which integer it is, is decided by the strategy of activating actions a_1 to a_n . In order to reach the target location *target*, the strategy has to eventually select the action *go*, which leads to location *wait* with its two outgoing rate transitions.

By construction, these two rate transitions have the properties that

1. their rates stay both constant when time evolves, even though the rate of the transition to *target* indeed is clock dependent;
2. their rates are equal, namely both 1, if and only if $q(c_0, c_1, \dots, c_n) = P_{nn}(c_0 - c_1, \dots, c_0 - c_n) = 0$;
3. if $q(c_0, c_1, \dots, c_n) = P_{nn}(c_0 - c_1, \dots, c_0 - c_n) \neq 0$, the rate of the autonomous transition to *target* is $\frac{1}{2}$, i.e. is half of the rate for transitioning to *null*.

As the probabilities for reaching locations due to autonomous transitions split according to the relation of the respective rates, the IMCC from Fig. 7.1 reaches *target* with probability $\frac{1}{2}$ iff $q(c_0, c_1, \dots, c_n) = P_{nn}(c_0 - c_1, \dots, c_0 - c_n) = 0$ and with probability $\frac{1}{3}$ otherwise.

Consequently, there is a strategy obtaining a reach probability for *target* of $\theta = \frac{1}{2}$ iff the Diophantine equation $P_{nn}(x_1, \dots, x_n) = 0$ has a solution, which is equivalent to the original Diophantine equation $P(x_1, \dots, x_n) = 0$ having a solution. By the undecidability of Hilbert's tenth problem, this implies that existence of a strategy obtaining a reach probability of at least $\theta = \frac{1}{2}$ is undecidable. $\square$

The above proof in fact also reveals that the optimum reach probability cannot even be approximated in an effective manner, thus negatively answering question 5 too:

Lemma 7.2 (Non-approximability of optimal reach probabilities) *Given an IMCC I and a location l in the IMCC I , the probability of reaching l under an optimal strategy i.g. can neither be computed exactly nor be approximated to any precision $\pm\varepsilon$ with $\varepsilon < \frac{1}{2}$.*

Proof. The reduction of Diophantine equations to IMCC reach probabilities from the previous proof yields an IMCC with an optimum reach probability of $\frac{1}{2}$ iff the respective Diophantine equation has a solution and an optimum reach probability of $\frac{1}{3}$ iff the respective Diophantine equation has no solution. As the gap between $\frac{1}{2}$ and $\frac{1}{3}$ is $\frac{1}{6}$, the existence of any algorithm that computes reach probabilities with an accuracy better than $\pm\frac{1}{12}$ would contradict the undecidability of the Diophantine equation problem.

The constant $\frac{1}{6}$ can be generalized to any constant less than 1 by replacing in Fig. 7.1 the rate of $\frac{1}{1+\min(q(c),1)}$ for the transition to location *target* by the rate $\frac{1}{r+s \cdot \min(q(c),1)}$ for two constants $r, s > 0$. The width of the above gap then becomes $\frac{1}{r} - \frac{1}{r+s}$ and thus widens if r becomes small and s large. It converges from below to 1 for $r \rightarrow 0$ and $s \rightarrow \infty$. Hence, non-approximability even applies for any precision strictly better than $\pm \frac{1}{2}$. $\square$

Corollary 7.3 *None of the following problems can be solved effectively in general given an IMCC I :*

1. *Compute an optimal strategy over I if such a strategy exists.*
2. *Compute the supremum of the expected rewards achieved by strategies over I .*
3. *Given a strategy π , decide optimality of π over I .*
4. *Given $\varepsilon > 0$, compute an ε -optimal strategy over I .*
5. *Given $\varepsilon > 0$, compute the supremum of the expected rewards achieved by strategies over I up to an accuracy of ε , i.e. up to a tolerance of $\pm \frac{\varepsilon}{2}$.*
6. *Given a strategy π and $\varepsilon > 0$, decide ε -optimality of π over I .*
7. *Given a target performance θ , compute a strategy over I that provides an expected reward of at least θ if such a strategy exists.*
8. *Given a target performance θ , decide whether a strategy over I exists that provides an expected reward of at least θ .*

Proof. Lemmata 7.1 and 7.2 showed that problem 8 as well as problems 2 and 5 are not computable. For all other problems, there exist reductions of these three problems to them. For example, computing optimal (or ε -optimal) strategies entails deciding optimality (or ε -optimality, resp.) of a strategy, as one can build an extended IMCC that first offers the choice between the original IMCC and its specialization to the strategy of interest. Hence, one cannot compute (ε -)optimal strategies without being able to decide (ε -)optimality of a given strategy. $\square$

The above corollary states that all time-unbounded properties of interest are not computable over IMCCs, even if reduced to their approximate form. It should, however, be noted that the above encoding of Diophantine equations requires an infinite time horizon such that computability of some of the problems could be recovered in a finite-horizon setting.

7.4 Implications

The optimal reachability problem in IMCCs is undecidable in its general form, a fact established by reductions to the solvability of Diophantine equations. This undecidability result highlights the expressive power of the model: once unrestricted clocks, concurrency, and arbitrary rate functions are allowed, no general decision procedure exists. Nevertheless, this undecidability does not close the door entirely. By restricting the model –whether through bounding clock domains, limiting global time to a finite horizon, or constraining the shape of rate functions– decidable subclasses can be identified. Not only does reachability regain decidability in these cases, but approximation schemes can also be systematically applied to obtain near-optimal strategies.

In other words, while unrestricted IMCCs resist algorithmic treatment, imposing natural temporal and structural constraints allows us to recover both theoretical guarantees and practical methods for strategy synthesis. When the structure of a real-world system fits within such subclasses, verification and scheduling tasks can be approached with existing algorithmic methods, providing practical leverage despite the model’s theoretical hardness.

Time-Bounded Problems: Exact vs. Approximate Quantitative Reachability Analysis

8.1 Computability over Bounded Horizons

Having shown that the hope for rigorous algorithmic solutions optimizing or analyzing time-unbounded strategies in IMCCs remains elusive, we now address the finite-horizon case, where a temporal horizon $d > 0$ of interest is given and the problems are to synthesize optimal or ε -optimal strategies over this time bound d or to algorithmically check optimality or ε -optimality of such a temporally bounded strategy.

In the bounded-horizon setting, the properties of interest are the eight properties that were also of interest for the unbounded horizon case, just relativized to a finite horizon, plus an new one:

1. Can an optimal strategy of duration d be computed effectively if such a strategy exists?
2. Can the supremum of the expected rewards achieved by strategies of duration d be computed?
3. Given a strategy π_d of duration d , is optimality of π_d decidable?
4. Given $\varepsilon > 0$, can an ε -optimal strategy of duration d be computed effectively?
5. Given $\varepsilon > 0$, can the supremum of the expected rewards achieved by strategies of duration d be computed approximately up to an accuracy of ε , i.e. up to a tolerance of at most $\pm \frac{\varepsilon}{2}$?
6. Given a strategy π_d of duration d and $\varepsilon > 0$, is ε -optimality of π_d decidable?
7. Given a target performance θ , can we compute a strategy of duration d that provides an expected reward of at least θ if such a strategy exists?
8. Given a target performance θ , can we decide whether a strategy of duration d that provides an expected reward of at least θ exists?
9. Given $\varepsilon > 0$, can we decide whether the supremum reward of all strategies that are time-bounded by duration d comes ε -close to the supremum reward of all unbounded strategies?

Of the above problems, we first address problem 4 and therefore leverage an appropriate approximation scheme from 6.2.3 and the precursor article [FKK24], which applies to IMCCs that are time-progressive. These approximability results from also immediately imply that the new problem 9. has no effective solution.

Our ability to approximately evaluate a given strategy over a given horizon now enables us to construct ε -optimal time-bounded strategies if we are able to enumerate these, which is possible in the δ -discrete setting and thus also in the continuous with the appropriate approximation error. As there are finitely many Δ -runs of duration d , an optimal strategy for Δ -runs can be computed by finite path enumeration. Hence, we obtain:

Corollary 8.1 *Given a progressive IMCC I , a target location $\vec{t}l$ in I , and $\varepsilon > 0$, a strategy π over I obtaining a reach probability $P^d(\vec{t}l)$ for reaching $\vec{t}l$ within d time that is at most ε worse than the supremum probability for reaching $\vec{t}l$ within d time can be computed effectively.*

This corollary answers question 4 affirmatively for the case of progressive IMCCs. Furthermore, corollaries 6.11, 8.1 and 8.2 together then immediately imply a positive answer to question 5, as they permit to compute an $\frac{\varepsilon}{2}$ -optimal strategy and to compute its performance up to an accuracy of again $\frac{\varepsilon}{2}$:

Corollary 8.2 *Given a progressive IMCC I , a target location $\vec{t}l$ in I , and $\varepsilon > 0$, the supremum of the d -bounded reach probabilities for $\vec{t}l$ of all strategies can be approximated effectively up to ε .*

Beyond these affirmative answers to problems 4 and 5, we can negatively answer quest 9:

Lemma 8.3 *Given an IMCC I , a target location $\vec{t}l$, a time horizon $d < \infty$, and $\varepsilon < \frac{1}{2}$, it is undecidable i.g. whether an optimal strategy π_d over the finite horizon d achieves a reach probability for l over horizon d that is ε -close to the long-run reach probability for l achieved by an optimal infinite-horizon strategy π over the infinite horizon.*

Proof. We have shown in [FKK24] and here that the optimal reach probabilities over horizon d can be approximated to an accuracy of $\delta < \frac{1}{2} - \varepsilon$. If we could decide whether this value in turn is ε -close to the optimal unbounded reach probability, we could compute the latter up to an accuracy of $\delta + \varepsilon < \frac{1}{2}$, contradicting Lemma 7.2.

□

It should be noted that the gadget from Fig. 7.1 actually is progressive such that the above Lemma even holds when restricting attention to progressive IMCCs.

We thus obtained positive results for problems 4 and 5 in the case of progressive IMCCs, and a negative one for problem 9. The status of all the horizon-bounded exact problems, i.e. problems 1–3, 7, and 8, remains undetermined at the time being, though we expect them to be undecidable. The same applies to problem 6, which has an intermediate status between exact and approximation problems.

8.1.1 With Time Bounds in Place: The Strategy Computation

IMCCs provide a general modelling framework for stochastic, time-dependent systems, with rescue missions serving as a representative application. While the previous section established approximability of bounded-horizon ϵ -optimality in principle, via reasoning over the induced path space, such an explicit enumeration of paths is not computationally viable. We therefore seek a more practical and efficient algorithmic approach preserving the same quantitative guarantees.

Let I be an IMCC, defined as a finite set of clocks $C = \{c_1, \dots, c_k\}$ each with ideal unit slope, location space L with autonomous and interactive transitions, continuous-time evolution of clocks and race conditions among autonomous transitions. We call I clock-bounded by M if each clock c_i is forced (by guards or resets) to remain in the interval $[0, M]$ for all reachable states of I . Equivalently, no clock can ever exceed M in any run of the IMCC. It should be noted that bounding the temporal horizon to d implies bounding the clocks to at most d as well, since all clocks start at value 0.

System dynamics. Let l be a location in the IMCC I with n outgoing autonomous transitions, and let $r_i : [0, M]^k \rightarrow \mathbb{R}_{\geq 0}$ be the respective rate functions of these autonomous transitions. Therefore, for any clock valuation v , the total outflow rate becomes $\hat{r}(v) = \sum_{i=1}^n r_i(v)$, and clock evolution is defined by time elapse, represented symbolically as $v + s$. It should be noted that $\hat{r}(v) \geq \alpha > 0$ so that an autonomous transition eventually fires if no forced guard is triggered.

Let $v : C \rightarrow \mathbb{R}_{\geq 0}$ be a clock valuation and $\delta > 0$ be the length of the slice $[t_k, t_{k+1}] = [k\delta, (k+1)\delta)$. The probability of survival $\mathcal{S}(l, v, \delta)$, meaning the probability of staying in l for more than δ time without an autonomous jump up to δ is a solution $p(\delta)$ to the initial value problem:

$$\begin{aligned} \frac{dp(s)}{ds} &= p(s) \cdot (-\hat{r}(v + s)), \\ p(0) &= 1, \end{aligned}$$

Therefore,

$$\mathcal{S}(\ell, v, \delta) = p_{\text{surv}}(\delta) = \exp \left(- \int_0^\delta \hat{r}(v + t) dt \right)$$

is the unique solution describing the probability for still residing in ℓ after δ time units, starting from state (ℓ, cv) . Once a particular autonomous event happens at the exact time τ as the first event (regardless of which edge fires), its probability density equals:

$$\mathcal{X}_{\text{event}}(\tau) = \hat{r}(v + \tau) \mathcal{S}(\ell, v, \tau) = \underbrace{\hat{r}(v + \tau)}_{\text{total rate at time } \tau} \times \underbrace{\exp \left(- \int_0^\tau \hat{r}(v + s) ds \right)}_{\text{survival up to } \tau} = - \frac{\partial \mathcal{S}(\ell, v, \tau)}{\partial \tau}$$

At time τ , several autonomous edges are enabled with rates $r_1(t), \dots, r_n(t)$. Each entity has its own rate independent from others. One of them fires earlier than others after which the state of the world changes to a new one continuing the process, without the global time being reset. However, it matters which autonomous transition has happened, as it determines the successor state and the timing of any strategy over the horizon. Therefore, these autonomous transitions are competing in time to be the next event.

This occurrence leads to an instantaneous branch weight at time $\tau \in (0, \tau^*]$ and edge j with $\hat{r}(v + \tau) > 0$:

$$\mathcal{W}_j(\tau) = \frac{r_j(v + \tau)}{\hat{r}(v + \tau)} \in [0, 1]$$

where intuitively, $\sum_j \mathcal{W}_j(\tau) = 1$. The joint probability density is, therefore,

$$\mathcal{X}_j(\tau) = \mathcal{X}_{\text{event}}(\tau) \cdot \mathcal{W}_j(\tau) = r_j(v + \tau) \exp \left(- \int_0^\tau \hat{r}(v + s) ds \right)$$

Consequently, the cumulative probability that the first autonomous event in $(0, \delta]$ occurs at edge j equals

$$p_{\text{jump}_{(\delta, j)}} = \int_0^\delta \mathcal{X}_j(\tau) d\tau = \int_0^\delta r_j(v + \tau) \exp \left(- \int_0^\tau \hat{r}(v + s) ds \right) d\tau$$

Also, the probability that some autonomous jump occurs before δ

$$\begin{aligned} p_{\text{jump}_\delta} &= \int_0^\delta \mathcal{X}_{\text{event}}(\tau) d\tau = \int_0^\delta \hat{r}(v + \tau) \exp \left(- \int_0^\tau \hat{r}(v + s) ds \right) d\tau \\ &= - \int_0^\delta \frac{\partial \mathcal{S}(\ell, v, t)}{\partial t} dt = \mathcal{S}(\ell, v, 0) - \mathcal{S}(\ell, v, \delta) = 1 - \mathcal{S}(\ell, v, \delta) = 1 - \exp \left(- \int_0^\delta \hat{r}(v + t) dt \right) \end{aligned}$$

Eventually, the conditional probability of adopting a particular autonomous transition $j \in \{1, \dots, n\}$ before time δ starting from state (ℓ, cv) , is

$$\begin{aligned}
p_{(j|jump_s)} &= \frac{\int_0^\delta \mathcal{X}_j(\tau) d\tau}{\int_0^\delta \mathcal{X}_{event}(\tau) d\tau} = \frac{\int_0^\delta r_j(v + \tau) \exp\left(-\int_0^\tau \hat{r}(v + s) ds\right) d\tau}{\int_0^\delta \hat{r}(v + \tau) \exp\left(-\int_0^\tau \hat{r}(v + s) ds\right) d\tau} \\
&= \frac{\int_0^\delta r_j(v + \tau) \exp\left(-\int_0^\tau \hat{r}(v + s) ds\right) d\tau}{1 - \exp\left(-\int_0^\delta \hat{r}(v + t) dt\right)}
\end{aligned}$$

Reach Probability Value. Our goal is to compute the maximum probability -also valid for minimum probability considering *infimum/min* instead of *supremum/max*- over all policies/strategies π of reaching ℓ_{target} within the time bound of d . Lipschitz continuity of the rate expressions (together with non-punctuality of guards, see below) intuitively furthermore implies that delaying (or advancing) a scheduled interactive action of the strategy by a small amount will also have controlled effect on the reward, i.e. in our case, on the reach probability.

The problem with this intuition is that even slightly delaying or advancing a transition may block later transitions due to guard conditions and clock resets. Insisting on non-punctual guard conditions, however, resolves this issue. As mentioned before, the c -progressivity results in time-bounded paths being enumerable. This implies that an optimal d -bounded strategy that interacts (i.e. provides control actions) only at times $k\Delta$ can be computed trivially by just enumerating all possible paths and collecting the optimal strategy among them.

Taking all of this into account, we then define a cumulative distribution function that enumerates all time points at which either a race of autonomous transitions happens or a policy chooses a forced urgent transitions, in a single formula. We define $\mathcal{V}^{max}(\ell, v, \tau)$, the final value function, as follows:

$$\mathcal{V}^{max}(\ell, v, \tau) = \sup_{\pi} (Pr_{reach}(\ell \rightarrow \ell_{target}) | t \leq d)$$

with the boundary conditions:

$$\mathcal{V}^{max}(\ell_{target}, v, t) = 1 \quad (t \leq d), \quad \mathcal{V}^{max}(\ell, v, d) = 0 \quad (\ell \neq \ell_{target}),$$

which means the value function equals to 1 if $\ell = \ell_{target}$ and $t \leq d$ while in case time exceeds d and $\ell \neq \ell_{target}$, the value is equal to 0 by definition as there is no time remaining to reach ℓ_{target} .

Let $\eta = \eta(\ell, v, t) \in [0, \infty]$ be the earliest time offset $\tau \geq 0$ (measured from t) such that an urgent interactive guard becomes true along the clock valuation slope $v(\tau) = v + \tau$. The urgent interactive transition is either controllable (to be chosen by policy π at time τ_π), or uncontrollable which will be taken immediately. If no urgent

guard occurs before the deadline, we set $\eta = \infty$. Once the forced urgent transition taken by the policy π , then the maximum probability value that a policy can achieve becomes:

$$\mathcal{V}^{max}(\ell, v, t) = \max_{a \in \mathcal{U}(\ell, v)} \mathcal{V}^\pi(\ell_a, (v + \eta) \oplus R_a, t)$$

where $v \oplus R_a$ resets the respective clocks on that jump and $\mathcal{U}(\ell, v)$ is the set of enabled urgent interactive transitions at the valuation where the guard becomes true.

As we evolve inside a location ℓ until the first urgent guard taken ($\tau = \{\eta, \tau_\pi\}$) or until the horizon offset ($\tau = d - t$) becomes exhausted, no matter which one arrives first, we can define the integration limit inside the horizon by $\tau^* := \min\{\eta, \tau_\pi, d - t\}$.

Eventually, given a policy π within time boundary $[0, d]$, an IMCC induces a probability measure over a set of timed execution paths. The probability of reaching the target location ℓ_{target} within the time bound is obtained by summing the probabilities of all paths that reach the target, which may involve the occurrence likelihood of several events namely, (i) probability of no autonomous transition firing, (ii) probability of some autonomous transition jumps taking place and (iii) probability of forced interactive transitions being taken.

We integrate τ from t up to each transition that fires at τ^* , then subdivide the state space in case that particular transition did not already hit the target location. Therefore, the final value function of the global optimal reachability over all admissible policies on horizon d is as follows:

$$\begin{aligned} \mathcal{V}^{max}(\ell, v, t) = & \int_0^{\tau^*} \underbrace{\left[\exp\left(-\int_0^\tau \hat{r}(v+s)ds\right) \right]}_{\text{Survival probability up to } \tau} \underbrace{\hat{r}(v+\tau)}_{\text{total rate}} \sum_j \underbrace{\mathcal{W}_j(\tau)}_{\text{Branch weighting}} \underbrace{\mathcal{V}(\ell_j, (v+\tau) \oplus R_j, t+\tau)}_{\text{value at successor } j} d\tau \\ & + \underbrace{\exp\left(-\int_0^{\tau^*} \hat{r}(v+s)ds\right)}_{\text{no autonomous event before time } \tau^*} \times \mathcal{B}(\ell, v, (t, \tau^*)) \end{aligned}$$

In this formula, $\mathcal{B}(\ell, v, (t, \tau^*))$ reflects the piece-wise boundary term at $\tau^* = \min\{\eta(\ell, v), \tau_\pi, d - t\}$, where:

$$\mathcal{B}(\ell, v, (t, \tau^*)) = \begin{cases} 1, & \tau^* \leq d - t \& \ell = \vec{tl}, \\ \max_{a \in \mathcal{U}(\ell, v(\tau^*))} \mathcal{V}(\ell_a, (v + \tau^*) \oplus R_a, t + \tau^*), & \tau^* = \eta, \\ \mathcal{V}(\ell_\pi, (v + \tau^*) \oplus R_a, t + \tau^*) & \tau^* = \tau_\pi, \\ 0 & \tau^* \geq d - t \& \ell \neq \vec{tl} \end{cases}$$

and η is the earliest urgent-guard time, τ_π is the next controllable firing time allowed by the policy, and $d - t$ is the remaining horizon. This indicates the probability of the occurrence of an interactive transition (urgent or controllable one) or no jump at all, before time t . If no interactive jump is taken before the deadline and the ℓ_{target} is not reached, the boundary payoff is 0. When controller fires its transition at next allowed time point tick τ_π , then the boundary term captures "wait until the tick (survival), then fire π 's action".

As stated before and assumed in the race of autonomous transitions from location ℓ , one will eventually fire at some time point $\tau^* \in [t, d]$. Intuitively, it can be derived that until time point τ , the race is governed by the total outflow rate (total hazard) $\hat{r}(v)$ while in case $\tau \leq \tau^*$, the race in location ℓ is interrupted by the triggered urgent transition. Therefore, the actual time point of taking any of those transitions is: $\tau^* \leq d$ according to definition of IMCCs being time-progressive.

In addition, the probability from next location ℓ' has been considered by incorporating the weight of probability distribution of the transition that has been taken into the value function of the successor state. If race of autonomous transitions is purely stochastic and happens at time t' , the next location ℓ' will be selected by the probability weight of each transition.

Thereby, the merged piecewise set of formulas for final value function covers all possible valuation of probability in many branching possibilities induced by different outflows at each location due to autonomous transition race in IMCC. Therefore, the overall probability of being in ℓ_{target} by some deadline d is computable via an integral over all branching splits, only when time discretization is applied.

It should be noted that our derivations have refrained from specifying any particular shape of the rate function $r_i(\ell \rightarrow \ell')$ and the only essential requirement is for these functions to be well-defined and continuous/piecewise continuous on the clock region of $[0, M]^k$. Within this constraint, one can embed linear, affine, non-affine, exponential and even more intricate functional dependencies on the clock valuations. As our computation scheme only cares for the existence of integrals and boundary conditions, once a particular function form is set, the same scheme leads to a fully explicit expression of time-bounded probability of reaching a desired location.

Computation Scheme. In computing optimal reach probability in time-bounded IMCC, we first construct a finite MDP approximation of the continuous-time, clock-dependent semantics over the horizon d , and then apply standard MDP algorithms to that finite model. Since the objective is to reach target $\vec{t}\vec{l}$ within horizon, we only consider runs of duration at most d .

Once the duration of a run exceeds d , it is irrelevant for calculating the probability of reaching ℓ_{target} within d . Thus, IMCC's behavior after $t > d$ does not affect the decision. As each clock valuation is confined to $[0, M]$, the set of possible valuations is a compact subset of $\mathbb{R}^k$. This boundedness as well as time discretization are crucial for defining a finite partition of the continuous state space.

We fix a small rational clock granularity $\Delta_v > 0$ and partition clock valuation space $[0, M]^k$ into finitely many guard-stable regions $\mathcal{R}_{\Delta_v}$. Each region $\rho_v \in \mathcal{R}_{\Delta_v}$ has two particular properties namely, (i) Guard-stability where for each ℓ , every guard predicate has constant truth value on each ρ_v , and (ii) the bounded variation of any rate function $r_i(\ell, \ell')(v)$ by the Lipschitz condition: $|r_i(v) - r_i(v')| \leq L_c \|v - v'\|_\infty \leq L_c \Delta_v$, for all $v, v' \in \rho$.

We now discretize time horizon $[0, d]$ into intervals $[n\Delta_t, (n+1)\Delta_t]$ with time step of length $\Delta_t > 0$. Over each slice of length Δ_t starting from (ℓ, v) , the integrated effect of autonomous transitions is approximated, distributing probabilities among them according to the overall rate outflow; this yields a discrete-time approximation. The combination of each location $\ell \in L$, each region ρ_v in the partition, and time grid of $t_n = n\Delta_t$ and $K = \lceil d/\Delta_t \rceil$ results in a *hyper-state* $(\ell, \rho_v, n) \in L \times \mathcal{R}_{\Delta_v} \times \{0, \dots, K\}$.

Transitions among hyper-states, thus, occur based on approximate autonomous transition rates. In IMCCs, per definition, forced interactive transitions take precedence over autonomous transitions and we cannot remain in (ℓ, v) if v belongs to a guard condition set. Let $\eta(\ell, \rho) \in (0, \infty]$ denote the earliest $\tau \in [0, \Delta_t]$ measured from each t_n during slice $[t_n, t_{n+1}]$ that some urgent interactive transition becomes enabled along the time evolution in region ρ_v . Thus, $\tau^*(\ell, \rho_v) = \min\{\eta(\ell, \rho_v), \Delta_t\}$ is set as an in-slice cut point. Guard stability implies that within $(0, \tau^*]$ there are no forced transitions except possibly at τ^* . Moreover, because guards are piecewise defined, the set of offsets where a destination region changes is finite.

Finite MDP. Combining locations, clock regions, and time layers yields a finite MDP on (ℓ, ρ, n) . Transition probabilities over one time step are built, same as before, by (a) integrating first-event densities of autonomous edges up to τ^* , and (b) sending the no-event survival probability to a single boundary destination at the cut (urgent inside the slice or a controllable choice at the tick).

Standard MDP algorithms (value iteration [Put05], backward induction [Bel57b], or graph-based analysis [BK08]) can compute whether the probability of reaching $(\ell_{\text{target}}, \rho, n)$ for some $n \leq \lfloor d/\Delta_t \rfloor$ is $\geq \alpha$.

$$P_{\Delta, \delta}^d(\ell, \rho_v, n) = \max_{\pi} \Pr \left[\text{reach} \left(\vec{t}, \rho'_v, n' \right) \mid n' \leq \lfloor d/\Delta_t \rfloor \right]$$

Since each step is a finite distribution and each hyper-state has a finite number of successors, this maximum probability of reaching a target state can be computed. We then fix Δ_t (the length of discretized time-intervals) and Δ_v (the clock partition) such that the approximation error is within an arbitrarily small margin ϵ .

8.2 How Far Are We From The Optimal Strategy

When we speak of an optimal strategy in our Interactive Markov Chain with Clocks (IMCC) setting, we are confronting an ideal that is generally not directly computable in the unrestricted model. The system's continuous time dimension, the stochastic branching of its clock-dependent transitions, and the combinatorial explosion caused by multiple simultaneous agents collectively create a search space so vast that no algorithm can enumerate it to completion -as shown in the previous chapter.

The ϵ -optimal variant in our IMCC fares no better in terms of decidability: as a result of Lemma 7.2, there is no procedure that can, in general, decide whether a given policy lies within ϵ of the optimal one. Nevertheless, this does not prevent us from computing the value induced by a concrete policy and quantifying its deviation from the optimum in the time-bounded setting.

Even if the identity of the optimal policy is unknown, one fact is evident in real-time planning: taking our rescue mission as an instance, every unit of delay pushes us further from the optimum. Survival functions for all casualty classes are strictly decreasing; postponing an action, whether due to network latency, planning overhead, sudden changes in the environment, or agent hesitation, reduces the maximum achievable value and inevitably widens the gap between what we achieve and what could have been achieved in an ideal, zero-latency world. This observation gives us a practical lever: rather than pursuing the optimal strategy directly, we analyze and compute how time-dependent delays translate into measurable deviations from the theoretical optimum.

Conceptually, that distance grows each time the system hesitates, re-routes, or waits for new information. By treating delay as a "cost of hesitation" or "penalty", we can calculate exactly how much performance slips away. In effect we transform a theoretically undecidable optimization problem into an engineering question: *How much performance do we lose, and where does the loss happen?* Quantifying the impact of each delay source may not reveal the perfect plan, but it tells us how much we are falling behind.

8.2.1 Error Calculation Scheme

Let $\mathcal{V}(\ell, v, t)$ be the true optimal reachability value. Along any trajectory that stays in the same location up to time $t + w \leq d$, we assume that in our model implementation, the controller π reacts on a time-grid $T_\Delta = \{0, \Delta_t, 2\Delta_t, \dots\}$, hence, any guard that becomes true at a non-grid instant t^d is serviced after an extra waiting time.

$$w = \left\lceil t^d / \Delta_t \right\rceil \Delta_t - t^d \quad (\in (0, \Delta])$$

During that wait, the system keeps flowing under the uncontrolled hazard (leaking valve or patient bleeding).

$$\hat{r}(t) = \sum_j r_j(v + t)$$

Some paths will autonomously jump out of the location before the controller fires, lowering the eventual reach probability; a valve breaks or a patient suddenly dies. Depending on the scenario, the extra drift or hesitation could even possibly move clocks into a better region, in other words, lead to better decisions. That, in the aforementioned scenarios, is of low probability. The discretizations error against the continuous-time optimum, therefore, becomes a cumulative error calculated over the path. We, therefore, introduce three types of error – on the local and global scale – that can occur along each path that the strategic controller chooses.

$$\varepsilon_{\text{total}} = \underbrace{\varepsilon_{\text{time}}}_{\text{wait error}} + \underbrace{\varepsilon_{\text{grid}}}_{\text{grid rounding}} + \underbrace{\varepsilon_{\text{choice}}}_{\text{sub-optimal choice}}$$

1. *Wait Error.* Assuming that no urgent event becomes enabled on $[t, t + w]$ where $0 < w \leq \Delta_t$, the delay of firing a decision at $[t + w]$ instead of $[t]$ along the clock evolution $v(s) = v + s$ results in:

$$|\mathcal{V}(\ell, v, t + w) - \mathcal{V}(\ell, v, t)| \leq \int_0^w \hat{r}_\ell(v + s) ds \leq \hat{r}_{\max} \cdot w$$

where $\hat{r}_{\max} := \sup_{\ell \in L, v \in [0, M]^k} \hat{r}_\ell(v)$, since every rate expression r_j is continuous on the clock domain $[0, M]^k$.

Even a tighter bound is applicable using:

$$|\mathcal{V}(\ell, v, t + w) - \mathcal{V}(\ell, v, t)| \leq 1 - \exp \left(- \int_0^w \hat{r}_\ell(v_\rho + s) ds \right) \leq \hat{r}_{\max} \cdot w$$

If an urgent event triggers at $\eta \leq w$, we change the bounds of the integral to $[0, \eta]$ and restart the calculation from the successor state due to uncontrollable actions being pre-emptive.

2. *Grid Rounding Error.* A slice starts at grid time t_n in location ℓ with a clock valuation v lying inside a guard-stable region of the clock partition. Let v_ρ be the representative of this region where $\|v - v_\rho\|_\infty \leq \Delta_v$. Clocks advance with unit velocity u ; within the slice, the system may either fire an autonomous jump at some offset $0 < \tau \leq \tau^*$, or survive until τ^* with no autonomous jump. For each autonomous edge j enabled in ℓ , the first-event density at time τ is computed as before with the true valuation $\mathcal{X}_{event}(\tau)$. If we round the valuation to the region representative v_ρ everywhere inside the slice, the one-slice rounding error is the lc distance between the true and rounded kernels which then becomes:

$$\varepsilon_{grid}(\ell, \rho) \leq L_c \cdot \Delta_v$$

3. *Sub-Optimal choice Error.* The resulting finite MDP gives $\mathcal{V}^*$ and the optimal Q -values:

$$Q_n^*(s, a) := \sum_{s'} P(s' | s, a) V_{n+1}^*(s').$$

to be computed by the a backward induction algorithm over the time horizon and a realized policy, where:

$$V_n^*(\ell, \rho) = \max_{a \in \mathcal{U}(\ell, \rho, n)} Q_n^*(s, a)$$

At the decision point $s_n = (\ell, \rho, n)$ the realized policy $\hat{\pi}$ picks $a_n = \hat{\pi}(s_n)$. The local regret is

$$\varepsilon_{choice}(s_n) = V_n^*(s_n) - Q_n^*(s_n, a_n) \geq 0.$$

The finite-horizon performance difference then results in:

$$\mathcal{V}_0^*(s_0) - \mathcal{V}_0^{\hat{\pi}}(s_0) = \mathbb{E}_{s_0, \hat{\pi}} \left[\sum_{n=0}^{K-1} \varepsilon_{choice}(s_n) \right],$$

expectation w.r.t. the trajectory distribution induced by $\hat{\pi}$. In practice, we can bound it by

$$\mathcal{V}_d^*(s_0) - \mathcal{V}_d^{\hat{\pi}}(s_0) \leq \sum_{n=0}^{K-1} \max_s \varepsilon_{choice}(s),$$

or compute the approximated expectation up to ϵ -accuracy by forward simulation using the known kernel P and $\hat{\pi}$.

1. Finite-Horizon Setup (ℓ, ρ_v)

- Compute the in-slice cut (where evolution stops) $\tau^*(\ell, \rho_v)$ for each layer

For each time layer $(n = 0, \dots, K - 1)$, set

$$\tau^*(n, \ell, \rho_v) := \min(\eta(\ell, v_\rho), \tau_\pi(\ell, v_\rho), d - t_n, \Delta_t)$$

Where $\eta(\ell, v_\rho)$ is the earliest offset in $(0, \Delta_t]$ (measured from t_n) at which an urgent transition becomes enabled along evolution. $\tau_\pi(\ell, v_\rho)$ indicates the earliest offset in $(0, \Delta_t]$ at which the controller intends to fire a controllable action (and fires at Δ_t on ticks). $d - t_n$ cuts the slice if the horizon ends before η or τ_π .

- Build subintervals for each autonomous edge

For every autonomous edge $j \in A(\ell)$ and each layer n , we partition the open interval $(0, \tau^*(n, \ell, \rho_v)]$ into finitely many subintervals ϱ_m^j such that, for any τ inside a fixed ϱ_m^j , we have:

- Jump at time $t_n + \tau$: clocks reset $cv_\rho(\tau) \mapsto v_\rho(\tau) \oplus R_j$, location changes $\ell \rightarrow \ell_j$. Each ϱ_m^j carries a single labeled successor (ℓ_j, ρ'_m) .
- Then drift (grid rounding) for the remaining $\Delta_t - \tau$ to reach time t_{n+1} .
- Record the resulting region ρ'_m .

- Record the boundary destination(s)

For each layer n :

- If $\tau^*(n, \ell, \rho_v) = \eta(\ell, v_\rho)$ (urgent inside the slice): we compute the unique urgent boundary destination at the end of the slice by applying the urgent reset at offset τ^* and drifting for $\Delta_t - \tau^*$, denoted by (ℓ_u, ρ_u) at layer $n + 1$.
- If $\tau^*(n, \ell, \rho_v) = \tau_\pi(\ell, v_\rho)$ (controllable fires inside the slice): for each controllable action a that is enabled at τ^* , we compute its boundary destination (ℓ_π, ρ_π) at layer $n + 1$.
- If $\tau^*(n, \ell, \rho_v) = d - t_n$ (horizon ends first): no boundary jump occurs which is handled in the boundary term $\mathcal{B}$.

2. Initialization and Terminal Condition (Layer K)

We set terminal values at time t_K . For every ℓ, ρ , we set $\mathcal{V}_K(\ell, \rho) = 1$ if $\ell = \ell_{target}$, else $\mathcal{V}_K(\ell, \rho) = 0$. Evaluation of a realized policy π , sets $\mathcal{V}_K^\pi(\ell, \rho) = \mathcal{V}_K$.

3. Backward Induction for $n = K - 1, \dots, 0$

For each hyperstate $\mathcal{H} = (\ell, \rho, n)$, we perform these steps:

- Accumulate one-slice autonomous transition probability to every successor at layer $(n + 1)$

- Initialize all successor cumulative probability of first autonomous event $p_{jump_{(\delta,j)}}(\ell', \rho') \leftarrow 0$.
- For each autonomous edge $j \in A(\ell)$, and for each subinterval ϱ_m^j labeled by successor (ℓ_j, ρ_m) , compute

$$Pr_{edge}(j, m) = \int_{\tau^* \in \varrho_m^j} r_j(v_\rho(\tau)) \cdot p_{surv}(\tau) \cdot d\tau$$

- Add this to the appropriate bin: $p_{jump_{(\delta,j)}}(\ell_j, \rho'_m) += Pr_{edge}(j, m)$.
- *Compute the survival probability to the cut.*
Set $p_{surv} = S(\tau^*[n, \ell, \rho]) = \exp(-\int_0^{\tau^*} \hat{r}(s)ds)$. This is the probability that no autonomous jump occurs before the cut τ^* .
- *Evaluate the piecewise boundary term $\mathcal{B}(\ell, cv_\rho(t_n, \tau^*))$*
 - If $(\tau^* = \eta(\ell, cv_\rho))$ (urgent inside), then

$$\mathcal{B}(\ell, cv_\rho(t_n, \tau^*)) = \max_{a \in \mathcal{U}(\ell, cv_\rho(\tau^*))} V_{n+1}(\ell_a, \rho_a)$$

where each (ℓ_a, ρ_a) is computed by applying the urgent reset at offset τ^* , then drift to t_{n+1} .

- If $\tau^* = \tau_\pi(\ell, cv_\rho)$ (controller fires inside), then

$$\mathcal{B}(\ell, cv_\rho(t_n, \tau^*)) = V_{n+1}(\ell_\pi, \rho_\pi),$$

where (ℓ_π, ρ_π) is the boundary destination of the action chosen by π at offset τ^* .

- If $\tau^* \geq d - t_n$ (horizon ends first), then
 $\mathcal{B}(\ell, cv_\rho(t_n, \tau^*)) = 1$ if $(\ell = \ell_{\text{target}})$, and $\mathcal{B} = 0$ otherwise.
- *Value update using autonomous transition probability and boundary value.*
The slice's total value is the sum of:

- the accumulated autonomous contributions (first events inside the slice), plus

$$\text{Autonomous part} = \sum_{\ell', \rho'} p_{jump_{(\delta,j)}}(\ell', \rho') V_{n+1}(\ell', \rho'),$$

- the boundary contribution (no autonomous event before the cut, then boundary behavior).:

$$\text{Boundary part} = p_{surv} \times \mathcal{B}(\ell, cv_\rho(t_n, \tau^*)).$$

We, therefore, set

$$V_n(\ell, \rho) = \text{Autonomous part} + \text{Boundary part}.$$

- *Controllable “tick” case and Q-values*

Since the controller fires its choice at the tick ($\tau^* = \Delta_t$), we compute Q-values for each controllable action a instead of using a single boundary condition $\mathcal{B}$:

$$Q_n(\ell, \rho, a) = \sum_{\ell', \rho'} p_{jump(\delta, j)}(\ell', \rho') V_{n+1}(\ell', \rho') + p_{surv} V_{n+1}(\ell_a, \rho_a),$$

we then set $V_n(\ell, \rho) = \max_a Q_n(\ell, \rho, a)$.

- *Policy evaluation and local choice regret*

If a realized policy π is supplied and makes a boundary choice in this slice, we let $a_\pi = \pi(\ell, \rho, t_n)$ and set $V_n^\pi(\ell, \rho) = Q_n(\ell, \rho, a_\pi)$.

We then record local regret (sub-optimal choice error)

$$\varepsilon_{\text{choice}}(\ell, \rho, n) = V_n(\ell, \rho) - V_n^\pi(\ell, \rho) \geq 0.$$

When a guard of a controllable edge becomes true, the edge fires instantly, and whatever decision the controller was still contemplating is discarded. However, the global clock advances by the actual fire time τ (typically $\tau < w$). Controllable edge (rescuing a patient or fixing a valve) can be taken only if no hazard has preempted the slice and the controller is at an authorized decision instant ($\theta = k\Delta$). After a jump, we re-enter the algorithm with a fresh state (ℓ', cv', θ') , which we call *restart point*; all subsequent calculations (decision sets, probabilities, error bookkeeping) are relative to this new clock value and its remaining time horizon $d - \theta'$.

8.2.2 Worst-Case Deviation Over the Time Horizon

A run of any strategy visits at most $K = \lceil d/\Delta_t \rceil$ time layers. Along those layers, it intersects at most $(1 + \sum_{i=1}^k \lceil \Delta_t/\Delta_v \rceil)K = (1 + k\lceil \Delta_t/\Delta_v \rceil)K$ valuation boxes for a k -dimension grid over K time slices, which is therefore, the upper bound for the number of hyper-states (ℓ, ρ_v, n) .

Let N_{off} be the number of slices in which an urgent event became enabled off-grid (worst case: $N_{\text{off}} = K$). This eventually makes the horizon-wise error bound explicit in terms of time budget and sets an upper bound on error components as follows: For $\varepsilon_{\text{time}}$, we will have

$$\varepsilon_{\text{time}} \leq \hat{r}_{\text{max}} \sum_{n \in N_{\text{off}}} w_n \leq \hat{r}_{\text{max}} N_{\text{off}} \Delta_t \leq \hat{r}_{\text{max}} d$$

As for $\varepsilon_{\text{grid}}$, this bound results in

$$\varepsilon_{\text{grid}} \leq L_c \cdot d$$

Since the controller do not necessarily take action at every instance, then ε_{choice} only depends on the actions that are actually taken and the grid kernel. The tighter bound will be a result of decision consideration together with the length of time budget d , thus, the observed count N . Therefore:

$$|\mathcal{V} - \mathcal{V}_\Delta| \leq (\hat{r}_{\max} + L_c) \cdot d + \varepsilon_{choice}^\pi$$

A conservative global bound (worst case over a path of $K = \lceil d/\Delta_t \rceil$ slices) is

$$\left| V(\ell_0, v_0, 0) - V^{\hat{\pi}}(\ell_0, v_0, 0) \right| \leq \sum_{n=0}^{K-1} \left(\underbrace{1 - e^{-\hat{r}_{\max} \Delta_t}}_{\text{wait at most } \Delta_t} + \underbrace{\varepsilon_{\text{round}}(\ell_n, \rho_n)}_{\text{grid rounding}} + \underbrace{\varepsilon_{\text{choice}}(s_n)}_{\text{of a realized policy}} \right)$$

where (ℓ_n, ρ_n) is the hyper-state visited at layer n (worst-case bound uses maxima over admissible states at layer n). If the actual trajectory distribution (under $\hat{\pi}$) is known, the sum will be replaced by its expectation. Since all factors are observed at run time, the absolute value of the difference between actual reachability probability and optimal one is almost always below the worst-case.

$$|p_{grid} - p_{opt}| \leq \sum_k \hat{r}(v_k) w_k + L_c \sum_k w_k.$$

8.3 Implications

Up to this point, the focus of the chapter has been on the theoretical side of the problem. We established that an optimal strategy in our setting is not computable, which immediately rules out the possibility of an exact algorithmic solution. To move forward, we introduced approximation methods that allow us to compute an ε -optimal strategy and estimate the optimal reachability value, and we derived bounds that quantify the distance from the true optimum. In particular, we showed how to calculate an upper bound on the error, thereby providing a measure of how far any synthesized strategy can deviate from the uncomputable ideal.

What we have achieved here is therefore a complete mathematical treatment of the problem: from non-computability, through approximation, to explicit error margins. This closes the circle of complexity and solution techniques within this chapter, and sets the stage for the next step, where these theoretical insights will be connected to practical algorithms.

Algorithm 1: Time-Bounded IMCC Grid DP with Q-Values and Error Certificate

```

Data : IMCC  $I = (L, C, \text{rates}, \text{guards}, \text{resets})$ ,      25 Return:  $V[0][\ell_0, \rho_0]$  (optimal start value);
        guard-stable partition  $\mathcal{R}_{\Delta_v}$  with reps  $cv_\rho$ 
Input : Target  $\ell_{\text{target}}$ , horizon  $d$ , time step  $\Delta_t$ , clock 26 if policy evaluation then
        width  $\Delta_v$ , policy  $\hat{\pi}$ 
        also return  $V^\pi[0][\ell_0, \rho_0]$ , and the error certificates
         $\varepsilon_{\text{wait}}, \varepsilon_{\text{round}}, \varepsilon_{\text{choice}}$ 
         $(\varepsilon_{\text{total}} = \varepsilon_{\text{wait}} + \varepsilon_{\text{rounding}} + \varepsilon_{\text{choice}})$ .
27
28
29 -----Details of The Steps-----
    // 1) Build autonomous one-step edge
    probability to successors at layer  $n+1$ 
30 foreach  $s' = (\ell', \rho', n+1)$  do
31    $P_{\text{jump}(\delta, j)}[s'] \leftarrow 0$ 
32 foreach  $j \in A(\ell)$  do
33   foreach  $(I, (\ell_j, \rho')) \in \text{intervals}[\ell, \rho, j]$  do
34      $Pr_{\text{edge}}(j, m) \leftarrow$ 
        $\int_{\tau \in I} r_j(v_\rho + \tau) \cdot \exp\left(-\int_0^\tau \hat{r}(v_\rho + s) ds\right) d\tau$ 
35      $P_{\text{jump}(\delta, j)}[(\ell_j, \rho', n+1)] \leftarrow$ 
        $P_{\text{jump}(\delta, j)}[(\ell_j, \rho', n+1)] + Pr_{\text{edge}}(j, m)$ 
    // 2) Survival (no autonomous inside the
    slice) up to the in-slice cut
36  $p_{\text{surv}}(\ell, v_\rho, \tau^*) \leftarrow \exp\left(-\int_0^{\tau^*} \hat{r}(v_\rho + s) ds\right)$ 
    // 3) Urgent inside the slice (singleton
    action)
37 if  $\tau^*[\ell, \rho] < \Delta_t$  then
38    $s_u \leftarrow (\ell_u, \rho_u, n+1)$ 
39    $V[n][\ell, \rho] \leftarrow \sum_{s'} p_{\text{jump}(\delta, j)}[s'] \cdot V[n+1][s'] +$ 
      $p_{\text{surv}} \cdot V[n+1][s_\square]$ 
40   if policy evaluation then
41      $V^\pi[n][\ell, \rho] \leftarrow V[n][\ell, \rho]$ 
     // no choice  $\Rightarrow$  no regret
     // Errors in this slice:
42      $\varepsilon_{\text{round}} \leftarrow \varepsilon_{\text{round}} + L_c \Delta_v$ 
43   continue
    // 4) No urgent in the slice: controllable
    choice at the tick
44 foreach  $a \in \text{Act}(\ell, \rho)$  do
45    $s_a \leftarrow (\ell_a, \rho_a, n+1)$ 
    //
    Q-update (the only explicit formula kept):
46    $Q[a] \leftarrow \sum_{s'} p_{\text{jump}(\delta, j)}[s'] \cdot V[n+1][s'] + p_{\text{surv}} \cdot$ 
      $V[n+1][s_a]$ 
47  $V[n][\ell, \rho] \leftarrow \max_{a \in A_C(\ell, \rho)} Q[a]$ 
48 if policy evaluation then
49    $a_\pi \leftarrow \hat{\pi}(\ell, \rho, n)$ 
50    $V^\pi[n][\ell, \rho] \leftarrow Q[a_\pi]$ 
51    $\varepsilon_{\text{choice}} \leftarrow \varepsilon_{\text{choice}} + (V[n][\ell, \rho] - V^\pi[n][\ell, \rho])$ 

1 Derived:  $t_n \leftarrow n\Delta_t$ ,  $K \leftarrow \lceil d/\Delta_t \rceil$ 
2 Initialize error accumulators
3  $\varepsilon_{\text{wait}}, \varepsilon_{\text{round}}, \varepsilon_{\text{choice}} \leftarrow 0$ .
4 Terminal layer init:
5 foreach  $(\ell, \rho) \in L \times \mathcal{R}_{\Delta_v}$  do
6    $V[K][\ell, \rho] \leftarrow \mathbf{1}\{\ell = \ell_{\text{target}}\}$ 
7   if policy evaluation then
8      $V^\pi[K][\ell, \rho] \leftarrow V[K][\ell, \rho]$ 

9 Preprocess geometry and in-slice urgent cuts (once
  per  $(\ell, \rho)$ ):
10 foreach  $(\ell, \rho) \in L \times \mathcal{R}_{\Delta_v}$  do
11    $\tau^*[\ell, \rho] \leftarrow$  earliest offset in  $(0, \Delta_t]$  where an urgent
     becomes true along drift from  $(\ell, v_\rho)$ ; if none, set
      $\tau^*[\ell, \rho] \leftarrow \Delta_t$ 
12   foreach autonomous edge  $j \in A(\ell)$  do
13      $\text{intervals}[\ell, \rho, j] \leftarrow$  partition of  $(0, \tau^*)$  into
       subintervals  $\rho_m^j$ 
14   if  $\tau^*[\ell, \rho] < \Delta_t$  then
15      $(\ell_u, \rho_u) \leftarrow$  boundary destination obtained by
       urgent at  $\tau^*$  followed by drift of  $\Delta_t - \tau^*$ 
16   else
17     foreach controllable  $a \in \text{Act}(\ell, \rho)$  do
18        $(\ell_a, \rho_a) \leftarrow$  boundary destination at the tick
        $t_{n+1}$  for action  $a$ 

19 Backward induction across time layers:
20 for  $n \leftarrow K-1$  to 0 do
21   foreach  $(\ell, \rho) \in L \times \mathcal{R}_{\Delta_v}$  do
     // 1) Build autonomous one-step masses to
     successors at layer  $n+1$ 
     // 2) Survival (no autonomous inside the
     slice) up to the in-slice cut
     // 3) Urgent inside the slice (singleton
     action)
     // 4) No urgent in the slice:
     controllable choice at the tick
     // Errors in this slice (optional):
22   if an enable time  $\tau^* < \Delta_t$  for the intended control
     is detected then
23      $w \leftarrow \Delta_t - \tau^*$ ;
      $\varepsilon_{\text{wait}} \leftarrow \varepsilon_{\text{wait}} + (1 - \exp(-\int_{\tau^*}^{\Delta_t} \hat{r}(v_\rho + s) ds))$ 
     // or the crude bound:  $\varepsilon_{\text{wait}} += \hat{r}_{\text{max}} w$ 
24    $\varepsilon_{\text{round}} \leftarrow \varepsilon_{\text{round}} + \varepsilon_{\text{round}_{n+1}}$ 

```

Model-Checking Tools — From Theory to Practice

Urban search-and-rescue, as our case scenario, operates on a life-or-death clock, so the familiar strategy of *keep testing until nothing fails* is simply inadmissible. When continuous timing, random faults, and close-coupled robot teams combine, only a method that examines every execution path can offer real assurance. Three decades of work on model checking have shown that such exhaustive state-space exploration can still yield formal guarantees, even for systems whose behavior is largely nondeterministic or stochastic.

Model checking supports decision-making procedures by translating informal concern, for instance, "Could all victims ever be rescued?" into temporal-logic formulae an algorithm can answer definitively. Also, its probabilistic extensions attach concrete numbers to those answers, such as the probability that at least one victim dies before help arrives, enabling planners to weigh resources against a quantified residual risk instead of relying on individual's estimates or even on the great performance in previous rescue mission as it might not be at all applicable to the current one. Pure simulation alone often fails to reveal the rare yet critical scenarios that occur with very low probability in the real world. Formal model-checking tools complement simulation by exhaustively exploring the model's entire behavioral space, thereby providing an additional level of assurance that these edge cases have been identified.

9.1 Probabilistic Model Checkers: A Survey

Model checking began in the mid-1980s, when it was shown that temporal properties could be verified automatically by exhaustively exploring a finite-state model [CE81; CG87]. Baier and Katoen subsequently systematized the theoretical foundations of model checking, including search algorithms, fixed-point logic, and complexity bounds [BK08]. The objective is clear: exhaustively explore every possible execution and, when a property is violated, provide a concise counterexample in time for engineers to address the issue.

Real systems, however, are rarely deterministic. Power grids shed load randomly, wireless links drop packets, and rescue robots work against survival curves that worsen over time. In order to analyze such uncertainty, model checking expanded to probabilistic models like discrete- and continuous-time Markov chains, together with temporal logics such as PCTL and CSL[Kon10; CT00; Kwi03]. Although these extensions face exponential state growth and numerical challenges, they yield risk estimates backed by mathematical proof.

Probabilistic Temporal Logic Specification

To specify quantitative requirements for stochastic systems, probabilistic model checking extends classical temporal logic with probabilistic and expectation operators. These extensions let us reason about probabilities of events, expected values of costs or times, and time-bounded behaviors over a mathematically defined probability space of paths.

- *Probabilistic Computation Tree Logic (PCTL)* augments Computation Tree Logic (CTL) with a probabilistic operator $P_{\bowtie p}[\psi]$, which asserts that the probability of the path formula ψ satisfies the bound $\bowtie p$ (with $\bowtie \in \{<, \leq, \geq, >\}$). Path formulae combine the usual temporal connectives such as next X , until U , eventually F , and always G [CG04]. Over DTMCs the semantics is direct; over MDPs, PCTL is interpreted under scheduler (adversary) resolution, yielding extremal values $\inf$ or $\sup$ over policies. For example,

$$P_{\leq 0.15}[\neg \text{fail}_A \ U \ (\text{fail}_B \wedge \neg \text{fail}_A)]$$

states that the probability of “ B fails before A ” is at most 0.15. Quantitative cost/time queries are supported via standard reward extensions (e.g. expected accumulated or long-run rewards).

- *Continuous Stochastic Logic (CSL)* targets continuous-time Markov chains (CTMCs) and refines temporal operators with real-time bounds. [CG04; Bai+03] The time-bounded until $P_{\bowtie p}[\Phi_1 \ U^{\leq t} \ \Phi_2]$ speaks about reaching Φ_2 within t time units while staying in Φ_1 in the meantime, and the steady-state operator $S_{\bowtie p}[\Phi]$ constrains long-run probabilities. Expected times or energy consumptions are again expressed through reward-augmented variants of CSL and computed by numerical CTMC analysis.
- *Probabilistic LTL (pLTL)* equips LTL with probabilistic quantification, typically asking whether the probability of all paths satisfying an LTL (Linear Tempo-

ral Logic) specification meets a bound (e.g. probability 1).[Kon10; Roy+25; BTT18] In general, the expressiveness of PCTL and pLTL is *incomparable*: each can state properties the other cannot. This makes the choice of logic a modelling decision, guided by the underlying stochastic model (DTMC/MDP vs. CTMC) and the nature of the requirement (branching-time vs. linear-time) [JM18]

The interpretation of these logics over probabilistic models reduces many verification tasks to solving linear equation systems and performing iterative numerical schemes (e.g. value iteration for MDPs, uniformisation for CTMC transients), optionally combined with reward analysis. This tight coupling between logical specification and numerical computation is what enables precise, time-bounded and quantitatively meaningful verification results.

Probabilistic Model Checking Tools

Over the last decade, quantitative model checking has coalesced around a small set of complementary toolchains (e.g., PRISM, Storm, the MoDeST Toolset/JANI), each pushing classical verification toward probabilistic, timed, and hybrid analyses [Kwi07].

Among these probabilistic models, PRISM and Storm define the current baseline. PRISM's uncertain models let users specify intervals for probabilities in DTMCs and MDPs and then perform robust verification -computing best-/worst-case probabilities over all selections within those bounds- rather than relying on sampling [KNP02]. Storm generalizes this picture with a highly modular back-end that handles DTMCs, CTMCs, MDPs, and Markov automata, and provides numerical engines (including robust value-iteration) plus broad language support via PRISM and JANI front-ends [Deh+17]. Together, these two tools deliver exhaustive guarantees for time-homogeneous stochastic dynamics and have become the default reference points in comparative evaluations.

When continuous dynamics enter the scene, UPPAAL-SMC takes the lead by shifting from exhaustive exploration to statistical model checking over (stochastic) hybrid automata [Bul+12]. Its semantics allow ODE-driven variables and hybrid mode switches while estimating probabilities and expected values from controlled simulations with confidence bounds - precisely the trade-off practitioners accept when exact exploration becomes intractable.

The MoDeST Toolset sits at the intersection of these worlds. Its MoDeST language provides a unifying front-end; *mcsta* performs explicit-state model checking and strategy synthesis for stochastic timed/parametric variants, while *modes* equips the same models (and full SHA) with statistical verification [Bud+20]. This makes MoDeST a promising tool for experiments that mix timed, probabilistic, and hybrid features without abandoning rigorous backends.

Although theoretical advances in probabilistic model checking have been significantly amplified by the development of robust and versatile software tools, here the question arises: Can present-day verifiers cope with such expressiveness? Among the most prominent are PRISM and the MoDeST Toolset, each offering unique capabilities for modeling, analysis, and verification of stochastic systems; in this chapter we adopt them as primary testbeds for IMCC verification and strategy exploration.

9.2 Challenges In IMCC Model Checking

Interactive Markov Chains with Clocks (IMCCs) represent a sophisticated formal model designed for the rigorous analysis of systems that exhibit a complex interplay of probabilistic, stochastic, and real-time behaviors. As previously introduced, our own modeling effort has merged Interactive Markov Chains (IMCs), Timed Automata (TA) and advanced Clock-dependent Probabilistic Timed Automata (Cd-PTA) in the sense of being able to capture stochastic incidents in real-world scenarios such as our rescue mission which includes synchronous robot–victim interaction, and reflect their dependency to time like the medically observed decay of survival probabilities. IMCC, therefore, lets rates evolve continuously as clocks tick so that a victim’s hazard rate accelerates the longer they are waiting to be rescued.

Modeling complex systems, particularly those that IMCCs are designed to represent, for formal verification presents formidable challenges due to their inherent characteristics:

- *Complexity and Heterogeneity*: Systems that IMCCs model often integrate a vast number of diverse agents/components, often developed by different vendors, leading to significant heterogeneity. This makes it difficult to create a unified, coherent model that captures all interactions and behaviors.
- *Stochasticity and Uncertainty*: Real-world systems operate in environments with inherent uncertainty, including sensor noise, unpredictable communication delays, component failures, and human interactions. These stochastic behaviors necessitate probabilistic modeling, which is directly addressed by

IMCC's probabilistic nature, but still adds a layer of complexity to traditional deterministic models.

- *Real-Time Constraints:* Many time-critical systems operate under strict real-time deadlines and dependencies, where timely execution and response are paramount in decision-making strategies and controller synthesis. Modeling and verifying these temporal properties accurately, especially in the presence of stochastic delays, is challenging, though directly supported by IMCC's clock variables.
- *Multi-agent and Game-Theoretic Analysis:* For systems involving multiple autonomous components or agents, probabilistic model checking can be extended to stochastic multi-player games, allowing analysis of competitive or cooperative behavior, and synthesis of optimal strategies or Nash equilibrium. This is critical for understanding system behavior when components act autonomously and concurrently.
- *State Explosion:* The high number of interacting components and variables across multiple domains leads to an exponential growth in the system's state space, making exhaustive verification computationally intractable for large-scale systems.

With these challenges in mind, and acknowledging that PRISM and the MoDeST Toolset each impose their own modeling languages and constraints, the remainder of this chapter presents how the IMCC model is encoded within both frameworks. We trace the complete translation workflow, noting the required assumptions and adjustments along the way.

9.3 Modeling Rescue Scenario via Multi-Agent IMCCs

In this section, we demonstrate the applicability and expressiveness of the IMCC model by encoding our motivation scenario, the rescue problem, based on the schematic model from Fig.4.3a and Fig.4.3b. This exercise provides a concrete example of how it can be used to formalize and analyze complex systems. By re-expressing the original model in terms of our framework, we can leverage the benefits of this new approach, including its improved clarity, flexibility, and rigor.

Through this reconstruction, we illustrate how the core concepts and expected behavior of the rescue mission can be naturally captured and extended within our framework. This involves mapping the original model's components and behaviors onto the corresponding elements of our framework, such as states, transitions, and

constraints. By doing so, the advantages of our new approach in terms of its ability to provide a more comprehensive and systematic understanding of the underlying system will be exhibited.

9.3.1 Agent Models In IMCC

Figures 9.1 and 9.2 show the model of two patients and one robot. The patient model from Fig. 9.1 is instantiated twice for two patients $i \in \{1, 2\}$, which are structurally identical but may have different death rates $\mathcal{F}_i$. Location *Idle* is left either by a rescue signal res_i from the robot or by spontaneous death. In the latter case, the patient model notifies the robot by an interaction d_i .

Figure 9.2 presents the model of the robot, which is a mere timed automaton, however with interactions to the patient models. From location *Idle*, it can decide for taking up any of the two patients. Being on the way then will be terminated either by death of the patient (via signal d_i) or by the time-of-way having elapsed.

For ease of presentation, we have simplified the model by not memorizing which patients still are alive, which would either require copies of the robot's locations or a querying signal between the robot and the respective patient, which the latter would only grant in location *idle*.

The robot makes a decision to execute the interactive transition way_i , thereby determining whether to proceed to patient 1 or patient 2. Upon selecting either of these transitions, the robot will be en route to the chosen patient. If informed about the death of this specific patient, the robot will take the interactive transition d_i and revert to the 'idle' state, thereby enabling it to make a decision regarding the next patient. Conversely, if the robot's clock equals t_{reach_i} , indicating its arrival at the patient's location, it will execute the interactive rescue mission transition ' rm_i '. Upon taking ' rm_i ' transition, the robot's clock is reset, and it enters a state of actively rescuing the designated patient. Notably, time continues to elapse in both the robot's and patients' models, introducing the possibility of the patient's demise or successful rescue. While either of these outcomes returns the robot to the 'idle' state, the ' res_i ' transition becomes available only after the rescue time has elapsed, at which point

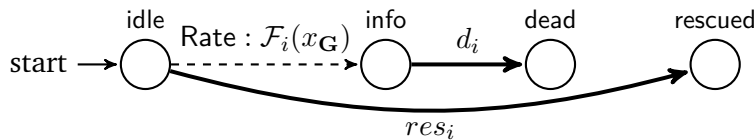


Fig. 9.1: Patients' IMCC model, to be instantiated for $i \in \{1, 2\}$.

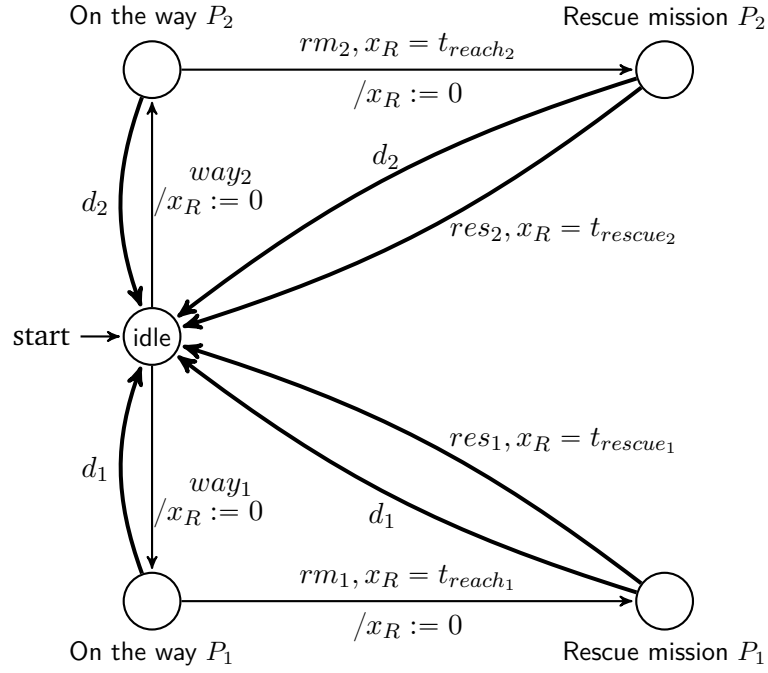


Fig. 9.2: Robot's IMCC model in a 2-Patient Rescue Mission

it resets the robot's clock. It is essential to distinguish that the ' res_i ' transition is robot-initiated, whereas the ' d_i ' interactive transition is triggered by patient.

Here, for clarification purpose only, we have visually distinguished the corresponding arrows of synchronized transitions between these three models by rendering them slightly thicker. It is intuitive that as the number of robots increases, the other interactive transitions will be synchronized across robot models to specify the patient assignment for each robot, thereby facilitating collaborative operations.

Figure 9.3, finally, shows an initial segment of the product IMCC equivalent to the concurrent composition of the two models above. It contains the states of robot and transitions that it can take. It is observed that the number of patients induces an exponential increase in the number of states of product automaton.

The greyed-out locations in the background refer to the symmetric alternative initial choice of moving towards patient 2 instead of patient 1. This choice deliberately is left non-deterministic, and the objective of strategy or policy construction then is to find a (potentially time- or clock-dependent) resolution of the non-deterministic choice(s) that maximizes expected reward. Here, the reward is the number of patients rescued. Its expectation can easily be inferred from the reach probabilities of the locations reflecting rescue of each of the patients. Thus, the model constitutes a $1\frac{1}{2}$ -player game, where the robot, or a set of them, as a collaborative group forms the strategic player.

Computing an optimal strategy in such rich game models inevitably is hard, especially when considering a multi-robot setting, inducing a need for a cooperative strategy letting multiple robots work together to rescue as many patients as possible. In this context, each robot must consider not only its own options but also the options and choices of other robots, and it may have for optimality to revise the own plan even when only another robot's plan is directly hit by the sudden death of a patient about to be served.

9.3.2 Product Model In IMCC

The figure (Fig. 9.3) demonstrates a schematic of possible traces in the product model. To facilitate understanding and visualization, we provide a simplified abstraction of the system, highlighting a selection of few states and transitions. In reality, the product model generates a finite, yet potentially large, number of states at any given time instance, which are derived from the time-dependent probabilities assigned to each patient. According to previous Robot and Patient IMCC models, we denoted set of states 'I', 'W' and 'R' associated with states 'Idle, On the way to P_i , Rescue Mission P_i ' for robot, respectively. Similarly, states that patients can occupy are depicted as 'I', 'inf', 'D', 'R' following notions of states 'Idle, Info, Dead and Rescued'.

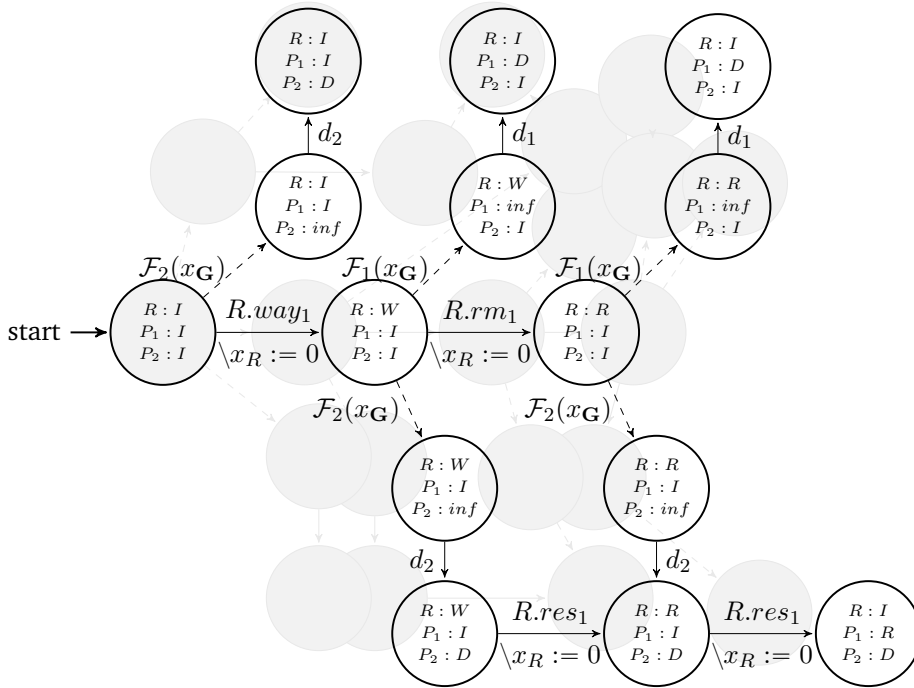


Fig. 9.3: Rescue mission for one robot and two patients, IMCC model

Figure 9.3 employs a horizontal representation of time progression. Here, vertical transitions do not involve in time passage and are exclusively associated with information regarding patient's death since 'info' states are urgent and there will be

no time difference between states 'info' and 'dead' in patient's model. As illustrated in the diagram, the rate of an autonomous transition being taken is equivalent to the sum of rates of all possible autonomous transitions leading to the successor state. When an interactive transition is initiated by the robot, global time continues to pass, but the local clock of the robot is reset. Noting that interactive transitions do not take time, the horizontal axis shows the time difference (due to clock-dependent guards and rates) between states. Additionally, the gray circles and arrows in the background serve to illustrate the presence of other, finitely many, possible states and transitions at any given time point.

9.4 IMCC Encodings in PRISM

PRISM is a symbolic probabilistic model checker for a broad class of stochastic models, including discrete-time Markov chains (DTMCs), continuous-time Markov chains (CTMCs), Markov decision processes (MDPs), and (priced) probabilistic timed automata (PTAs). Specifications are written in logics such as PCTL and CSL, with support for reward/cost operators. Symbolic state-space construction (BDDs/MTBDDs) and numerical engines (sparse, symbolic, hybrid) enable quantitative verification and exact computation of transient/steady-state probabilities and expected rewards, complementing simulation-based methods [KNP06].

A structural limitation of PRISM is its single-controller view: nondeterministic choices represent one entity whose decisions are resolved by a single adversary during verification. This abstraction suffices for many reliability questions but cannot capture scenarios where multiple agents pursue possibly conflicting objectives: for example, attacker vs defense mechanisms or cooperative teams under partial trust. The need to reason about such strategic interaction, and to synthesize policies that are provably optimal against rational opponents, motivated the development of a dedicated extension.

PRISM-games addresses this by lifting PRISM's engines to *stochastic multi-player games* (turn-based or concurrent) and by supporting rPATL, a quantitative extension of ATL for coalitional objectives (probability and reward bounds), together with automated strategy synthesis. This makes it possible to reason about strategic ability and to extract witness strategies for cooperating or competing coalitions [KNP11].

Compared to core PRISM, PRISM-games therefore adds explicit multi-agent semantics, strategic operators and automated policy generation. These features allow analysts to ask not only *whether* a requirement can be met but also *which* strategy achieves it and *how* opposing behavior affects success probabilities, such capabilities that are absent from PRISM's MDP-based analysis. The trade-off is higher computa-

tional cost and a focus on discrete-time game models; continuous-time constructs such as CTMCs and general Markov automata remain outside its present scope. Nevertheless, by extending PRISM, PRISM-games inherits the mature symbolic infrastructure of its parent while charting the game-theoretic frontier of probabilistic model checking [Che+13a].

PRISM and its game-theoretic companion convey a synthesized strategy that rely on a policy extracted internally by value- or policy-iteration algorithms. For common reachability and many reward objectives, optimal memoryless deterministic policies suffice; more complex objectives may require randomized or finite-memory strategies. Exported strategies annotate controlled states with action choices/probabilities for inspection or deployment.

PRISM-games inherits this mechanism and, for stochastic multi-player games, automatically tags each decision with the owning player; then the strategy listing enumerates each controlled state together with the action probabilities. It can also export finite-memory or stochastic-update strategies when those are required by multi-player objectives.

9.4.1 IMCC's Rescue Scenario as a Multi-player Game

The rescue scenario (and disaster incident in general) involves several agents that act simultaneously. The clinical states of the victims, the robots' rescue actions, and the evolving environment all progress in parallel. Although victims perform no deliberate actions, their changing conditions directly influence the robots' decisions. As injuries worsen or hazards intensify, the robot must recalculate its priority metric, here defined as maximizing the number of survivors, at each instant. These changes occasionally force the robot to adjust its strategy while the mission is under way, and the scale of uncertainty grows exponentially once several robots enter the scene.

Viewed through the lens of game theory, this interaction constitutes a 1.5-player stochastic game. All robots that actively choose their manoeuvres form the single *full player*: they select actions so as to optimize the rescue objective under real-time constraints. The opposing *half-player* is nature, embodied by the environment, the victims' physiological decay, and any form of hazard. Nature introduces nondeterminism and stochastic transitions but makes no strategic choices of its own. Hence the robots face an adversary that is probabilistic rather than rational, making Interactive Markov Chains with Clocks (IMCCs) a suitable semantic framework for capturing both deliberate control and random evolution within a unified model.

Modeling such a game in PRISM is feasible yet not without limitations. The tool natively supports Markov decision processes and turn-based stochastic games, but constraints on expressions, clock usage, and module instantiation require careful abstraction. Time-varying survival curves must be discretized, interactions among multiple robots have to be encoded explicitly, and strategy synthesis is limited to memoryless or finitely-memory policies. The following sections detail each of these restrictions and the work-arounds adopted in our implementation.

9.4.2 Limitations & Considerations

In the PRISM software, this concurrency of our model can only be represented in PRISM Games through the Concurrent Stochastic Games (CSG) semantics. We have formulated our scenario by separate modules for each victim, each robot, and the environment, all synchronized on a common time step defined as a separate module. The robot module observes the state variables published by the other modules, updates its local preference list, and makes a non-deterministic choice of which victim to approach. This construction preserves the inherent parallelism of disaster response while making the assumptions and simplifications required for formal verification explicit.

A key limitation of both PRISM and PRISM-games is that a probability or rate cannot be expressed as an arbitrary real-valued function of time. Even in the real-time setting, transition guards may only compare clocks with constants or with one another; they cannot reference a general time-dependent hazard or survival curve. Consequently, the robot’s decision procedure cannot query “the exact value of a victim’s survival probability at the current time or at some time point in the future” unless that function has first been discretized.

To bridge this gap, we introduced an explicit time-step and pre-computed, piece-wise-constant survival functions offline. For every casualty, a random survival curve was generated in MATLAB; the curve was then sampled at the chosen time resolution, producing a table of survival values indexed by the integer step $k = 0, 1, 2, \dots$. These tables were exported as constant values per time points and embedded in the PRISM model as read-only global variables. Each time, the dedicated clock module increments the step counter, the robot module reads the corresponding entry of the array and updates its prioritization logic accordingly. This discretizations preserves the qualitative trend of the continuous function while keeping the model within PRISM’s syntactic constraints and the state-space small enough for verification (Only for small number of agents)

An additional modeling obstacle in PRISM game is the absence of true module instantiation: every agent or player must be declared as a separate module, and any formula that involves pairwise interactions must be re-written explicitly for each pair. Consequently, the specification grows quadratically with the number of agents, and maintaining consistency across duplicated equations becomes error-prone.

To mitigate this redundancy, we developed a MATLAB script generator that automatically rewrites the base model for both single- and multi-agent scenarios. Given the desired number of players, the script clones the template module, renames variables and synchronization labels, and regenerates all interaction formulas, producing a complete PRISM model that scales linearly in authoring effort even as the underlying system scales in size. In what follows, we present in detail the modeling considerations adopted for both single-agent and multi-agent configurations.

9.4.3 Single Agent Model in Disaster Responses

This section introduces the baseline configuration in which a single autonomous robot operates within the disaster scene. The element that evolve without deliberate control, the victims' health state, is modeled as stochastic processes. Against this environmental background, the robot constitutes the sole decision-making entity. Its objectives, observations, and actuation capabilities are encapsulated in two modules, while other modules describe each victim and time progress. Since only one agent chooses actions, the resulting system's state space combines the robot's location and internal clock with the discrete health states of every victim. We explain how the priority function *Cumulative survival probability* is encoded, how time is discretized to satisfy PRISM's syntactic constraints, and which assumptions namely, full observability, instantaneous treatment, and travel time, are imposed to keep verification tractable.

Time Module

In the time module, we have two considerations: either time advances incrementally by the defined time step (here the time step is 1), or it reaches the maximum time limit defined in the assumptions and stops. The synchronization available in PRISM games allows to inform robot module about this restriction when it's over and the robot module terminates robot's mission accordingly. (Fig. 9.4)

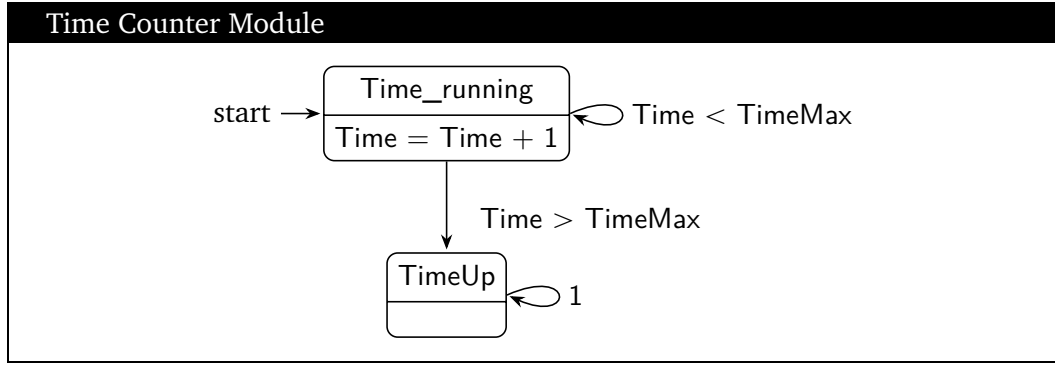


Fig. 9.4: Time progress module in PRISM.

Decision Module

The decision-making module is arguably the most intricate component of modeling IMCC in PRISM games and accordingly demands the greatest compromise in both the single-agent and multi-agent configurations.

In the single-robot scenario, while the robot has not yet committed to rescuing any victim, the PRISM model is nevertheless required to choose one; we have formulated in the form that it favors the victim with the highest expected probability of survival. (Fig. 9.5) To operationalize this choice, we assign to every still-viable victim a priority probability. This probability is computed from (i) the robot's current distance to the victim, (ii) the victim's projected survival likelihood, obtained from its time-dependent survival curve at the moment the robot is expected to arrive, and (iii) the time needed to complete the rescue once contact is established.

Because each of these probabilities may assume any value in the interval $[0, 1]$ and is not intrinsically linked to the probabilities assigned to other victims, their sum can be an arbitrary positive number. PRISM, however, requires that the outgoing transition probabilities of any state sum to one. To satisfy this requirement we introduce a normalization factor, hereafter termed the *fix coefficient*, that scales the individual survival probabilities so that their aggregate equals unity.

However, this is only one of the considerations associated with the decision-making layer. The main limitation that PRISM games cannot capture is the effect of different sequences of decisions on current decision. While the cumulative survival probability can, in principle, be expressed analytically and imported into the model as a reward structure (an approach we have indeed adopted), PRISM Games does not allow that reward to be incorporated into the robot's *immediate* transition logic.

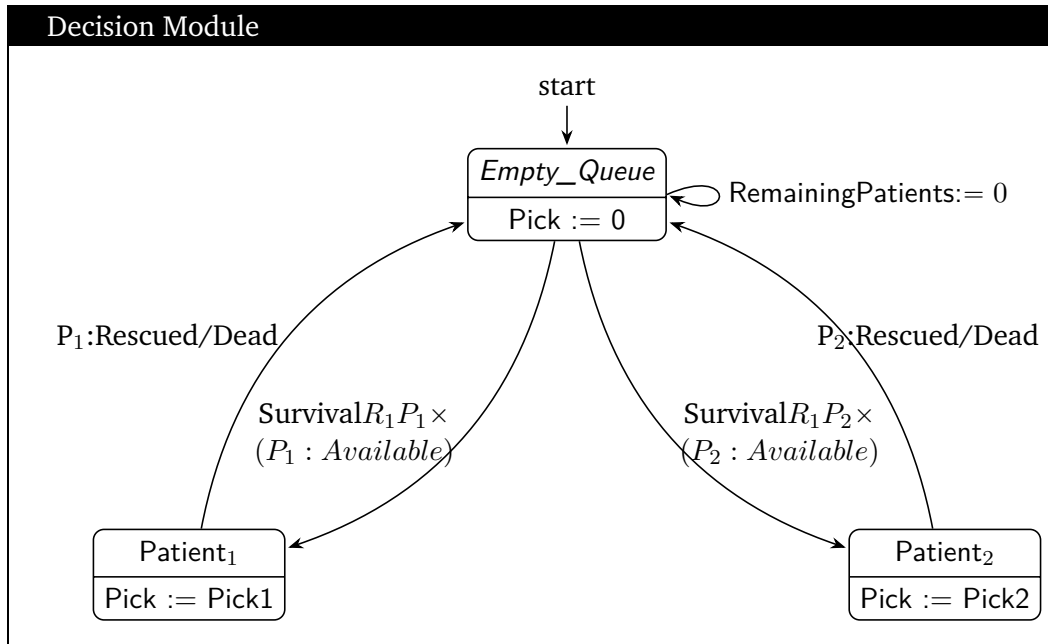


Fig. 9.5: Transition diagram for the single-robot, two-patient decision process in PRISM.

Consequently, the probability attached to any transition (and thus to the successive state) is computed exclusively from data available at the current time step; it cannot factor in how rescuing the present victim will affect the survival prospects of the *remaining* victims in subsequent time steps. In short, future-dependent utility cannot be folded back into the robot’s moment-to-moment policy within the standard PRISM framework.

Robot Module

The robot module is organized around three mutually exclusive phases. First, the robot is in the decision–exploration phase, during which it scans the environment and selects a victim to rescue. Second, if the global time horizon elapses or no available patient is remained, the robot immediately enters an idle phase and performs no further actions. Third, once a victim has been chosen, the robot switches to the rescue phase, which itself comprises two sequential steps: the robot travels toward the victim, and upon successful arrival before the victim’s death, administers treatment.

Here we encounter a further limitation of PRISM. Although it is straightforward to encode an expression that yields the total travel time from the robot’s current position to the victim’s location, PRISM cannot incrementally update a continuous state variable such as the robot’s coordinates at each time step. Consequently, if the

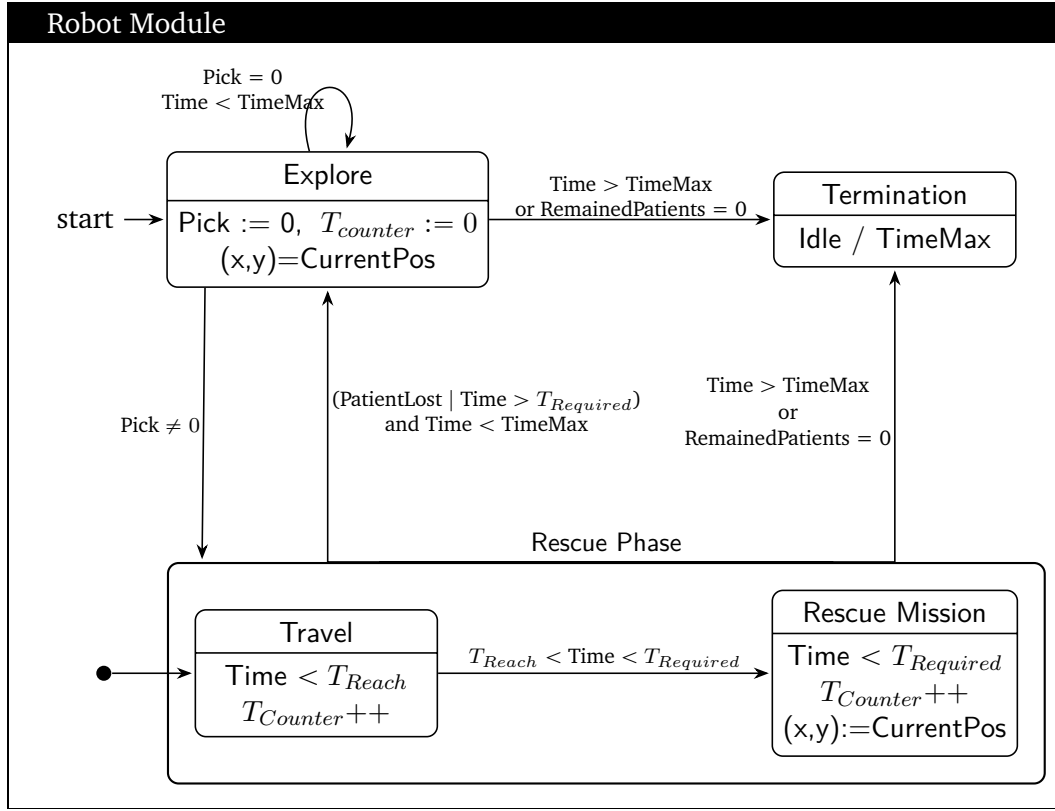


Fig. 9.6: State-transition diagram for a single robot in PRISM.

victim dies before the robot reaches them, the model cannot compute the partial distance of “how far the robot actually traveled”, thus, the robot’s position remains frozen at its previous coordinates. This approximation propagates error into all subsequent decisions that degrades plan optimality in later stages. When the robot does reach a living patient, we explicitly reset its position to the patient’s node, partially mitigating this drift. Once treatment is complete, the module returns to the exploration phase to select the next target.

Patient Module

In the patient module, each victim is modeled by an independent automaton, because PRISM does not allow a single template to be instantiated and reused. Conceptually, a patient can be in one of three clinical conditions: *injured but alive*, *rescued*, or *dead*. However, in order to distinguish whether a robot has already committed to this patient, we specify the current action in the synchronization part, e.g. *bleeding*.

As stated in time module, time advances in discrete steps. At every tick, the patient’s fate is governed by a survival-probability curve that has been pre-discretized. If the current survival likelihood is p , the automaton remains in *Alive* with probability p and

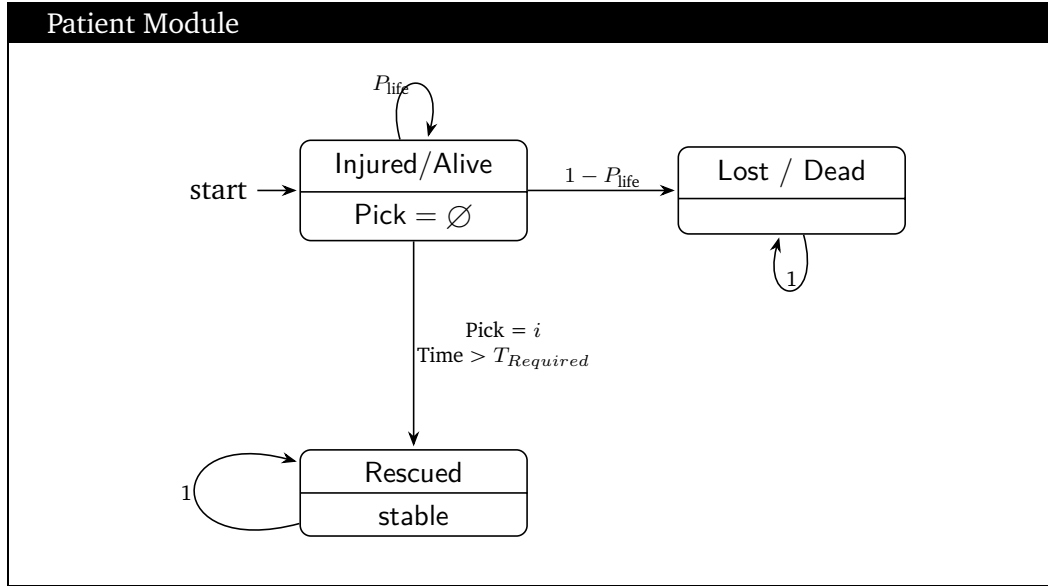


Fig. 9.7: State-transitions for patients in PRISM

moves to *Dead* with probability $1 - p$. Should a robot assign itself before the patient expires, the same probabilistic decay continues to apply until the robot finishes the rescue procedure, at which point the state changes irrevocably to *Rescued*. Thus, even after allocation, the patient still faces a non-zero chance of dying at each intermediate time step, faithfully reflecting the clinical uncertainty that the survival curve encodes.

This structure captures the key clinical reality: a numerical survival curve does not guarantee that “sufficient time remains”. A present moment survival probability of 0.8 merely states that, right now, the patient has a 20% chance of dying before the next tick. Incorporating that fact into PRISM requires nothing more than this simple three-state automaton, updated at each step by the discretized values drawn from the curve (Fig. 9.7).

9.4.4 Multi-Agent Model in Disaster Responses

The multi-agent configuration extends the baseline by introducing autonomous robots that act concurrently. Each robot maintains its own clocks, local perception, and action module, yet all share the same global objective of saving as many victims as possible. When several robots compete for limited access routes or for the same victim, their choices interact, producing a stochastic game with decision-making players and one probabilistic *nature* component.

In order to capture these interactions in PRISM, we generate copies of the template robot module and couple them through synchronization labels that encode patient

assignment and rescue sequences. Priority updates rely on the discretized survival curves introduced earlier, but now each robot computes its preference order independently, leading to potential conflicts that the verification engine must resolve. The following details scale of the model, the additional fairness constraints required to prevent deadlock, and the limits encountered when synthesizing joint strategies under PRISM’s memoryless-policy assumption.

The component most profoundly affected by the presence of multiple robots is the decision-making module. In the multi-agent model, every robot is equipped with its own private instance of that module, yet the modules cannot be allowed to issue concurrent, potentially conflicting choices. If two robots were to select the same victim at the same instant, the system would have to compare their bids, adjudicate the winner, and force the losers to reconsider; an interaction that is tractable for a pair of agents but causes the state-space explosion once the team grows or when all robots start from identical positions.

Random Priority Module

Our approach to avoid this state-space blow-up is to introduce an auxiliary *random-priority* module. Whenever more than one robot enters the decision state simultaneously, this module assigns the right to choose in a turn-based manner, selecting the next robot uniformly at random. The elected robot finalizes its target first, after which the remaining robots, still queued in the priority scheduler, take their turns in sequence. A robot that finishes its current mission, re-enters the random-priority module; if every other robot is busy, it bypasses the queue and proceeds directly to its own decision module.

The local optimization performed during each turn is identical to the single-robot case aiming to maximize the arrival-time survival probability computed from the robot’s current distance to every viable patient. Also, the global constraint that a patient may be allocated to at most one robot is preserved. Because PRISM cannot update robot positions continuously, more sophisticated consensus mechanisms were not implementable, but the random-priority scheduler proved sufficient to eliminate deadlocks and keep the state-space manageable.

9.4.5 Relations and Formulas

Beyond individual robots, patients, decision-making, and timing modules, the model had to capture the way those modules interact. In PRISM, these interactions cannot be expressed graphically and therefore had to be written as explicit algebraic rela-

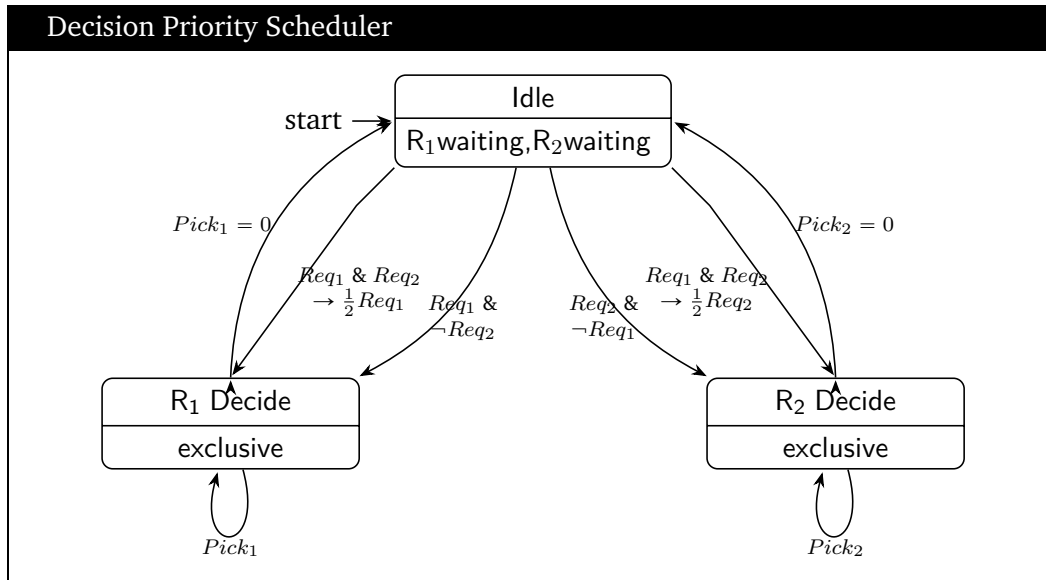


Fig. 9.8: Random priority assignment as decision-making conflict resolution in PRISM.

tions. Each relation comes with its own modeling constraints and implementation tricks; the most important ones are:

- *Each-robot-one-patient* At every tick, a robot's assignment variable should hold at most one patient ID. This rule guarantees that no two patients are simultaneously assigned to the same robot and is enforced in PRISM by mutually exclusive guards on the robot's transitions.
- *Robot's local time counter* As soon as a robot starts moving toward a patient, its internal clock is reset to zero and increments once per step. The timer is compared with the pre-computed travel-plus-treatment duration: reaching the travel threshold triggers the transition from Rescue to Treatment and updates robot's coordination; reaching the combined threshold triggers a return to Explore, where the timer is reset.
- *Patient survival probability from robot's perspective* The baseline survival curve of a patient is adjusted for every robot-patient pair by taking the robot's current distance (and hence travel time) into account. The longer that travel time, the lower the effective survival probability at the robot's expected arrival time, and this adjusted value is what drives patient assignment.
- *Cumulative survival probability* The sum of survival probabilities over all patients is used as the model's reward to be maximized. While the value depends initially on which robot is assigned to which patient, once a patient is success-

fully rescued, its individual probability is set to 1 -regardless of which robot performed the rescue- and the global reward is updated immediately.

9.4.6 Implementation Results

Having established our PRISM(-games) encoding of the rescue scenario, we now present outputs driven by the properties we specified, first for only one robot followed by a multi-robot setting. Concretely, with robot (R_1) as the sole strategic player and a global time budget (T_{max}), we query the maximum expected number of patients rescued before (T_{max}) as :

$$<< R1 >> Pmax = ?[F <= T_{max} \text{ SurvivedPatients} >= n]$$

Consequently, the same query has been made considering both robot as strategic players in the multi-robot setting to mutually maximize the reward:

$$<< R1, R2 >> Pmax = ?[F <= T_{max} \text{ SurvivedPatients} >= n]$$

In PRISM-games, this is framed as a reward-maximization problem on the CSG model: each successful rescue contributes one unit to a cumulative reward, while time elapse is captured by the model's timing structure. We evaluate the maximum expected value along with the induced strategy as a function of the robot's speed (V_R), reporting the curve ($V_R \mapsto n[\text{rescued}]$) to show how increased mobility shifts the attainable optimum under our stochastic survival processes.

Even under the limitations and simplifications outlined above, and after the encoding choices needed to fit PRISM's syntax, our baseline instance (one robot, four patients at fixed locations, survival curves discretized with time step ($\Delta t = 1$)) still yields an interpretable trend: as we increase the robot's speed v , the probability of rescuing the patients rises monotonically (Fig. 9.9 and Fig. 9.10). This is the expected effect of shrinking travel times in a time-critical scenario with decaying survival, and it holds across the admissible parameter range of our encoding.

On the strategy side, PRISM Games does not synthesize a full optimal policy as strategy synthesis is currently only feasible for zero-sum concurrent games; instead, we present a representative interactive trace of the robot's behavior detailing target selection, travel, rescues, and aborts upon death signals. Each possible transition indicates a certain probabilistic branching dependent on the current distance from robot to each patient and their survival probability expectation. This makes the decision flow explicit despite the abstraction choices (Fig. 9.11 and Fig. 9.14).

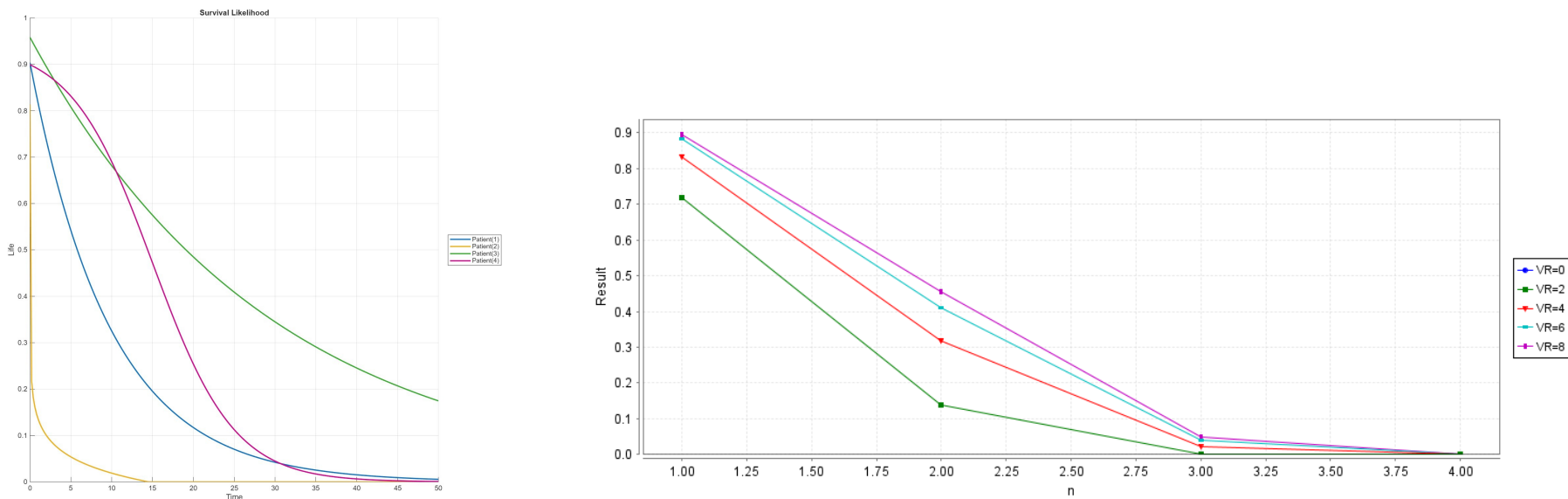


Fig. 9.9: PRISM synthesis for a mission with one robot and four patients

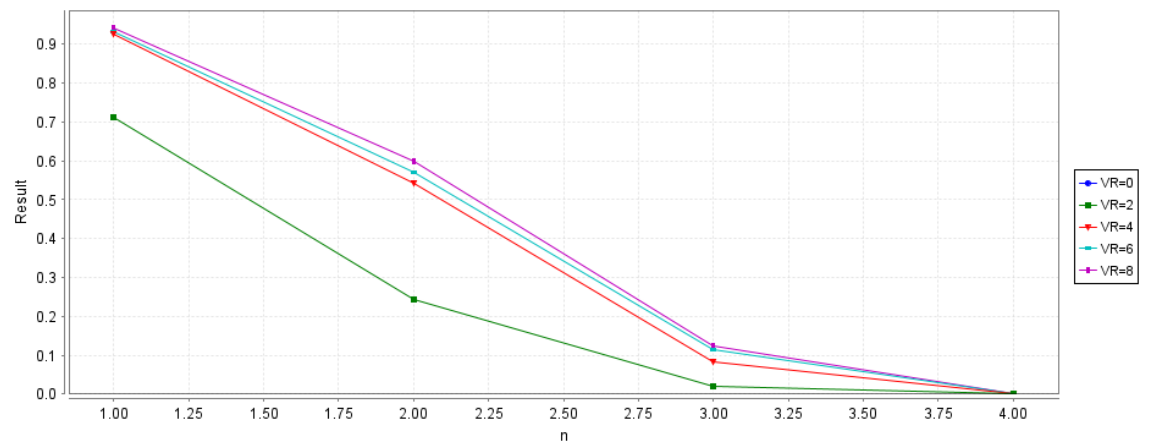
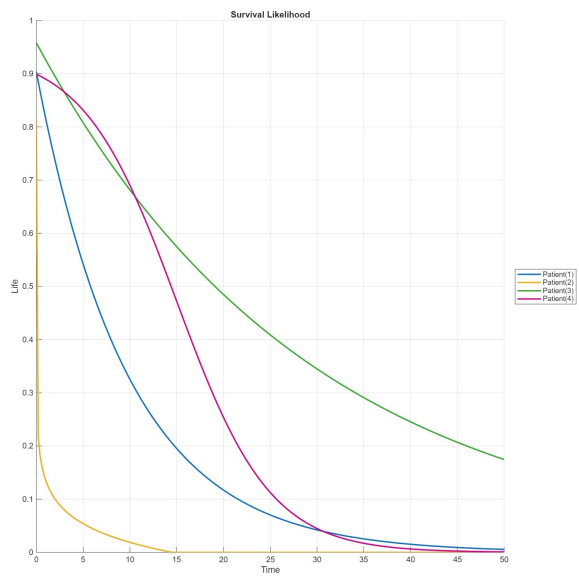


Fig. 9.10: PRISM synthesis for a mission with two robots and four patients

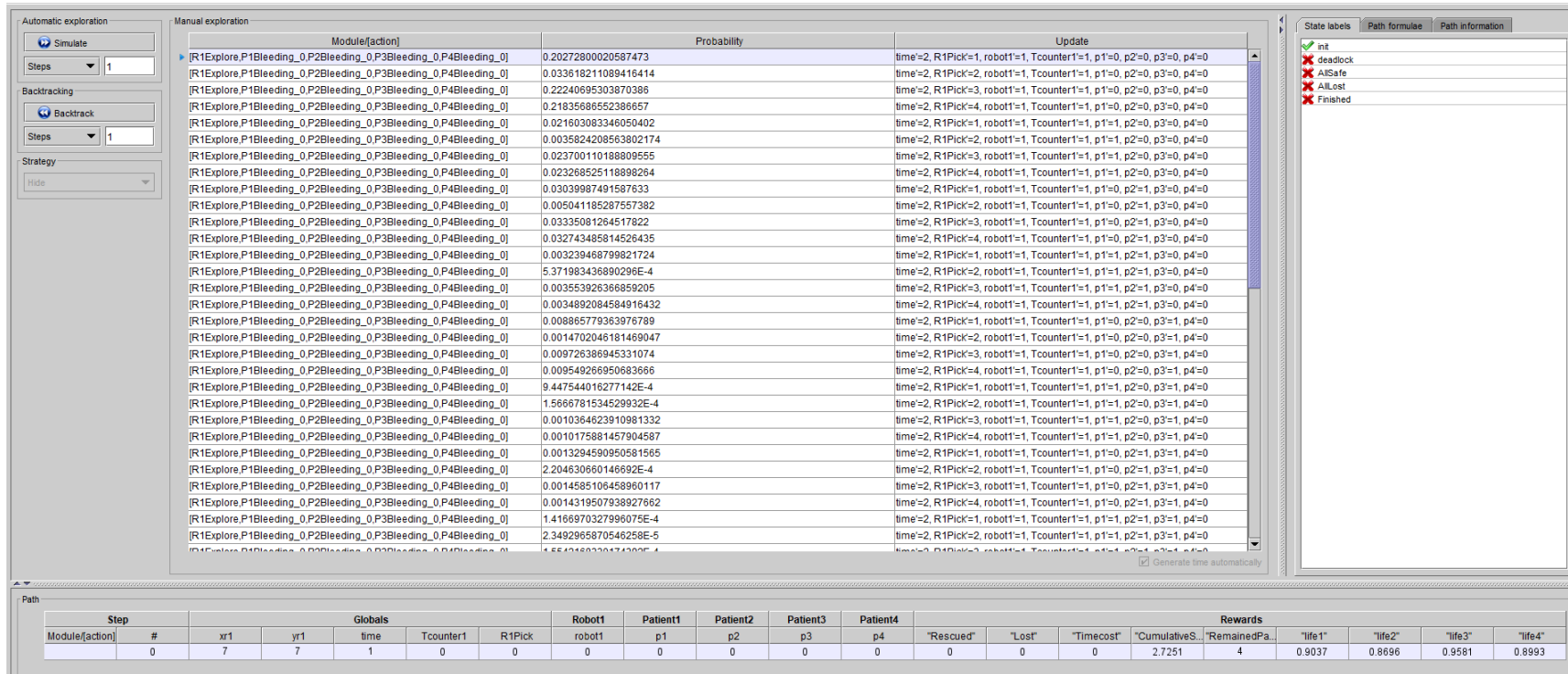


Fig. 9.11: PRISM interactive simulation (Trace 1) — 1 Robot, 4 Patients

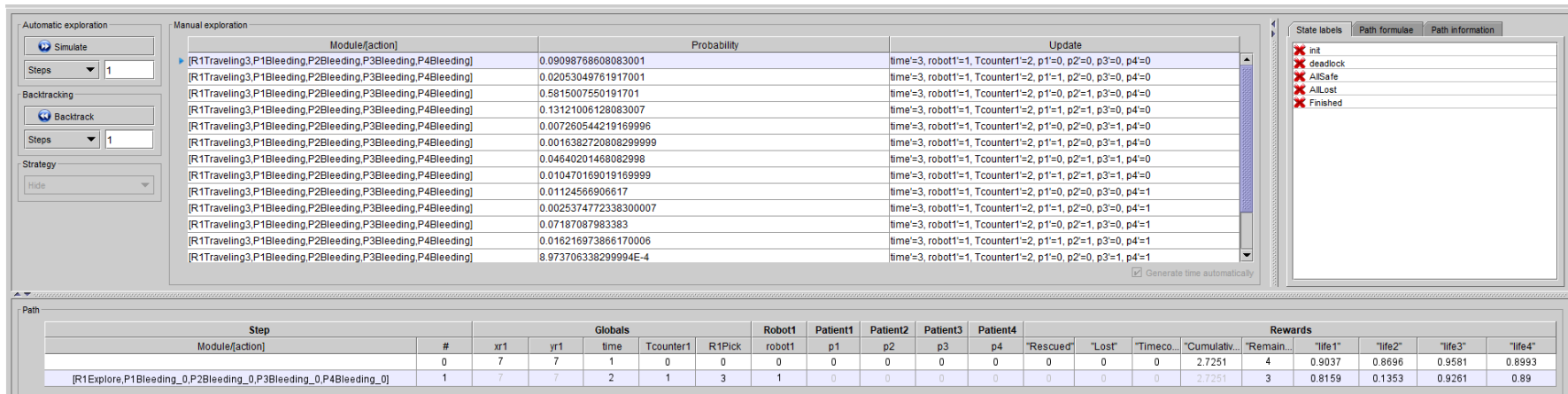


Fig. 9.12: PRISM interactive simulation (Trace 1) — continuation

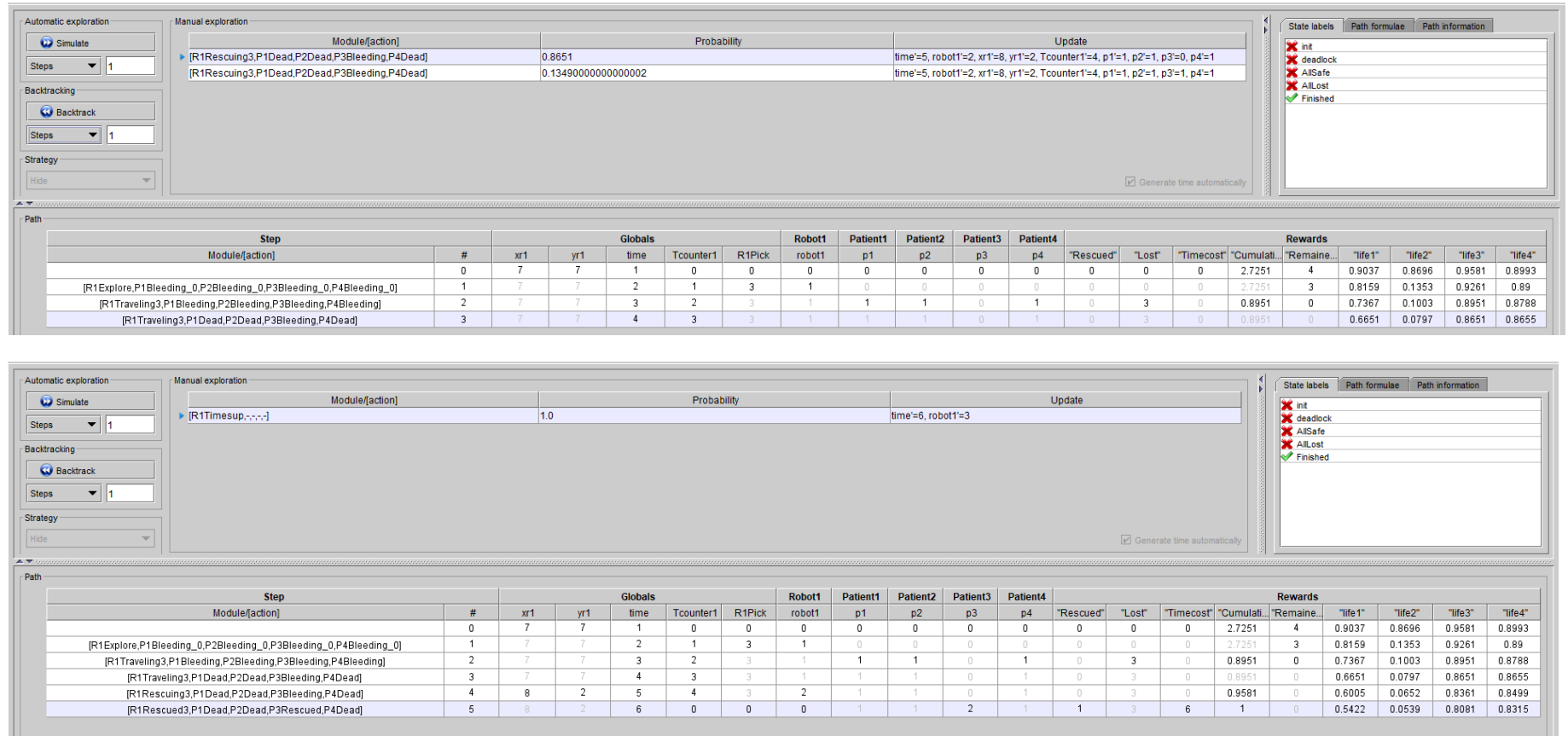


Fig. 9.13: PRISM interactive simulation (Trace 1) — continuation

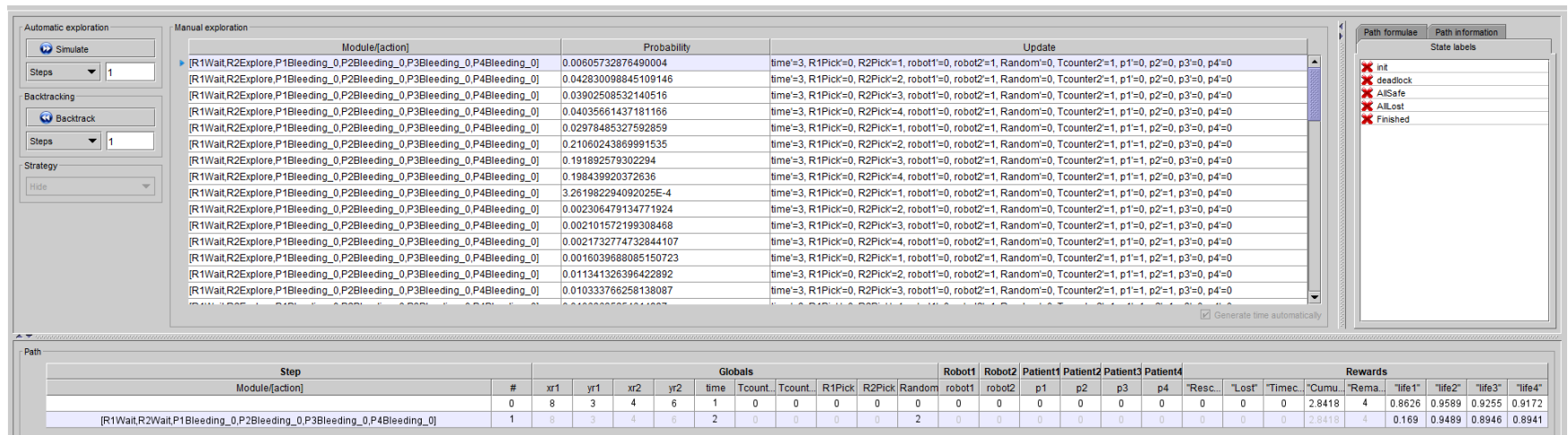


Fig. 9.14: PRISM interactive simulation (Trace 2) — 2 Robot, 4 Patients

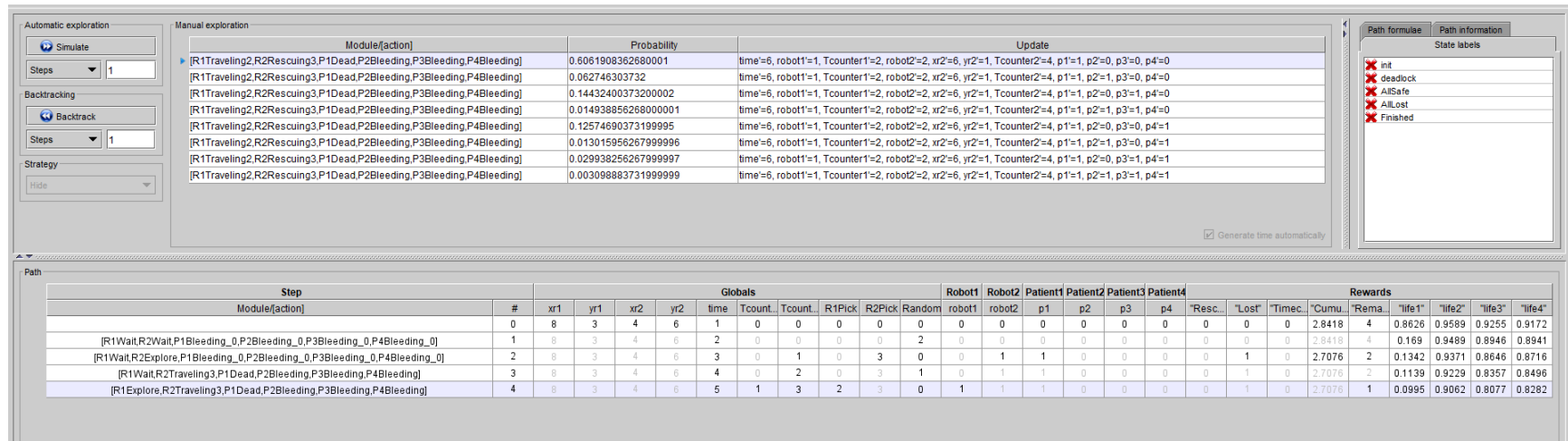


Fig. 9.15: PRISM interactive simulation (Trace 2) — continuation

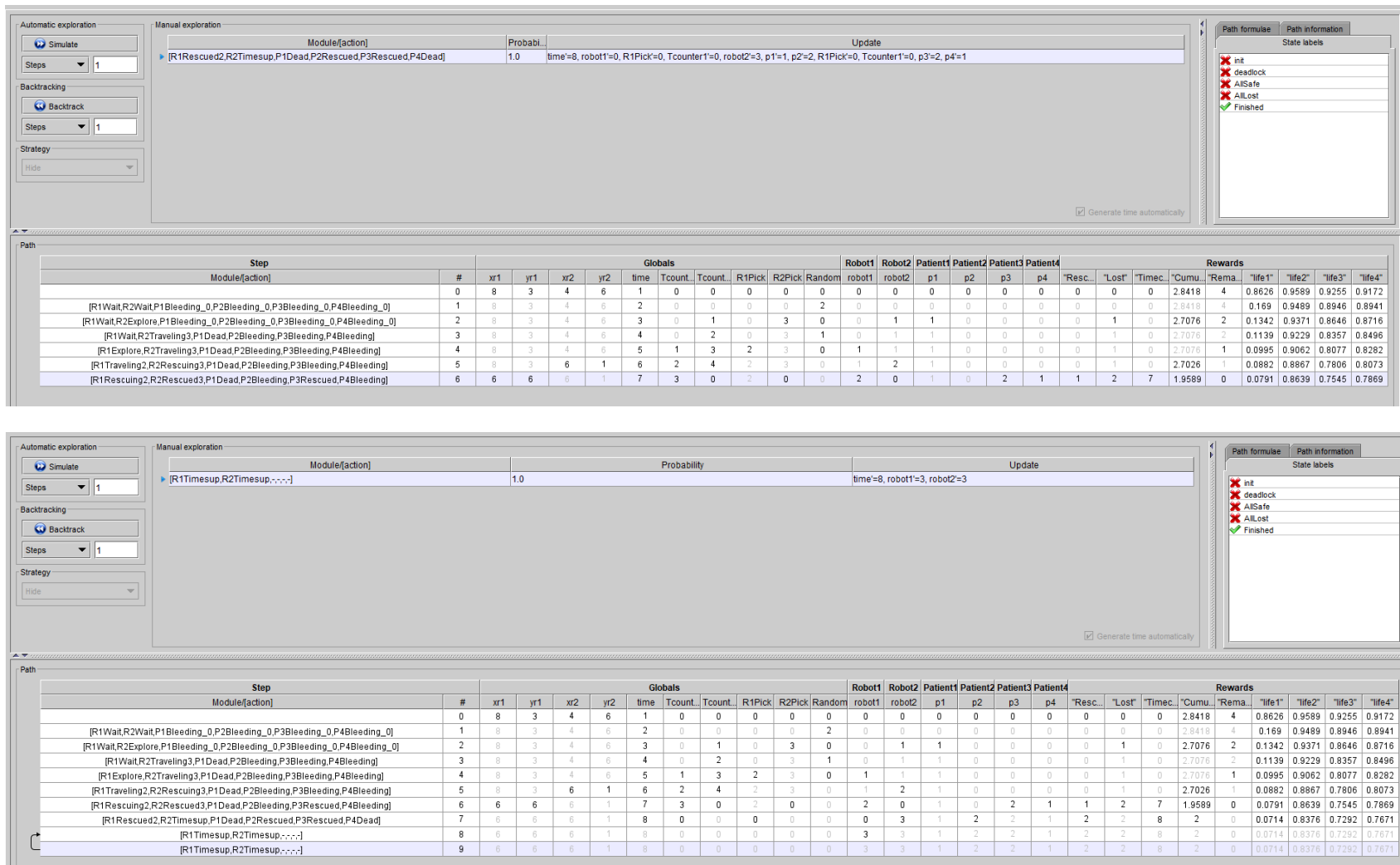


Fig. 9.16: PRISM interactive simulation (Trace 2) — continuation

9.5 IMCC Encodings in MoDeST

The MoDeST Toolset is an integrated environment for the quantitative modeling and verification of stochastic timed systems. At its core lies the MoDeST language, a high-level compositional language rooted in process algebra, designed to capture probabilistic, real-time, and nondeterministic aspects within a single formalism [DAr+01; Boh+06].

MoDeST’s underlying semantics covers mathematical models ranging from LTS and DTMC to stochastic hybrid automata (SHAs), supporting deterministic, non-deterministic, and stochastic choices and delays based on arbitrary discrete and continuous distributions, which makes it particularly adept at modeling complex cyber-physical systems [Har15]. Moreover, its language support for compositional operators allows breaking down complex systems into smaller, more manageable components. MoDeST’s ability to model non-determinism directly is beneficial when dealing with components whose internal behavior is not fully specified (black-boxes) or when employing abstraction techniques to simplify the model [Bud+24].

In terms of capabilities, the toolset supports modeling and verification of quantitative properties such as performance and dependability through networks of automata family [HH15a]. It offers discrete-event simulation for both deterministic and non-deterministic models, and includes a statistical model checker that can efficiently estimate probabilities of rare events and approximate optimal schedulers in nondeterministic settings [Bud+20]. A distinctive strength of the toolset is its “single-model, multiple-solution” approach: the same model can be subjected to discrete-event simulation, statistical model checking, or more traditional exhaustive model checking, depending on the properties of interest [Har12]. Thanks to its modular architecture, the MoDeST Toolset also connects with external verification backends such as CADP and Uppaal, while the JANI exchange format enables interoperability with a variety of other input formalisms [Boh+06].

The toolset has been applied in diverse case studies, ranging from power supply noise in network-on-chip systems, to message routing in satellite constellations [HH19; Har22]. These applications demonstrate its flexibility in handling both hybrid dynamics and large stochastic models. At the same time, challenges remain: rare events can still cause excessive runtimes despite the efficiency of statistical model checking, and nondeterminism cannot be fully resolved by simulation-based approaches alone, though lightweight scheduler sampling provides partial mitigation [BHH21].

Taken together, the MoDeST Toolset offers a comprehensive and flexible environment for quantitative verification. While limitations such as state-space explosion and

nondeterminism remain active research challenges, its ability to integrate simulation, model checking, and statistical analysis within a unified framework, combined with its successful use in real case studies, makes it one of the most advanced environments available for analyzing stochastic timed systems.

9.5.1 IMCC's Rescue Scenario as Stochastic Hybrid Automata

Similar to PRISM encoding, the MoDeST Toolset required us to adapt the semantics of our model to the formalisms currently supported by the language. Since IMCCs semantics are not yet available for implementation directly from the tools, What we can do is to build an STA/SHA abstraction with coarse time buckets and static assignments to obtain upper and lower probability bounds, and complement it with statistical sampling to explore candidate heuristics. This choice was not arbitrary but dictated by specific limitations and requirements of the tool.

9.5.2 Limitations & Considerations

MoDeST, in its current implementation, provides strategy synthesis for stochastic timed automata (STA). However, in an STA clocks cannot be observed directly: one may assign values to them, constrain transitions through equalities or inequalities, and reset them when conditions are satisfied, but the model cannot continuously “read” the current value of a clock. In our case, the robot’s control logic depends on monitoring its local time, since strategy decisions are explicitly governed by how much time has elapsed. In addition, once an urgent interactive transition is taken, robot has to update its location according to the current local time. Without the ability to observe and use the clock value as a variable in equations, the STA formalism cannot capture the decision-making dynamics at the heart of our model.

To overcome this, we defined robot’s clock as continuous variables with rate 1 and embedded them into continuous dynamics, effectively lifting the model from an STA into an SHA. This allows the robot to use its local time directly within its strategy formulation. The drawback is that MoDeST currently cannot generate strategies for SHA models: it can only provide execution traces under simulation.

A second limitation concerns time-dependent rates. In the intended IMCC semantics, the hazard rate of each victim evolves with time, reflecting the accelerating risk of death as treatment is delayed. MoDeST, however, only supports a number of distributions with fixed rates, from which it samples the possible outcome. Thus, in our encoding, each victim was modeled with a constant exponential death rate, which restricts the realism of the clinical dynamics.

Eventually, in order to preserve observability for the robot, we treated the patient's state as partially observable, which is a recent feature of MoDeST: the robot can detect coarse-grained clinical states but does not have access to the full internal progression. This compromise is in line with what can reasonably be perceived in an actual rescue mission, while still keeping the model compatible with the tool's capabilities.

Further limitations and assumptions emerge when a large-scale, multi-robot rescue scenario is to be encoded in MoDeST which are explained in details where the implementation approach is introduced. These hurdles include Perfect-information assumption, Lack of rate racing, clock-dependent continuous functions, Limited reward optimization, Taken together, these limitations mean that an IMCC model faithful enough to reflect traveling robots, dynamic victim status, and decentralized decision making cannot be truly synthesized end-to-end in MoDeST.

9.5.3 Single Agent Model in Disaster Responses

The same rescue scenario that we previously encoded in PRISM was also modeled in the MoDeST Toolset, with adjustments to reflect the different semantics and capabilities of the language. As before, victims are represented as stochastic processes whose health deteriorates over time, while the robot acts as the single decision-making entity. The optimization objective is now formulated in terms of a reward structure that maximizes the total number of survivors, expressed as an integer count rather than as cumulative survival probability.

Synchronization between transitions remains an essential mechanism, ensuring that robot actions and victim processes evolve consistently. Unlike in the PRISM encoding, MoDeST allows us to incorporate partial observability: the robot does not have access to the full internal state of each victim but instead perceives only coarse-grained health categories, a feature that adds realism and reflects actual rescue conditions.

This baseline configuration first examines the system with a single robot, after which the same modeling approach is extended to groups of robots, allowing us to assess how coordination and concurrency affect the overall rescue outcome.

Patient Process

For patient process, similar to PRISM, instantiation among a number of victims is not possible. Therefore, each victim is modeled by an individual process. The

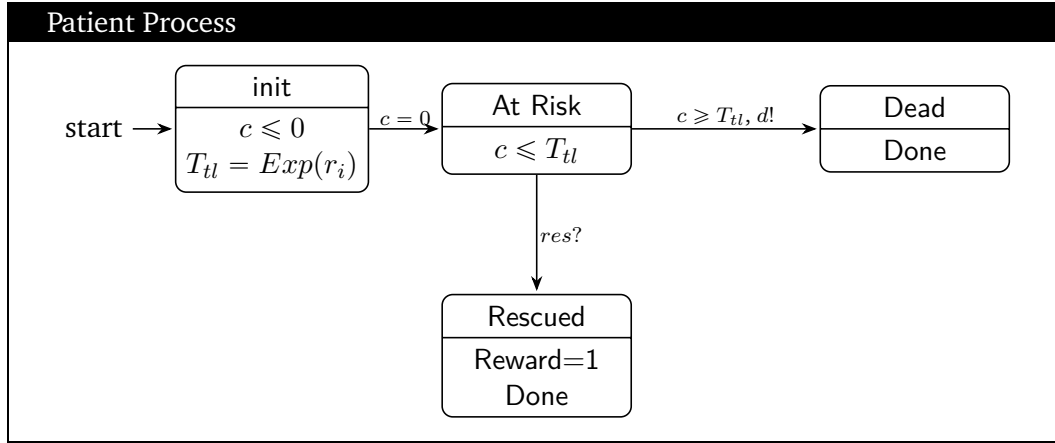


Fig. 9.17: State-transitions for patients in MoDeST

clinical conditions is same as before namely, *At Risk*, *Rescued* and *Dead*. Unlike our modeling approach in PRISM with descritized survival curve and stochastic probabilistic death, here in MoDeST, death time is sampled from a probability distribution and incorporated as clock constraint for patient process. Up to this time, the patient stays in state *At Risk* if not rescued, following by a death transition after this time in case rescue robot has not finished the treatment. Otherwise, the state will change to *Rescued*.

The level of abstraction is therefore evident in the absence of survival functions, continuous stochastic survival probabilities and clock-dependent rate constraint. Although this structure tries to encode key elements of IMCC model, it falls short in capturing real time behavior of victim's status. However, the fact that the robot is not aware of this *Time to Live*, imitates the least desired level of uncertainty.

Robot Process

For the robot process in MoDeST, the trajectory still mirrors the logic we introduced in the PRISM encoding, yet the structure of its states is aligned more closely with the definition of IMCC itself. This similarity arises from the compositional nature of our model, which fits naturally with the semantics of STA and SHA, while at the same time reflecting the fact that the semantic foundation of MoDeST differs from the game-based encoding used in PRISM.

In this setting, the robot is faced with several options corresponding to the set of victims. Its decision step is modeled as nondeterministic: the robot selects one victim and then measures the travel time required to reach that individual. This elapsed time becomes the basis for determining whether the rescue is possible. If

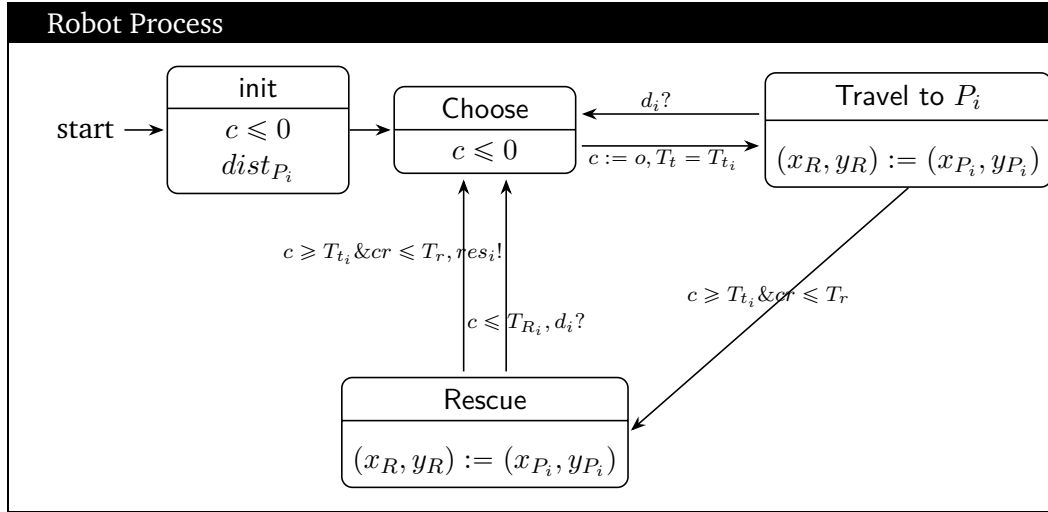


Fig. 9.18: State-transition diagram for a single robot in MoDeST

the patient's death occurs before the robot's local clock reaches the required travel time, communicated to the robot by a death signal emitted from the victim process, the robot abandons the attempt and returns to *Idle/Choose* to make a fresh selection among the remaining patients. Conversely, if the robot's clock indicates that the travel time has been met or exceeded, it concludes that it has arrived at the victim's location and initiates the rescue action.

In this formulation we deliberately omit both a stochastic rate and an explicit duration for the rescue operation itself. The outcome is thus determined without resolving the race condition between the robot's intervention rate and the patient's mortality rate. When a rescue is deemed successful, the robot sends a rescue signal to the corresponding patient process, which transitions its state to *Rescued*.

In MoDeST, we can avoid one of the key limitations that appeared in the PRISM encoding. In PRISM, although the total travel time could be expressed, the robot's position could not be incrementally updated during its movement. As a result, whenever a victim died before the robot arrived, the model failed to compute the partial distance already covered and simply left the robot at its previous coordinates. This frozen position introduced an accumulating error that undermined the optimality of subsequent decisions.

By contrast, in MoDeST we model the robot's clock explicitly as a variable with rate 1. This allows us to read the passage of time continuously and, upon receiving a death signal from the victim process, to compute the robot's actual position according to the elapsed local time (assuming straight-line travel). While this still simplifies the geometry, it captures the progress of the robot's movement more accurately than the static approximation in PRISM.

Nevertheless, a central difficulty remains unresolved: the robot's choice of which victim to approach is nondeterministic and does not incorporate the priority structure we ultimately aim to model. Since patient deaths occur stochastically and the robot has no foreknowledge of when they will happen, it cannot align its choices with either distance to the patient or the survival probability curve. As a result, the selection mechanism here still abstracts away from the intended priority-based strategy.

Observable Variables and Synchronization

In this MoDeST encoding, the processes interact through synchronized actions. Death and rescue signals are broadcast across modules, ensuring that all relevant processes update consistently when an event occurs. The distance between a robot and each victim is represented as a function of their respective coordinates, which allows us not only to compute travel times but also to reconstruct the path traversed by the robot during execution.

For the robot, the computed distance to each victim is treated as observable: it is assumed to know the spatial layout of the disaster scene and the locations of the victims. Beyond this positional knowledge, however, the robot has no access to internal patient variables; those aspects remain hidden. Finally, the entire system executes in parallel, with robot and patient processes evolving simultaneously, reflecting the inherently concurrent nature of the rescue scenario.

9.5.4 Multi-Agent Model in Disaster Responses

The multi-agent configuration follows the same approach as the single-agent, with an additional constraint among robots and patients reflecting robot's available options. This is encoded in the patient's availability, not only when rescued or dead, but also when chosen to be rescued by one particular robot. This way, the second robot cannot choose this patient anymore and it will be eliminated from the list.

9.5.5 Implementation Results

For MoDeST, we pose the same question – expected rescues before the maximum time($T_{\max}$) with one or two robots as controlling agents, using the Modest language and the *modes* back end. Since the conversion of clocks turns our model from STA to SHA, generating strategies via *mcsta* was no longer feasible. As mentioned, to accommodate time-varying survival hazards, we sample each patient's hazard/injury

from a rate distribution and define guards and travel/treatment clocks while keeping analysis tractable.

We then compute time-bounded reward results and provide representative execution traces and bounded intervals for maximum reward expectations. Each iteration, collects the number of patients saved and accumulates them over a run as a cumulative reward. We then evaluated the property `MaxRescue` (our reward structure) that coincides with the total number of rescues achieved in that run. Results in Figures 9.19 and 9.20, show the output of *modes* engine for rescuing three patients by one and two Robots, respectively, mapped to Stochastic Hybrid Automata.

Over 50 simulation runs and rescue traces, the estimated expected reward is 1.35 for one robot and three patients, meaning that the model and policy rescue about 1.35 (between 1-2) victims on average per execution under the assumed uncertainties.

As illustrated in the results, the estimated expected reward(rescued patients) over 50 runs is not significantly different for scenarios where two robots are involved where the confidence interval is [1.09,1.5] compared to the [1,16, 1.55] from one robotic agent. Same results are achieved when the number of runs are much higher (i.e. 400 times), with only a tighter confidence interval and closer estimated mean value of rescued patients between these two cases.

Intuitively, it is expected that having two robots would increase the likelihood of rescuing more patients. However, random sampling method for survival likelihood and the resulted confidence interval, reflect the high stochasticity of the the problem merged with the implementation constraints.

9.6 From Verification Limits to Practical Planning

Both PRISM and MoDeST represent significant advances in the field of probabilistic model checking, offering powerful capabilities for analyzing complex stochastic systems, each with its own strengths and areas of ongoing research.

PRISM offers efficient numeric engines for Markov decision processes and stochastic games; MoDeST adds expressive guard syntax, continuous distributions, and direct links to statistical model checking. For many safety-critical applications either tool delivers analyses that would have been unattainable only a decade ago.

Yet our Interactive Markov Chains with Clocks (IMCC) and related scenarios it models, pushes both environments beyond their comfort zones. To encode them in PRISM, we had to flatten the rescue sequence into a myopic “best-current-choice”

rule, omit continuous position updates, and rely on randomized tie-breaking to lessen combinatorial blow-ups. Even after those concessions, the state space becomes intractable once the team exceeds three robots and five victims.

MoDeST, in turn, can generate strategies only up to Stochastic Timed Automata; the moment we intend to query clocks, the model lifts to a Stochastic Hybrid Automata implementation and strategy synthesis is no longer supported. The richer timing semantics we need including hybrid clocks, dynamic travel times, and nonmemoryless priorities, remain outside the reach of both solvers.

In short, the answers produced by the tools are informative but cannot be taken as decisive evidence: key parts of the real decision process are either approximated or dropped. Nevertheless, our formalization effort diminishes through the many compromises and considerations.

Those insights point to a pragmatic way forward. The unavoidable gap between the level of abstraction and reality motivates the next chapter, where we shift from exact verification to near-optimal real-time planning. We have shown that once we accept bounded-horizon look-ahead, the rescue problem admits strategies whose expected performance is provably within a small, quantifiable gap of the theoretic optimum; an optimum that is unreachable in practice but no longer needed for effective field deployment.

The next part, therefore, turns the bounded-horizon insight into a concrete decision pipeline, focuses on decision requirements explicitly, and shows how to confront uncertainty and runtime variability while still achieving robust, consistently acceptable performance in the field.

```

+ Property MaxRescue
  Estimated expected value: 1.3599999999999999
  Confidence interval:      [1.1629988313887607, 1.557001168611239]
  Interval half-width:     0.19700116861123917
  Runs used:               50
  Distribution support:     [0, infinity)
  Run type:                Singular SHA
  Status:                  Finished

+ Sample data
  Mean:                    1.3599999999999999
  Variance: 0.48000000000000004
  Skewness: 0.5123932625209708
  Kurtosis: 3.160627516312646

+ Statistical guarantees
  Method: Student's t confidence interval
  Statement: Approx. 100*γ % of the confidence intervals contain the true probability
  Caveats: CLT assumption
  γ:      0.95

+ Property MaxRescue
  Estimated expected value: 1.2625
  Confidence interval:      [1.1933536674312184, 1.3316463325687815]
  Interval half-width:     0.06914633256878155
  Runs used:               400
  Distribution support:     [0, infinity)
  Run type:                Singular SHA
  Status:                  Finished

+ Sample data
  Mean:                    1.2625
  Variance: 0.4948308270676691
  Skewness: 0.05971223969053265
  Kurtosis: 2.7169329133030735

+ Statistical guarantees
  Method: Student's t confidence interval
  Statement: Approx. 100*γ % of the confidence intervals contain the true probability
  Caveats: CLT assumption
  γ:      0.95

```

Fig. 9.19: MoDeST Synthesis for one Robot and three Patients' Mission (at 50 & 400 runs)

```

+ Property MaxRescue
  Estimated expected value: 1.2999999999999998
  Confidence interval:      [1.0908908288185337, 1.509109171181466]
  Interval half-width:     0.2091091711814661
  Runs used:               50
  Distribution support:     [0, infinity)
  Run type:                Singular SHA
  Status:                  Finished

+ Sample data
  Mean:                    1.2999999999999998
  Variance: 0.5408163265306121
  Skewness: 0.0933015151349586
  Kurtosis: 2.7144891420434325

+ Statistical guarantees
  Method: Student's t confidence interval
  Statement: Approx. 100*γ % of the confidence intervals contain the true probability
  Caveats: CLT assumption
  γ:      0.95

+ Property MaxRescue
  Estimated expected value: 1.2500000000000016
  Confidence interval:      [1.184161361024288, 1.315838638975715]
  Interval half-width:     0.06583863897571351
  Runs used:               400
  Distribution support:     [0, infinity)
  Run type:                Singular SHA
  Status:                  Finished

+ Sample data
  Mean:                    1.2500000000000016
  Variance: 0.44862155388471164
  Skewness: 0.0626342256496032
  Kurtosis: 2.794076339689774

+ Statistical guarantees
  Method: Student's t confidence interval
  Statement: Approx. 100*γ % of the confidence intervals contain the true probability
  Caveats: CLT assumption
  γ:      0.95

```

Fig. 9.20: MoDeST Synthesis for two Robot and three Patients' Mission(at 50 & 400 runs)

Decisions

Algorithms that Save the Most Lives

Rescue Scheduling as an Optimization Problem

“We can only see a short distance ahead, but we can see plenty there that needs to be done.

— Alan Turing

With the modeling groundwork finally in place: environmental dynamics encoded, agent interactions formalized, clocks ticking and survival curves fading, we can step back and ask what all those numbers still leave unsolved. Part I surveyed the nature of disaster scenarios, delineating the metrics and operational constraints that shape rescue-oriented decision making. Part II introduced a high-fidelity mathematical model of a representative rescue scenario in disaster environment and showed that, although the model itself captures the underlying dynamics with precision, the sheer scale and uncertainty of the system make it computationally infeasible to extract an exact, real-time optimal strategy using current verification techniques. Confronted with this tractability barrier, Part III turns the limitation into a design mandate: to craft practical, real-time scheduling policies that maximize expected lives saved while remaining consistent with and informed by, the rich structure of the theoretical model.

Rescuing victims is more than moving robots from point A to point B; it is a scheduling problem where every assignment trades one life and gives chance to another. A robot's choice of victim, route, and timing can be written as an optimization model: maximize the expected number of survivors subject to travel times, uncertain hazards, limited medical kits, and strict global deadlines. In other words, we are no longer just describing the disaster scene; we are turning it into a set of variables, constraints, and an objective function that must be solved, as fast as it still matters.

The question addressed in this part is “How to optimize operation time and cost of an autonomous rescue mission in unknown catastrophe environment using a swarm of robots?” which will be answered through two sub-questions:

- What are the content and process of strategic decision making in unknown environments?

- What approach will best support decision making process to result in an almost optimized solution?

The order of these questions is important. The first question identifies the system requirements and the second detects a possible system design to meet those requirements.

Prior to proposing a solution, we need to know what has already been tried. The next section therefore reviews decision-making methods for dynamic and high-risk setting among which are classic Markov Decision Processes (MDP) and Partially Observable MDP (POMDP) formulations, auction-based multi-robot task allocation, heuristic algorithms, reinforcement learning in changing terrains, and the recent surge of statistical-model-checking hybrids used in search and rescue missions. Seeing where these approaches succeed and where they stall sets the stage for identifying the gaps our own method must fill.

10.1 Decision-Making under Uncertainties: A Survey

Decision making plays the key role in unexpected incident scenarios such as search and rescue. The faster we decide what to do, the fewer the losses become. Time is therefore a first-class design variable, and time management sits at the top of any decision procedure in such incidents; when decisions lag, consequences quickly become irrecoverable. In practice, this urgency intersects with massive uncertainty: survival chances of victims degrade stochastically over time; maps are incomplete; sensing is noisy; and communication is intermittent. All these conditions demand procedures that act before the state is fully known and adapt as information arrives.

From Uncertainty to Problem Structure Through the SAR Lens

A useful way to organize uncertainty for Search and Rescue (SAR) problems and related ones, is to separate (i) environmental/hazard dynamics (time-varying risks, blocked passages, evolving threats), (ii) perception and action (sensor noise, localization drift, actuation variability), and (iii) coordination and communication (team size, link quality, latency). This taxonomy is consistently tied to path planning and task allocation design where the choice of algorithm is shaped as much by which uncertainty dominates as by raw problem size. For example, studies cluster environmental change, partial observability, stochastic task appearance, and communication loss, as the core uncertainty families [MM24; AM25; Mal+25];

then map the algorithmic responses (probabilistic planning, robust optimization, heuristic/metaheuristic, and learning-based control) to those families [Tha+24; KPG25; KJ23; Pu+25].

SAR adds a distinctive time-critical layer: survivor viability decays with waiting time, which couples routing and scheduling in a way that makes objectives nonconvex and discontinuous (rescue before a deadline or lose the patient). In our modeling, this appears as patient-specific survival functions with instantaneous death hazards that force the planner to re-evaluate sequence decisions as clocks advance. Formally, this welds a routing objective to time-dependent reward (expected survivors), and it explains why *good but in time* is often better than *perfect but too late*.

Modeling Choices under Partial Observability

On the formal side, probabilistic planning frameworks such as Partially Observable Markov Decision Processes (POMDPs) and multi-agent extensions, encode partial observability, belief updates, and stochastic outcomes in a principled way [GD05; Spa12]. Among literature, there is broad consensus that POMDP-style reasoning and reinforcement learning (RL) are the canonical models for acting under uncertainty in robotics, including SAR missions [Kur21; CFH16; Ong+10]. Their value lies in coupling information gathering with action meaning that a robot decides not only *what to do* but also *what to observe next*, turning sensing into part of the plan [Zho+19]. The challenge here is scale; exact POMDP solution is intractable for large belief spaces and multi-robot teams, which motivates online and approximate solvers (point-based value iteration, limited-lookahead control) and hierarchical decompositions that preserve probabilistic grounding while meeting time budgets [Bha+23; Ama+16].

Reinforcement learning (RL) bridges modeling and computation by learning policies directly from interaction. In SAR-style domains, recent work emphasizes multi-agent RL, belief-aware or Bayesian RL, and heuristic-guided Multi Agent Reinforcement Learning (MARL) to improve sample efficiency and stabilize learning under partial observability [Omi+17; Mao+20]. These strategies support adaptive path planning, victim identification, and resource allocation with online updates, though many systems still require guardrails including risk-aware rewards, curriculum design and safety layers, to remain stable in the field. Therefore, probabilistic foundations remain, but practical implementations skew toward hybrid stacks such as limited-horizon planning with belief updates, plus learning for prioritization and operator tuning [Ahm+20; GPL20].

Robust, Stochastic, and Scenario-based Formulations

Beyond belief-state planning, two complementary families appear frequently in SAR logistics and task scheduling:

1. *Robust optimization / stochastic programming.* Here, uncertainty enters as intervals, scenarios, or chance constraints on demand, travel times, or task success. They produce plans that remain feasible or near-optimal across many plausible worlds. However, it comes at the cost of being conservative and, at times, incurring computational overhead. Robust formulations are effective for relief distribution, evacuation routing, and heterogeneous-team scheduling when the uncertainty set is well-bounded [KHH18; BJM13]; they struggle when the world shifts outside the scenario family or when the decision cycle is too short for large scenario sets [Mai+16; YJY20; Noo+25].
2. *Heuristic/metaheuristic and hybrid planners.* These methods accept that objectives will be nonconvex, discontinuous with noisy, simulation-based evaluations, and that the world will change mid-computation. Their value proposition is anytime performance as they find a feasible plan fast and improve it when time allows while tolerating noises. Scientific literature emphasize that Genetic Algorithms (GA), Ant Colony Optimization (ACO), and Particle Swarm Optimization (PSO) adapt well in dynamic rescue contexts, especially when hybridized with clustering or QoS(Quality of Service)-aware constraints; but it also indicates the limits among which are parameter sensitivity, premature convergence, and the persistent gap between simulation and field validation [Bla07; Wu+21; Wu+22].

Architectural Patterns for Autonomous SAR

Algorithm choice cannot be separated from architecture. Previous studies converges on decentralized or hybrid hierarchical designs because communication is imperfect and teams are heterogeneous [PR20; Mac11]. At the *team* layer, lightweight market/auction allocation or consensus distributes tasks quickly and degrades gracefully when links drop [Wan+22; BB21]. At the *robot* layer, a route-level planner, no matter exact, heuristic, or hybrid, performs local scheduling and replanning [Rob+15; HD23; Yan+23]. Eventually, at the *parameter* layer, continuous optimizers tune hazard-rate coefficients, penalties, or timing offsets online [Kam+24; Mah+22]. This division of labor keeps global intent coherent while allowing local autonomy to handle surprises, and it meshes cleanly with both probabilistic planning and heuristic stacks.

Within this architecture, two simple rules can keep things fast and stable. First, the scheduler should not start each iteration from scratch but carries over the current plan and belief state, then update only what has changed. This cuts reaction time and avoids jitter when data keeps arriving. Second, the constraints can be enforced inside the planner, therefore, eventually shaping the search itself. Doing this is what makes real-time operation feasible and deals with underlying uncertainty.

According to literature, heuristics and metaheuristics -individually or in a hybridized setting– regularly produce high-quality plans under uncertainty [Jua+23; Zha+22; Tal09]. Probabilistic and learning-based methods contribute principled reasoning about belief and risk [Fis+18; Guo+22]. In addition, decentralized coordination improves resilience and scale. Yet, many studies are simulation-heavy and scenario diversity is limited [Jua+23; Yan+20]. The performance metrics vary across papers and field validations remain scarce, especially at scale and under hard communication constraints. This heterogeneity of metrics (path length, smoothness, collision avoidance, coverage, QoS compliance, response time, robustness to change) complicates direct comparison and underscores the need to state uncertainty assumptions explicitly and to measure adaptation speed and noise tolerance alongside solution quality.

Connecting these strands back to the rescue objective, a coherent picture emerges that points straight at the challenges of dynamic environments. The SAR decision problem is *routing plus scheduling* under degrading rewards, observed through partial and noisy information, executed by heterogeneous teams with imperfect information and links. The literature motivates procedures that are (i) anytime (fast to feasible, improving when possible), (ii) belief-aware and noise-tolerant (stable under stochastic evaluation), and (iii) constraint-expressive and warm-starting (able to honor safety/resource limits and adapt quickly to new reports).

10.2 Dynamic Environment and Its Challenges

Unlike factory or warehouse settings, a disaster zone evolves while the rescue operation unfolds. Aftershocks can block previously clear corridors, new hazards may emerge, and triage information is updated irregularly as victims are discovered. From a modeling perspective, the environment is not merely stochastic but non-stationary: the transition dynamics themselves change over time, invalidating policies that assume fixed probabilities. Such drift necessitates decision rules that can learn or re-optimize on the fly without exhaustive recomputation. Among the vast variety of parameters, which constitute to an almost never-ending list, following subset of variables are of the most prominent ones:

- *Run-Time or Real-Time*: Quickly re-optimizing in a dynamic environment can be challenging without additional optimizations or parallel implementations.
- *Uncertain/Varying Data*: Patient states, death rates, and detections drift over time. Policies should be robust to model error and update beliefs on-line where sensors are noisy or new patients appear, so the scheduler can adapt when those clock-dependent risks change.
- *Scalability*: Large-Scale Deployments or multi-robot coordination explode the search space and may overwhelm the algorithm. Hierarchical or hybrid schemes and approximate dynamic programming/heuristics are essential to keep planning tractable.
- *Resource Constraints*: Limited battery, tools, medications, and treatment time windows shape feasibility. Plans must be energy-aware, respect payload/consumables, and encode time/energy budgets explicitly so that “reachable and treatable within budget” is enforced, not assumed.
- *Multiple Rescue Units*: Coordination introduces interference and coupling (duplicate visits, congestion at narrow passages). We need conflict-aware task allocation (auction, consensus, or priority rules), clear tie-breakers, and light-weight conflict resolutions so that local choices remain globally consistent.
- *Communication Delays*: In multi-robot or distributed scenarios, each robot might not have up-to-date information on the others’ positions or tasks. This can delay the discovery and communication of new constraints. Delay-aware policies should rely on locally verifiable signals, tolerate asynchrony, and plan with conservative assumptions when acknowledgments are missing.
- *Physical Constraints and Hardware Limitations*: Terrain, kinematics, speed caps, sensing range, and access constraints bound what can actually be carried out. Travel and treatment times should account for these limits (and their uncertainty), and path planning must avoid unsafe slopes, debris, heat, gas, or collapse zones by design.
- *Safety Protocols*: Human safety, no-go areas, and collision-avoidance reduce optimality. The controller must guarantee reach-avoid constraints, insert safety buffers around humans/first-responders, and support safe fallbacks (safe stop/retreat) when risk estimates spike.

After reviewing decision-making in dynamic environments and, specifically, in robotic systems and under the aforementioned key parameters we must consider, one

conclusion is clear: the problem remains chaotic even under the IMCC structure, with the uncertain data and turbulent environment. Existing system architectures offer partial remedies, but many of them degrade in real deployments due to stale information, communication delays, or prohibitive run-time. We therefore require a strategic control algorithm that can deliver a near-optimized solution under hard deadlines, not just in simulation but on the ground.

Concretely, the controller we need must (i) be anytime and replan incrementally, (ii) absorb uncertainty via online belief updates, (iii) scale with team size and map complexity, (iv) enforce safety/triage constraints by design, and (v) remain auditable for mission leads. According to the undecidability result established for our model, that determining an optimal strategy, and even an ε -optimal one, is undecidable in the general case, our algorithmic stance cannot be solved exactly and be done. Rather, it must be approximate wisely and adapt. This result formalizes why exact dynamic programming or exhaustive search is untenable when decisions must be made in a non-stationary, partially observed rescue environment with external events and continuous feedbacks.

Previous chapters and reviews lead directly to two realities: disaster environments change as we act, and exact optimization explodes once a few robots and victims are added. Dynamic hazards break many of the independence assumptions that make closed-form solvers attractive, while partial observability forces huge belief spaces. These limits explain why field teams still rely on well-chosen heuristics that give near-best results in real time.

In the restricted variant of the model (Section 6.2.3), where state evolution and action spaces are reduced and information structures are bounded, we can bound the approximation gap and drive it below a target ε by tightening horizons, refining discretizations, and calibrating penalties; but even there, solution time scales sharply with problem size. Consequently, the practical path to high-quality decisions is to combine controlled approximation with heuristic search that is resilient to uncertainty and time pressure. The final part of this chapter clarifies which heuristics we keep, which we discard, and why they remain the most practical bridge between theory and a robot that has seconds –not hours– to decide.

10.3 Choosing a Heuristic: Rationale

There are a vast variety of heuristic algorithms available for decision making problems, most of which tested and proved for their functionality and performance. However, there are some shortcoming when dealing with specific problems and their extra complex characteristics. Countless research studies have shown that

heuristics (Constructive Greedy, nearest neighbor, A^*) or metaheuristics (Genetic Algorithms, Particle Swarm Optimization) can yield near-optimal tours in practice, despite the computational intractability of guaranteed optimal solutions at scale [BD16; Des+15; TTK23].

The search and rescue problem that we are addressing involves high dimensional decision space: as the number of patients and possible visitation sequences grow, the complexity quickly escalates, making exhaustive search infeasible. Also, it includes exponential decay in survival probabilities where each patient's likelihood of survival decreases over time leads to a dynamic, time-sensitive objective function. In addition to all those consideration, conditions can change in real-world scenarios not only in patient's health, but also a new patients might appear, or travel times vary.

Heuristic methods are appropriate here for three reasons. First, the search space is high-dimensional and mixed (discrete task allocations and route choices, continuous timing and resource levels), with nonconvex and discontinuous objectives arising from risk, survivability, and time-critical penalties. Second, objective evaluations are simulation-based and therefore noisy, reflecting sensor errors, delayed information, and evolving hazards. This undermines gradient-based optimization and makes local methods brittle. Third, decision quality is an anytime resource: in a rescue operation, producing a feasible plan quickly and improving it as time permits is strictly better than waiting for a certificate of optimality that will never arrive. Under these conditions, a well-designed metaheuristic provides the right balance of exploration, exploitation, and robustness.

What a Rescue Mission Needs

In our setting, exact optimization is off the table. We have shown that, in the general setting of our model, deciding the existence of an optimal policy, and even an ε -optimal one, is not feasible. The practical consequence is straightforward: we cannot rely on exhaustive search or dynamic programming with global guarantees. In a restricted variant (bounded information structure and finite horizon with controlled discretization), approximation becomes meaningful: we can push the error below a target ε by tightening the model and calibrating evaluation, albeit with steep runtime growth.

Given this landscape, we frame solution design as controlled approximation under uncertainty. The rescue problem couples discrete assignment and routing with continuous timing and resource dynamics. Objectives are nonconvex and often discontinuous due to time-critical penalties and risk thresholds. What we need is

a procedure that (i) finds feasible plans quickly, (ii) improves them as time allows, and (iii) stays robust to noisy feedback.

10.4 Shortlisted Heuristic Algorithm and Final Selection

We now assemble a disciplined shortlist of heuristic families that can satisfy these requirements under disaster-style dynamics. The set spans constructive–improve methods and population-based search at the route level, plus lightweight team-level coordination and inner-loop continuous tuners. Each candidate is evaluated by the same scale: time to a first feasible plan, quality growth under fixed budgets, stability under stochastic evaluations and partial information, expressiveness for safety and resource constraints, ability to warm-start and adapt when events arrive, readiness for multi-objective control with operator oversight, and ease of integration into a layered architecture that separates task allocation, routing, and parameter tuning. Listed below, we outline what each algorithm can deliver for our rescue setting, where it breaks, and how it complements the others; this provides the evidence base for the controller backbone we ultimately adopt.

- *Genetic Algorithms (GA).*

For our rescue scheduler, GAs provide breadth and resilience that local methods cannot [Dav91]. By evolving a population of plans in parallel, the search preserves multiple structural hypotheses as the environment changes. Carefully designed recombination operators can swap entire subroutes, reassign patient clusters, or exchange resource blocks, which yields meaningful leaps in solution space rather than incremental nudges [Sch01; LA11; GL23]. With best-solution carryover and warm starts, GAs exhibit strong anytime behavior and tolerate noisy or partial fitness signals that arise from survival curves and probabilistic patient locations.

- *Multi-Objective Evolutionary Algorithms.*

MOEA variants extend this to multiple objectives so we can expose the trade-off surface between expected survivors, response time, and risk for operator choice. The liabilities are substantial with quality often needing several generations to stabilize, representation and crossover requiring careful engineering to avoid costly repairs, and variance reduction by resampling still taking time [HYT21; VBB14]. A GA extended by MOEA is therefore ideal when we can afford a few iterations between events and we need global restructuring and explicit

trade-offs; it is less suitable under sub-second deadlines or when the instance is small enough for an exact solver [Deb15; CLV07].

- *Adaptive Large Neighborhood Search (ALNS / LNS).*

ALNS is suitable for online re-optimization. Large-neighborhood destroy–repair can rewrite substantial parts of a route when a patient is discovered or a path is blocked. Operator weights adapt to what is working, and time-window and resource constraints are handled naturally by insertion and removal designs, which keeps latency predictable and makes parallelization straightforward [PR18; PGM17].

The limitation, however, is its scope. ALNS reasons around the incumbent, and can stall if the operator set is narrow or poorly tuned, and it may miss far-away structures that require coordinated global change. It should be the primary repair engine between sensor updates, while acknowledging that structural overhauls or explicit multi-objective balancing may require a GA layer above it.

- *Variable Neighborhood Search (VNS) and Iterated Local Search (ILS)*

VNS and ILS are lean improvers that give steady gains under tight budgets. They make controlled tweaks to the current plan and then clean it up with a fast local solver. This is useful for polishing ALNS or GA outputs and for absorbing small environmental changes without invoking heavier machinery [LMS18; LMS22; DD20].

Their local view, however, makes progress depend on neighborhood richness and on a good shake-refine balance, which means sparse moves or weak shaking slow them down, and coordinated long jumps are often out of reach. They are, thus, best used as low-overhead intensifiers between major replans, smoothing small disturbances instead of driving global redesign.

- *Tabu Search (TS).*

Tabu Search offers memory-guided exploration that avoids cycling and handles complex constraints through move design and tenure control. It is a dependable baseline for dynamic routing with time windows, precedence, and safety buffers, and it integrates cleanly as an intensifier around ALNS or GA incumbents [Gmi+21; BG02b; Joh25].

The limits remain those of single-trajectory search: it is fundamentally local and can become brittle if tenures and aspiration rules are mismatched to the pace of events. TS is best when predictable, constraint-aware improvement

with modest tuning effort are desired, and less appropriate when structural changes dominate.

- *Greedy Randomized Adaptive Search Procedure (GRASP).*

GRASP excels at producing a good feasible plan fast. A randomized greedy constructor gives you a valid schedule under changing information, and a short local improvement pass tightens it. This makes GRASP an excellent seeding mechanism for ALNS, TS, or GA, and a practical fallback when compute is scarce [RR13; RR10].

Its horizon is intentionally short and it tends to plateau early unless complemented by path-relinking or later intensification, so it should not be the sole solver when repeated nonlocal disruptions are expected.

- *Ant Colony Optimization (ACO).*

ACO builds routes through pheromone-guided choices that favor subpaths with a history of good performance. It is a credible constructor for routing, and hybrids that pair ACO with a strong local search can handle time windows and side constraints effectively [YY11; BLR11; He+21].

The downside appears in non-stationary settings: historical pheromones can mislead when costs change rapidly, evaporation and restart policies need care, and tightening solutions can be slower without an improver. ACO fits when conditions are moderately stable between updates and when reuse of subpaths is valuable. Thus, it is a weaker choice under rapid shifts.

- *Market- and auction-based MRTA (team-level coordination)*

Auctions provide fast, decentralized task allocation across robots, which removes a central bottleneck and tolerates intermittent communication. This pairs well with a route-level heuristic on each robot and gives a practical path to scale [ENT11; SSP17; LS13].

The limitation is short-sightedness, because simple bidding overlooks strong task interactions, while combinatorial auctions capture those interactions at the cost of heavy computation and extra messaging. In practice, auctions work well when sectors can be partitioned and cross-sector coupling is moderate, but they should not be the only mechanism when precedence chains and tight interference are routine.

- *Monte-Carlo Tree Search (MCTS / UCT) as a real-time decision gate*

MCTS is useful when the mission presents discrete, high-impact branch points such as choosing search versus rescue now or prioritizing one sector over another. It runs under a strict time budget while using a heuristic evaluator for rollouts, so it can return a usable decision even with limited simulation [Cap21; JKR17; Świ+23].

Quality depends on the fidelity and speed of the simulator and on a manageable branching factor; if these are lacking, MCTS will waste effort on dead ends. In practice it fits best as a gatekeeper for a small set of pivotal choices, not as a full planner.

- *Differential Evolution (DE), Covariance Matrix Adaptation Evolution Strategy (CMA-ES), and Particle Swarm Optimization (PSO) for continuous tuning*

These derivative-free optimizers are valuable for tuning continuous parameters inside the discrete planner, such as hazard rates, penalty weights, or timing offsets. They are robust to noise, easy to parallelize, and can run in short bursts with warm starts [RBL21; ASV22].

They are not primary solvers for combinatorial routing and tend to stall on flat landscapes or drift when the signal is weak, so they work best as bounded inner loops that refine parameters and then return control to ALNS, Tabu search, or GA, with simple guardrails (box constraints, step caps, early-stop on no-improvement) to avoid overfitting transient noise.

In our setting, anytime performance comes from methods that can keep improving a feasible plan under tight budgets: GRASP delivers a fast constructive start and short local tightening, ALNS/LNS, VNS/ILS, and Tabu Search act as steady improvers with predictable latency, and population methods such as GA/MOEA provide continued global exploration while preserving good incumbents. Under stochastic evaluations and partial information, population diversity in GA/MOEA and multi-start GRASP tends to smooth out noise, whereas single-trajectory improvers like VNS/ILS or Tabu Search can become brittle if move acceptance and aspiration are driven by unstable local metrics; ACO is also sensitive when its historical bias conflicts with rapidly changing costs.

For constraint handling, ALNS/LNS and GA admit rich, problem-aware moves and repairs for time windows, capacities, precedence, and safety buffers, while ACO needs careful encoding to respect such constraints and DE, CMA-ES, and PSO are best reserved for continuous parameter tuning rather than discrete feasibility. As for exploration versus exploitation, population methods maintain diversity and resist early lock-in, whereas single-trajectory families exploit well but may stall in complex basins unless neighborhoods, tabu tenures, and perturbations are curated.

Online adaptability is strongest when planners warm-start and repair rather than rebuild: ALNS/LNS, VNS/ILS, and Tabu Search naturally pick up from the current plan, GA/MOEA can be warm-started with small populations and elitism, and MCTS is effective as a real-time gate for a few high-impact choices but does not capture long-range resource coupling on its own. Finally, for multi-objective demands, GA/MOEA are natively ready to expose trade-off frontiers among expected survivors, response time, and risk, while the other heuristics typically require scalarization and careful weighting to approximate the same balance.

10.4.1 Chosen Method: Genetic Algorithm (GA).

Exact optimization of the full rescue decision processes becomes computationally prohibitive once the team size or victim count exceeds a handful, a fact underscored by the state-space explosion results in Chapter 9. Consequently, practical deployments rely on heuristics that approximate the optimal policy while guaranteeing bounded computational effort. Selecting an appropriate heuristic is itself a design decision that must balance ease of explanation for human operators, robustness to model specification, and empirical performance across diverse scenarios.

The rescue problem we are addressing involves:

- *High-dimensional decision space:* As the number of patients and possible visitation sequences grow, the complexity quickly escalates, making exhaustive search infeasible.
- *Exponential decay in survival probabilities:* Each patient's likelihood of survival decreases over time, leading to a dynamic, time-sensitive objective function.
- *Potential for real-time adaptation:* In real-world scenarios, conditions can change (e.g., a patient's health deteriorates, new patients appear, or travel times vary).

While exact methods (e.g., Integer Linear Programming, Dynamic Programming) can guarantee an optimal solution, their computational complexity often becomes exponential as the problem scales. In particular, dynamic programming approaches to routing or scheduling can become infeasible for larger sets of patients. Incorporating exponentially degrading survival probabilities complicates the structure of the objective function, making it non-trivial to formulate a direct closed-form solution.

In many path-finding or routing problems, like those solved by A*, we assume a single known destination (e.g., get from A to B as quickly as possible) [Foe+21; Duc+14]. In our scenario, however, there is no single destination to reach. Instead, the route must incorporate multiple goals (patients), each with a rapidly degrading survival probability. Thus, the algorithm must perform the planning in a dynamically

changing timeframe (some patients may die before being reached). The goal is to rescue as many as possible (thus maximizing the cumulative survival probability), rather than to get to one final endpoint.

Given that visiting multiple nodes in an optimal or near-optimal sequence is an essential approach to our problem, it is more aligned with the Traveling Salesman Problem (TSP) paradigm [Goy10; App+11]. TSP problem formalizes the problem of ordering visits among multiple nodes, essentially capturing the essence of multi-patient rescue scheduling. Like TSP, we aim to find a tour or sequence that visits each relevant node (patient) within constraints. By casting the rescue task into a TSP-like form, we tap into a rich toolkit of proven methods and well-understood theoretical properties. In rescue mission, however, we may not always visit all patients if they expire, but the concept of framework is very similar: we have to order visits to multiple locations.

The TSP is well-known to be NP-hard, meaning finding an exact solution quickly becomes computationally expensive for larger instances. As a result, heuristic or metaheuristic algorithms such as Genetic Algorithms, are widely used to approximate TSP solutions effectively [GSB25; Vik16]. As heuristic approaches, essentially, mean guided trial-and-error or incremental improvement, they are particularly well-suited for large or dynamic problems. On the other hand, the TSP-like structure ensures the route is systematically improved (via selection, crossover, mutation), yet remains flexible to incorporate special constraints like survival times.

Moreover, GAs have a long history of success for TSP and related combinatorial optimization problems, using specialized crossover and mutation operators that preserve route structures [KA11; CR06]. This track record reassures us that GA-based solutions are not just theoretically sound but also practically effective for route optimization. The parallel search with multiple candidates naturally fosters diversity, crucial for avoiding local optima [DGT12]. This diversity is especially beneficial in a problem like search and rescue, where “nearby” solutions may yield vastly different outcomes due to exponential decay effects. Unlike purely greedy or local search methods, GAs maintain a population of candidate solutions. This allows simultaneous exploration of multiple routes, helping avoid local optima, particularly important in a dynamic rescue setting where certain patients might become unreachable as time passes.

In addition, GAs are inherently capable of parallel exploration. This principle naturally extends to multiple robots because we are effectively evolving multiple routes simultaneously. Also, the real-coded or permutation-based route representation can be modularly extended. Markers, segments, or multi-chromosome structures can handle multiple tours. Even if the environment is partially known, we can

incorporate expected survival probabilities, travel times, or incorporate real-time data updates. The GA remains flexible enough to handle these changes.

These factors together explain why Genetic Algorithms are a strong, pragmatic approach to maximizing cumulative survival probability in a time-constrained, exponentially degrading environment, as encountered in search and rescue missions. In essence, a GA provides sufficient flexibility, robustness, and scalability to handle the complex and uncertain nature of real-world search and rescue scheduling, making it a justifiable and effective choice. If needed, GAs can be hybridized with simpler heuristics. Such synergy can boost performance while retaining the global exploration benefits of GAs.

Strategy of a Savior

When disaster strikes, the clock starts running long before the first robot set foot in or the first medic reaches the scene. In real-world scenarios, those in which we can no longer fine-tuning parameters inside a verifier, we are standing at the edge of the rubble deciding whom to reach first, which corridor to risk, and when to pivot because a victim's chance has fallen below the point of no return.

Up to this point, we have pulled that chaos apart, layer by layer: quantifying the dynamic nature of the disaster, charting survival curves and death rates, mapping travel times, and encoding it all in formal models. Metrics gave us criteria, Automata and state-transition systems provided the structure, and model checking revealed the point where theory collides with real-world combinatorial complexity. We now know how the scene evolves, what each minute costs, and that numbers alone do not save lives.

11.1 Whom to Rescue First?

Our core goal is to maximize the expected number of lives saved by determining which patients to visit and in what order, knowing that each patient's probability of survival decreases over time unless they are rescued. The main consideration is that not all patients may be reachable in time if their survival probability drops to near zero (or if time runs out).

The order of visits critically affects the arrival time for the rest of patients, influencing who is rescued in time and who is not. In other words, the later our agent arrives, the lower the chance a patient is still alive, and this fact defines the decision point. A rescue mission is, therefore, more than just an algorithm; it is a chain of inter-locking functions that turn raw sensor data into concrete life-saving actions. This chapter moves from static diagrams to real-time operational behavior. It examines which cues a single agent must observe, disregard, or periodically re-assess so that each action it executes, moment by moment, remains consistent with the life-maximizing schedule established in the preceding analysis.

Prioritizing victims is the moral and computational core of any rescue strategy. Each potential assignment must weigh a victim's current survival probability, the time it

will take a specific robot to arrive, the feasibility and safety of the path under current hazards, and the practical cost incurred by forgoing alternative rescues. We represent the scene through survival-probability curves that reflect the evolving hazard field (heat, smoke, structural risk) around each victim and update them whenever fresh sensor data arrive or a robot changes position, keeping the system responsive to unfolding events.

11.1.1 Criteria of a Rescue Mission

As stated earlier, real-world rescue mission in disastrous incidents comes across conditions that evolve continuously: New patients might appear (e.g., newly discovered casualties). Existing patients' health can deteriorate faster or slower than predicted. Travel paths can be blocked or open up unexpectedly. For a truly effective rescue mission, the algorithm needs to adapt to these changes rather than rely on a static plan determined at the start. If robots have limited battery life or fuel, the algorithm must factor that into the route cost.

The current model focuses mainly on time and patients' survival probability. Extending the cost function and the algorithm to handle resource usage or recharging stops complicates the problem further, and algorithm would need more computation to handle these multi-objective aspects. Real robots might face movement constraints (e.g., limited degree of freedom, maximum speed, terrain limitations). Resource and physical constraints such as battery life, communication delays, and complex robot motion all demand a more sophisticated cost function. Detailed constraints, thus, inflates the state space, increasing run times. By making the boundaries of the implemented system explicit, we ensure that every additional consideration—whether clinical triage or swarm-level consensus—has a clearly marked entry point, making the algorithm modular and flexible enough for further advancements.

Fig. 11.1 illustrates the decision-making approach a robotic agent follows to issue and execute rescue task in the disastrous environment. The planning stack is as follows: everything that sits between the *scenario input* (map, robots, victims, hazards, obstacles) and the *robot's on-site actions* is a set of functions that fire every few seconds. No matter which extra objective is added later to inner layers namely, robot's battery level, victim and robot risk assessment, cost of path, CO_2 footprint, these core steps have to stay in place. New objectives, which will be considered incrementally in current and future work, become just another term in the cost and constraint criteria as build-in modules of optimization part of the algorithm.

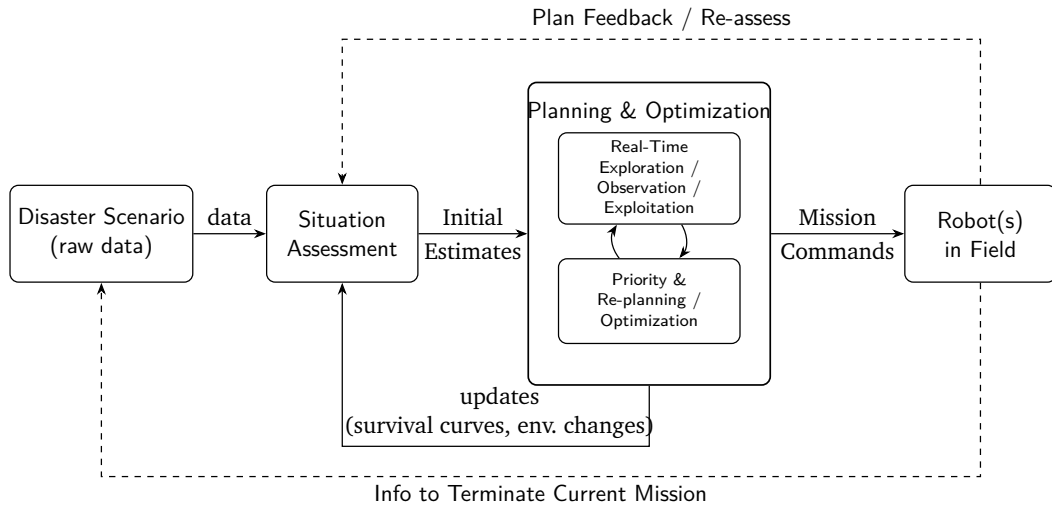


Fig. 11.1: Rescue mission decision-making approach in disastrous incident

11.1.2 From Decision Flow to System Architecture

The high-level decision flow only describes the decision process; it does not ensure a system can make those choices fast and correctly in the field. Whether the system is centrally supervised or fully autonomous, effective operation requires a deliberately engineered system architecture: modules must exchange their data through well-defined, versioned pathways, with explicit freshness and priority guarantees. Otherwise data are under-utilized or processed unnecessarily, computation budget is wasted on the wrong inputs, and –most critically– *time* is lost, which in this context is equivalent to *lives*.

What ultimately matters is not only how we prioritize which victims to rescue, but also how we prioritize the processing of information so the robot receives the right data at the right moment. A decision process is therefore only as good as the system architecture that feeds it. The architecture and the algorithms must be co-designed to minimize latency, avoid redundant work, and support rapid, well-founded re-planning in a highly dynamic environment.

Prioritized Information Processing and Re-planning Policy

Since the environment is dynamic and uncertain, not every change should trigger a full recomputation. Instead, the system needs explicit rules that determine when to propagate an update, when to interrupt ongoing actions, and when to defer work to the next cycle. Concretely:

- *Prioritized data flows.* Safety-critical updates (e.g., hazard spikes on the current path, loss of localization) pre-empt compute and communication; mission-level updates (e.g., discovery of a likely victim nearby) are handled within seconds; background refinements (e.g., small smoke drifts far from active routes) are batched.
- *Thresholds with hysteresis.* Re-planning and task re-allocation should be triggered only when a change exceeds a material threshold and persists long enough, preventing oscillations and wasted computation.
- *Interruption conditions.* The system must specify how much change in the environment justifies pausing the current action: How significant must a new detection be for the robot to care now? How large must a hazard change be to warrant an immediate detour rather than a scheduled update?
- *Separate modes of operation.* Criteria of the boundary between search (information gain and coverage) and rescue (extraction and egress), should be well-defined with formal transition guards. This prevents constant mode-switching and keeps computation budgets focused on the current objective.
- *Time and bandwidth budget.* Every stream (maps, status, commands) carries a freshness target and a time-to-live; stale data are either down-weighted or dropped, and background traffic yields first when resources tighten.

These considerations ultimately captures the essence of rescuing patients before they perish, rather than simply minimizing travel time or distance. The design target is simple: *Right data*, at the *right time*, to the *right module*. This directly aligns with real-world concerns e.g. if a patient’s survival probability is near zero, visiting them later yields almost no benefit whereas the loss of current target prior to reaching them, enforces re-planning; so the algorithm should naturally shifts priorities. “Arrange a route and schedule so that you reach patients early enough to rescue as many as possible”.

11.2 Implementation and Simulation

The core goal of our control algorithm is to maximize the expected number of lives saved by determining which patients to visit and in what order, knowing that each patient’s probability of survival decreases over time until they are rescued.

11.2.1 Model Assumption

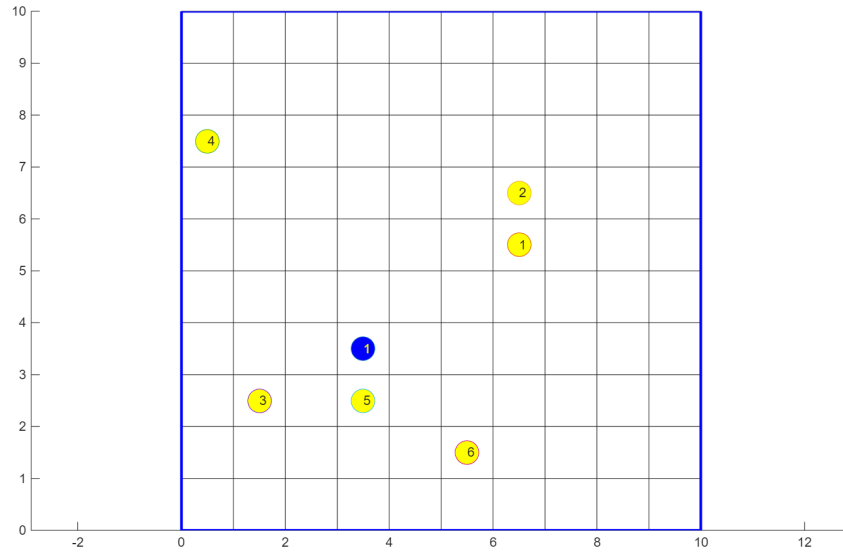


Fig. 11.2: Random location of Robot and Patients in MATLAB

Environment

The abstracted environment for this algorithm consists of an obstacle-free grid map where robot and patients are randomly located. Throughout the simulation, a sweep among variables changes the location of agents while preserving the survival probability curves as before and vice versa. It should be noted that, locations of the patients are currently considered observable to the robot.

Survival Probability Functions

In order to elaborate this purpose, each patient i has a time-dependent survival probability defined by an exponential or similar decay function. Here, the gradient of the curves capture how quickly patients i 's condition deteriorates and is eventually the clock-dependent rate that we addressed from the beginning.

Severity Levels

One of the approaches that firefighters or, in general, emergency response teams take in disastrous incident is to prioritize regions of the scene where it is more likely to find victims, as well as categorizing the severity level of their health state [HN16]. Similar to those search and rescue missions, we classify our patients' health state into four distinct levels of severity: High-risk, Medium-risk, Low-risk and lost. These

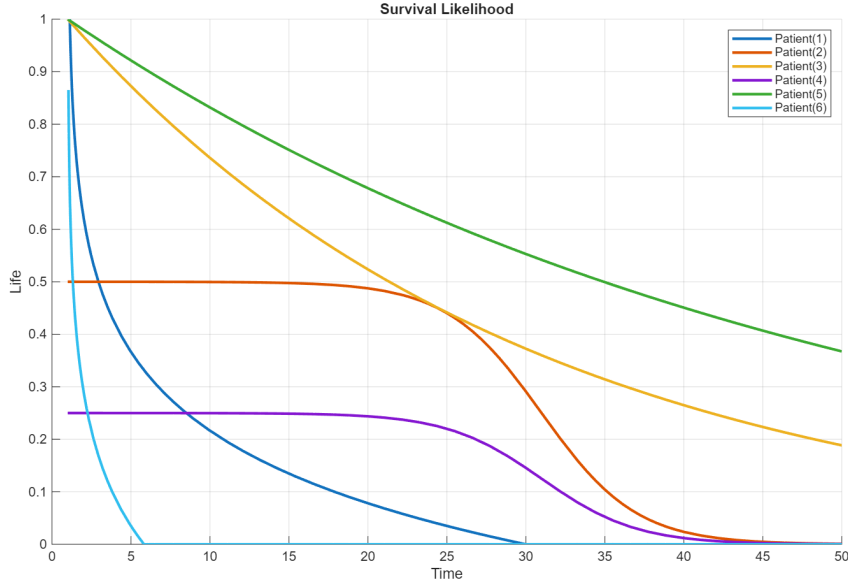


Fig. 11.3: An example of patients' survival Probability functions ($f_{\text{Survival}_{P_i}}$)

levels are attained by calculating the gradient of their survival curves, meaning their health rate.

$$\text{Severity Levels} = \begin{cases} \text{High-risk} & \rightarrow \nabla f_{\text{Survival}_{P_i}}(T + T_S) < 0, \\ \text{Medium-risk} & \rightarrow \nabla f_{\text{Survival}_{P_i}}(T + T_S) = 0, \\ \text{Low-risk} & \rightarrow \nabla f_{\text{Survival}_{P_i}}(T + T_S) > 0, \\ \text{Lost} & \rightarrow f_{\text{Survival}_{P_i}}(T_S) < \text{Survival}_{\text{threshold}} \end{cases}$$

As a result, our problem is broken down into four interconnected problems based on these priority levels. Incorporating priority weights or separate categories of severity helps ensure those needing urgent care get appropriate attention. Moreover, the cost in path planning varies from simple Euclidean distance (current implementation) to some cumulative metric resulted from a robot moving [SPG21].

Sudden Unfortunate Death

We introduce sudden death to reflect what the survival curves already imply. A survival curve $S_i(t)$ gives the probability that patient i is alive at time t which means that, at any moment, there is a non-zero chance of an abrupt fatal event. The probability that patient i dies in the next short interval $[t, t + \Delta t]$, given they are alive at t , is not an arbitrary coin flip as explained in details in chapter 8, it is:

$$\Pr\{\text{sudden death in } [t, t + \Delta t] \mid \text{alive at } t\} = 1 - \exp\left(-\int_t^{t+\Delta t} r_i(\tau) d\tau\right) \approx r_i(t) \Delta t,$$

i.e., derived directly from the survival curves $S_i(t)$. Two consequences follow:

1. By acknowledging sudden death, we align the decision problem with field reality where patients' health states do not decline only smoothly; they can be lost abruptly even when the trend looks gradual. Our stochasticity is grounded in the survival model, not injected noise.
2. The algorithm does not change structure. We simply add an event guard that uses the same inputs:

$$P_i^{SD}(t, \tau) = 1 - \exp\left(-\int_t^{t+\tau} r_i(\tau') d\tau'\right).$$

If, for the robot's current estimated time of arrival (ETA) τ , this sudden-death probability exceeds a threshold $Survival_{threshold}$ (and passes our hysteresis window), the event-based trigger fires and we may reweight/replan. If it does not, the detection is deferred to the next cycle. Ultimately, sudden death is not a new rule set. It is the real-world face of the same survival curves we already use, encoded as an additional trigger so we stay responsive to catastrophic risk without breaking the algorithm's foundation or wasting time that could be saving lives.

Time as Resource, Time as Risk

In our algorithm, *time* has two distinct meanings. The first is physical/mission time: the local or global clock against which we estimate travel and survival. In other words, how long it takes a robot to reach a location, and how long a victim is likely to survive under current hazards. The second is algorithmic/computation time: the latency we spend thinking, sensing, fusing, planning, and re-planning. In the field, these two are not independent. Every second spent computing is a second taken from the victim's survival window. That is the core trade-off in this setting, since here the cost of time is human life.

Practically, this means we must decide, at each step, whether to compute longer or act now. We compute longer only if the expected benefit of a better plan (e.g., improved arrival-time survival for the next targets) exceeds the expected loss from delaying action by that extra compute time. Otherwise, we execute immediately and defer refinements to the next cycle. Formally, for a short extra planning window Δt , we favor extra computation only when the gain in projected survival at arrival dominates the survival decay incurred during Δt , which is however unlikely in current setting; if not, we move. Therefore, the algorithm should be built to spend time only when time buys lives, and to act without least possible hesitation when it does not.

New Patients, Old rules

When an exploring robot discovers a new patient, the core of the algorithm does not change. The system is designed to be flexible enough that we simply add the patient to the state space, attach its survival-probability curve and keep the pipeline running. Allocation and planning continue under the same architecture and rules.

Crucially, we do not pause the robot's current plan just because new data arrived. Before the new patient can influence the current plan, the system pre-screens its importance. We estimate arrival-time survival for the relevant robot and check it against our decision gates. If the new finding does not clear those gates, it is queued for the next cycle and the robot does not lose time. If we were to stop and re-compute on *every* detection, overall performance would drift far from any practical optimum.

The only signals allowed to change the robot's current target are event-based triggers encoded as formal guards (e.g., a high-confidence patient whose arrival-time survival exceeds the current target by a defined margin, or a safety event (heat/collapse) on the current path). These guards are part of the model; they do not alter the foundation of the algorithm. They simply say *when* a new patient is important enough to interrupt, and *when* it should be deferred so we keep saving time and lives.

Arrival Time Calculation

If the rescue team (or robot) visits patient i at time t , then the probability that this patient is still alive (and thus rescue-able) is $f_{Survival_{P_i}}(t_i)$. The arrival time t_i depends on the route taken before reaching patient i , including travel time and any rescue durations for previously visited patients.

Let $\pi = (\pi_1, \pi_2, \dots, \pi_k)$ be the rescue sequence, where π_m is the patient chosen to be rescued at position m in the list. Maximizing the total (cumulative) survival probability across all patients we attempt to rescue, we have our cost function objective:

Cumulative Survival Probability $\equiv$

$$\max_{\pi} N_{P_{saved}} \equiv \arg \max \sum_{k=1}^{P_{SL}} f_{Survival_{P_k}}(T_{Current} + \sum_{m=0}^{k-1} ((\frac{\|q_{m+1} - q_m\|}{v_R}) + T_{Rescue}))$$

This sum is effectively the expected number of survivors if each patient's fate is probabilistic and independent. Here, $q_0 = R_{init}$ which corresponds to Robot's initial location and $q_m = P_{\pi_m}$ indicates the location of patient π_m in the sequence generated by policy π , v_R is robot's speed and $T_{Current}$ is the current time. Intuitively, the term $T_{Current} + \sum_{m=0}^{k-1} ((\frac{\|q_{m+1} - q_m\|}{v_R}) + T_{Rescue})$ is the expected horizon of patient's survival function from robot's perspective which essentially accounts for robot's local clock. Concretely, it computes the expected arrival time of robot to patient π_k in the sequence and accounts for the expected survival likelihood of patient π_k at the time of arrival:

$$T_{arrival(i)} = T_{Current} + \frac{\|P_{\pi_1} - R_{init}\|}{v_R} + \sum_{m=1}^{k-1} ((\frac{\|P_{\pi_{m+1}} - P_{\pi_m}\|}{v_R}) + T_{Rescue}(\pi_m)), i = \pi_k.$$

This means that eventually, what the algorithm intends to maximize, is the cumulative probability of what robot seems to achieve. Therefore, not all patients may be reachable in time if their survival probability drops to near zero (or if time budget runs out) by the end of this time horizon, no matter how small it is eventually computed.

The order of visits, thus, critically affects t_i for each patient, influencing who is rescued in time and who is not. In short, the objective function is:

- *Explicitly time-dependent.* The later robot arrives, the lower the chance a patient is still alive.
- *Cumulative.* The goal is to maximize the sum of these survival probabilities (or equivalently, maximize how many are still save-able).
- *Route-driven.* Robot must decide where to go next, in what order, to achieve the highest overall survival

11.2.2 Encoding a Genetic-Algorithm-Based Approach

Each solution (individual) represents a route of patient visits. We simulate the route by cumulatively adding travel times and rescue times as the fitness evaluation. For each patient i in that sequence, we compute an arrival time t_i . The arrival time to another patient, is computed dependent to which patient was rescued last before current target. We then plug t_i into each patient's survival function $f_{Survival_{P_i}}$. Finally, we sum these probabilities to get each possible route's total survival score. Through crossover and mutation, the GA explores different visit orders. Our modified GA tries to maximize that total survival probability.

The proposed strategic algorithm 2 for a single robotic agent and several victims is implemented in MATLAB software. The robot is randomly located in an environment

Algorithm 2: Rescue Mission Algorithm

Data : Coordination of robot and patients; patients' health grade
Input: $R, P = \{P_i\}, (x, y)_i, f_{\text{Survival}_{P_i}}, v_R, T_{\text{Rescue}}, T$

```
1  $FinalSequence \leftarrow \emptyset$ 
2 for  $i \leftarrow 1$  to  $|P|$  do
    // Generate Severity Matrix
3    $Severity \in \mathbb{R}^{4 \times |P|} \leftarrow \text{Severity}(f_{\text{Survival}_{P_i}}(T + T_s))$ 
    // Update lost list
4    $RedList \leftarrow P_{\text{Lost}} \leftarrow \text{row}_4(Severity)$ 
5   for  $j \leftarrow 1$  to 3 do
6     if  $\text{row}_j(Severity) \neq \emptyset$  then
7       // Approx. argmax via modified genetic algorithm
        $PriorityList \leftarrow$ 
        $P_\pi\left(\arg \max_{(S \subseteq P, |S|=P_{SL})} \sum_{i \in S} f_{\text{Survival}_{P_i}}(T_{\text{arrival}_i})\right)$ 
8     break
9    $P_{\text{Next}} = P_\pi(1)$ 
10   $FinalSequence \leftarrow FinalSequence \cup P_{\text{Next}}$ 
11   $T \leftarrow T + T_{\text{Rescue}} + \frac{\|Loc_R - Loc_{P_{\text{Next}}}\|}{v_R}$ 
12   $P \leftarrow P \setminus P_{\text{Next}} \setminus P_{\text{Lost}}$ 
13   $P \leftarrow P \cup P_{\text{New}}$ 
14   $Loc_R \leftarrow Loc_{P_{\text{Next}}}$ 
```

grid map. Victims are also positioned in the same environment at different locations and survival curves are pre-computed for each victim, randomly generated at each run and stored by victim index.

11.2.3 Trade-Offs in Rescue Scheduling

A patient with high risk (fast-decaying survival probability) might be far away, but if rescued in time, it adds significantly to the overall count of survivors. A nearby patient with slower decay may not require immediate attention; visiting them first might waste valuable time that could save more urgent patients. Each decision about which patient to visit next affects how many additional patients can be rescued before their probabilities drop to zero. If we choose the closest patient but skip a more distant, critical patient, we risk losing the critical one altogether.

Handling Trade-offs via Modified GA

Our algorithm is designed to maximize the overall survival probability sum, which correlates with maximizing the expected number of patients saved. This means the

robot might skip a patient who is physically close but still has plenty of time left, to rescue a farther patient whose survival window is about to close. If at any point data changes or time has progressed, the GA can re-evaluate and identify if going for the “far but high-risk” patient improves the total outcome over the course of the mission. If it is already too late, the algorithm won’t divert the robot unnecessarily.

Sometimes, no matter how one schedules, there simply is not enough time to reach everyone. Eventually, this algorithm simulates the sequence of visits, cumulatively adding travel and rescue times. If it arrives too late to a critical patient, that adds essentially zero to the final fitness. Hence, the algorithm weighs whether traveling extra distance to save that patient is worth the cost. Also, depending on total time budget, it might sacrifice certain low-survival patients in favor of saving multiple others who still have a chance. Hence, in the animated results, there are cases that some nearby patients turning red before rescued which is the algorithm’s strategic choice to focus on more urgent cases that yield a better overall outcome.

To test the algorithm and evaluate the impact of parameters on robot’s strategy, changes in the values of following parameters have been considered : (i) robot’s speed, (ii) treatment time required for patients. However, the stochasticity of losing patient (similar to models in PRISM and MoDeST) is preserved as the possibility of sudden death.

Moreover, tracking the total or average time spent per rescue, or the time at which the last rescue is completed is crucial to a proper rescue schedule. In some scenarios, rescuers need to minimize the overall duration of the operation (e.g., limited battery, time-based hazard expansions). A route that saves the most patients but spans an extremely long time may not be acceptable if the mission must end sooner.

Although maximizing cumulative survival probability is central since it correlates with saving the most lives possible, real-world rescue operations and ethical considerations often demand multiple metrics. Each of these metrics adds a dimension to how we evaluate the rescue mission’s outcome and ensures that the chosen route is viable and ethical in real-world contexts, not just theoretically optimal in one narrowly defined objective.

11.2.4 Implementation Results

The previous sections introduced the rescue setting, the optimization framework, and the time-critical nature of the decision problem. At this stage, it is therefore necessary to examine not only whether the proposed algorithm can generate rescue plans, but also how its structure affects performance under limited decision time. Since

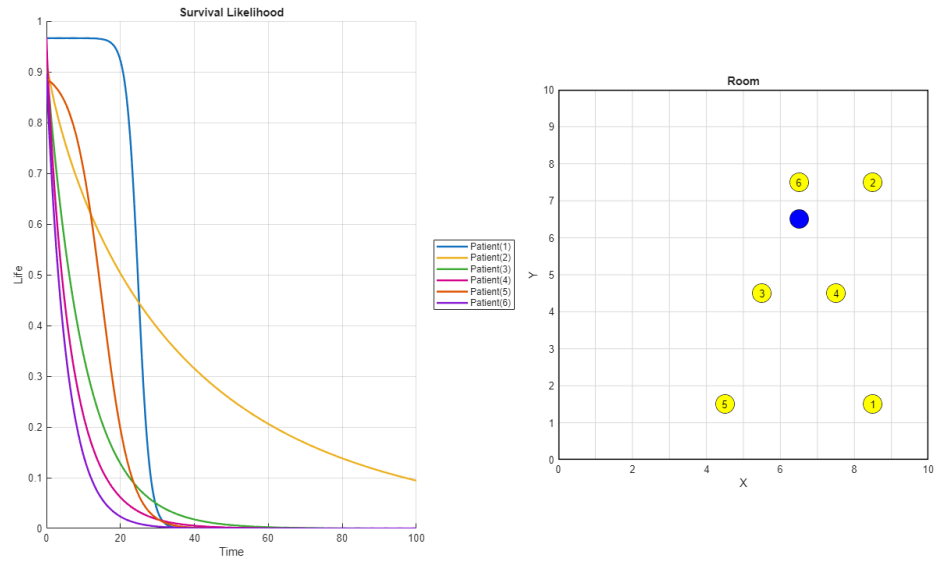
planning is carried out while the incident continues to evolve, the computational cost of the method directly influences how much time remains for execution and replanning. This motivates a separate comparison between the proposed severity-based variant and a non-partitioned version of the same algorithm.

Strategies under Uncertainty

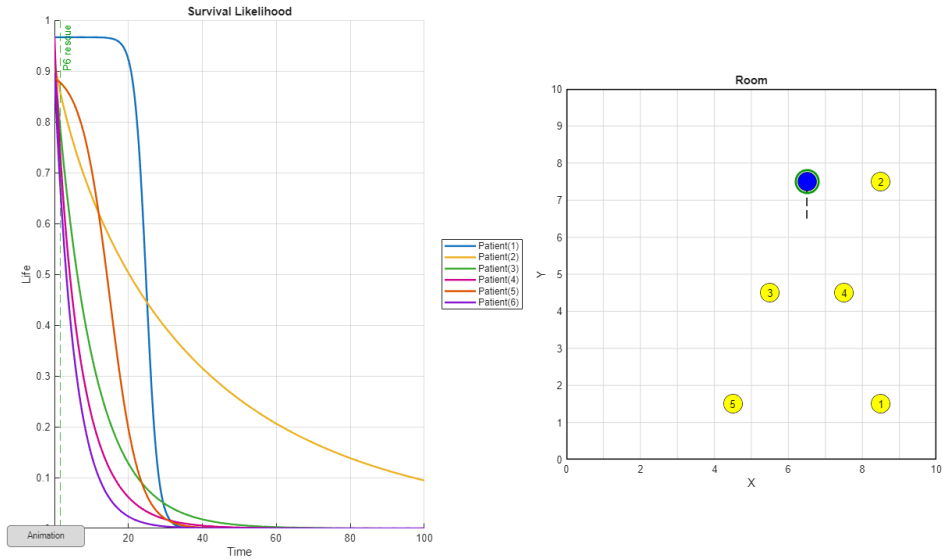
Simulation of the proposed approach has been carried out using a dedicated simulator together with an implementation of Algorithm 2 in MATLAB. The output includes an animated trajectory of the robot as it moves from one patient to another. In this subsection, representative examples are presented to illustrate the final rescue sequence and the dependence of the resulting strategy on parameters such as robot speed, travel distance, rescue time, and stochastic changes in the environment, partially presented in [Kia23].

To illustrate the behavior of the proposed strategy under uncertainty, figure 11.4 presents a representative execution scenario generated by the simulator. This scenarios are not intended primarily as a computational comparison, but rather as a behavioral demonstration of how the algorithm reacts when the environment changes during execution. In particular, the presented case covers three typical forms of disruption: the appearance of a new patient during the mission, the death of a patient while the robot is moving toward a previously selected target, and executions in which no special intervention is triggered beyond the usual stochastic evolution of patient conditions. Taken together, this example shows how the rescue sequence is updated online as new information becomes available, and previously planned actions lose validity.

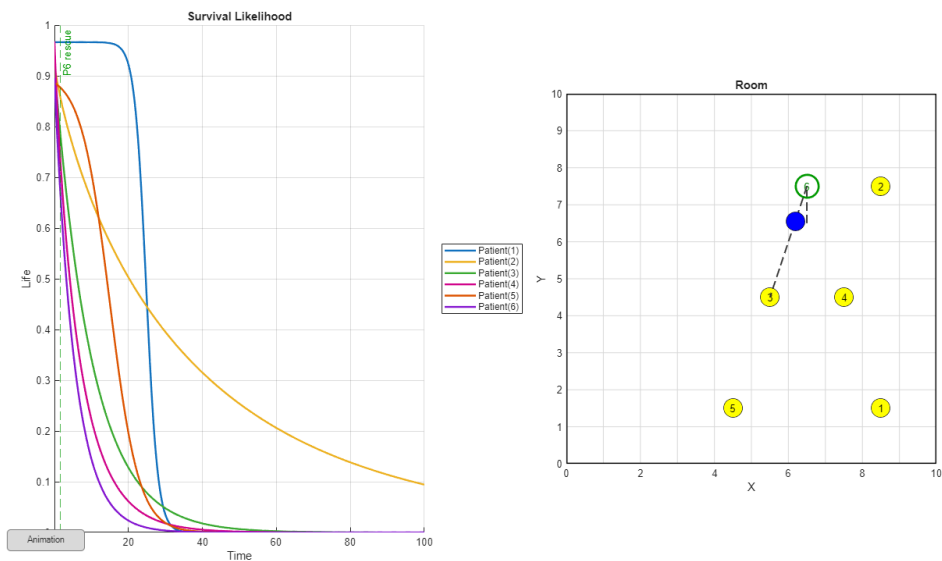
As visible in the figure 11.5, the environment is highly stochastic, so the strategy may need to change at any point during execution. Since the death time of each patient is not known in advance, this factor has a particularly strong influence on the outcome, often exceeding the effect of parameters such as robot speed or required rescue time. Although one would generally expect higher robot speed or lower rescue time to increase the number of rescued patients, the final outcome remains strongly dependent on the stochastic development of the environment, including sudden patient deaths and the appearance of new patients. The resulting rescue sequence should therefore be understood as an approximately optimal path obtained through step-by-step assessment and replanning as the timeline of the incident progressively unfolds.



Start of the mission!

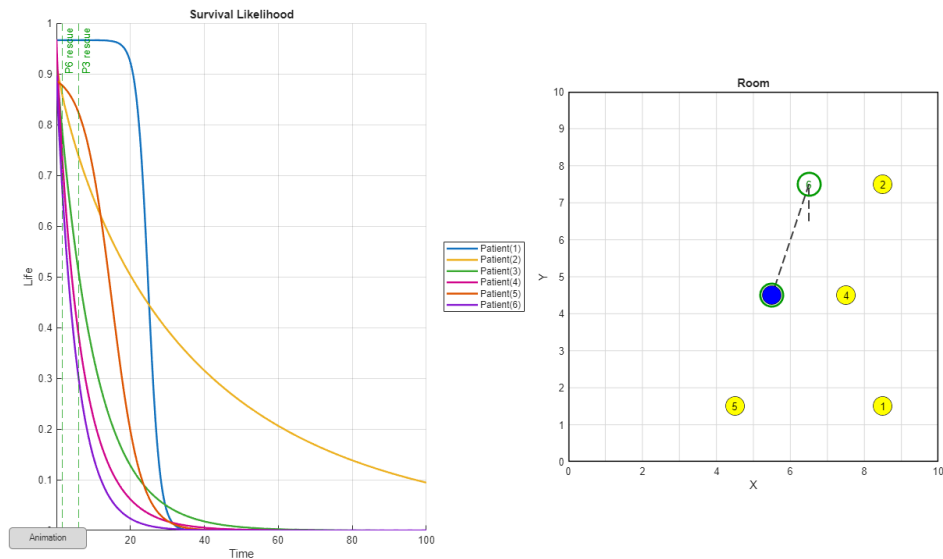


Rescued Patient #6!

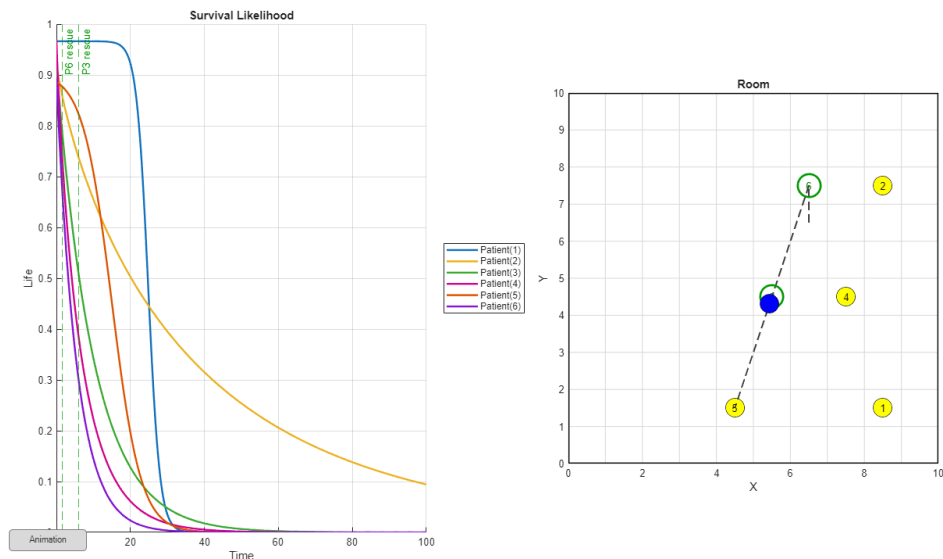


Travels to Patient #3..

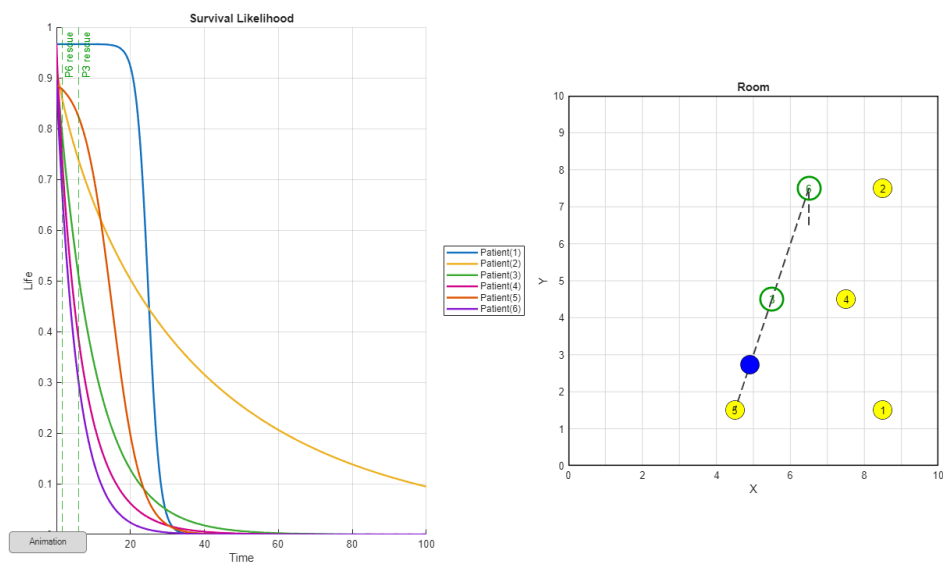
Fig. 11.4: Snapshot sequence of robot's mission in rescuing initially 6 patients.



Rescued Patient #3!

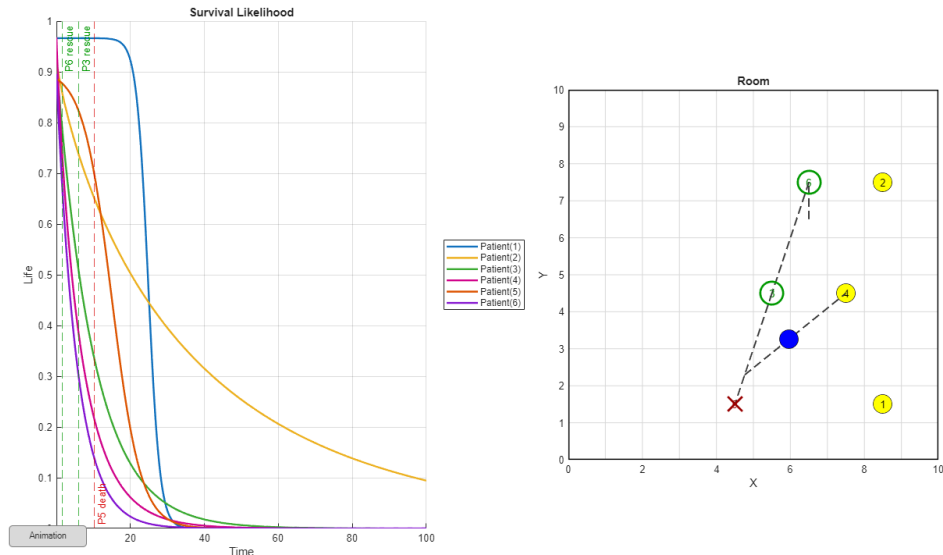


Chose Patient #5!

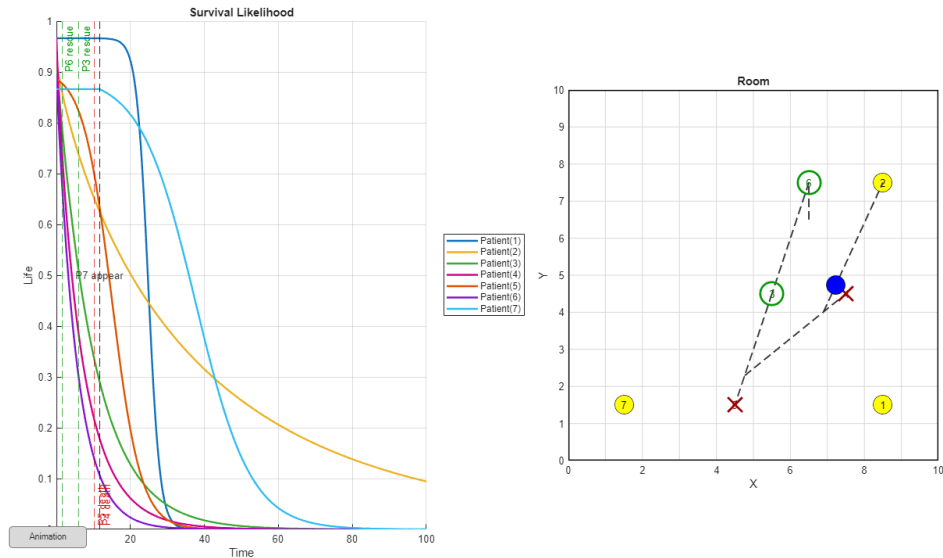


Travels to Patient #5..

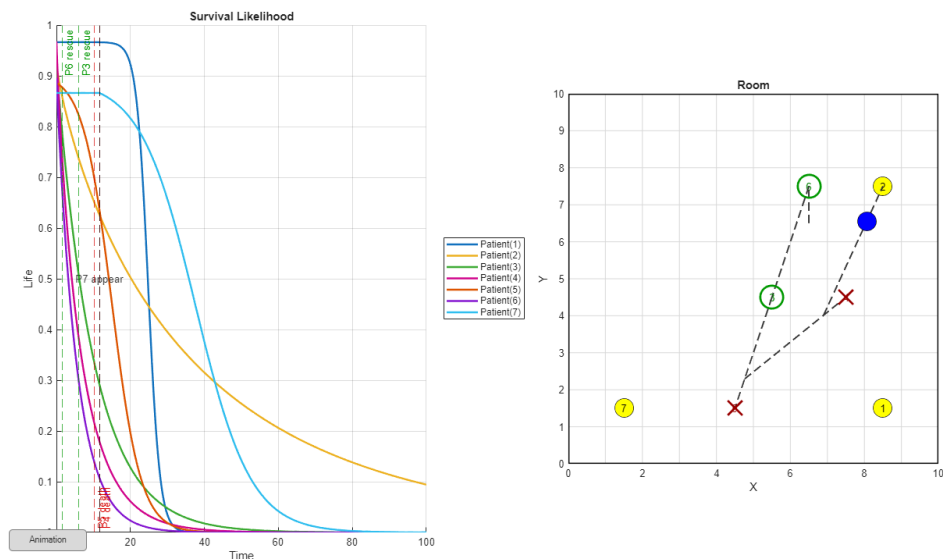
Fig. 11.4: (continued) Snapshot sequence of robot's mission in rescuing initially 6 patients.



Patient #5 died en-route! Travels to Patient #4..

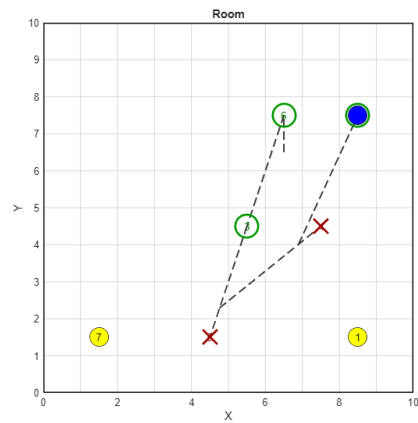
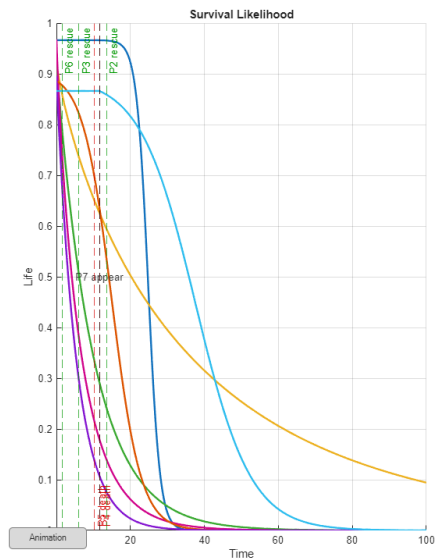


Patient #4 died en-route! Patient #7 is found! Travels to Patient #2..

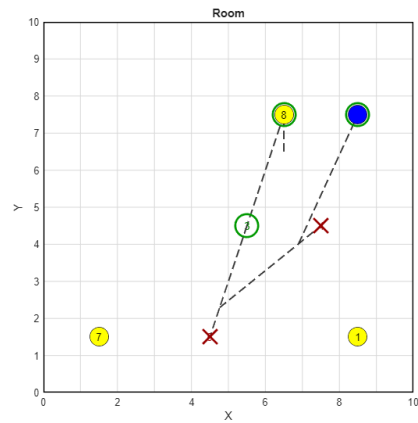
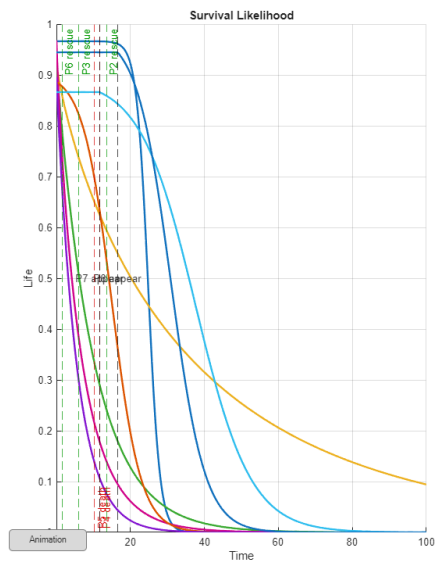


Travels to Patient #2..

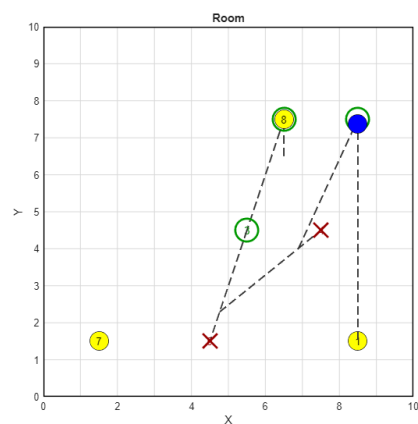
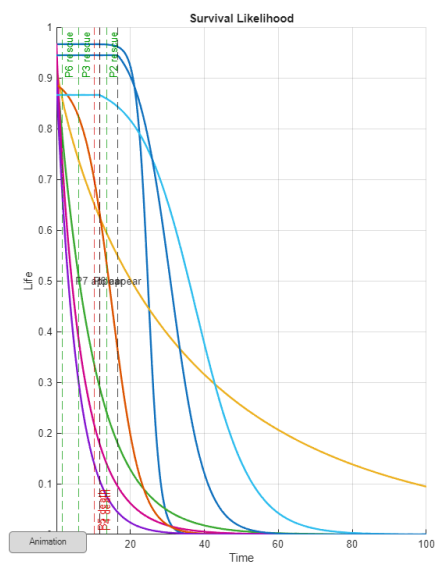
Fig. 11.4: (continued) Snapshot sequence of robot's mission in rescuing initially 6 patients.



Rescued Patient #2!

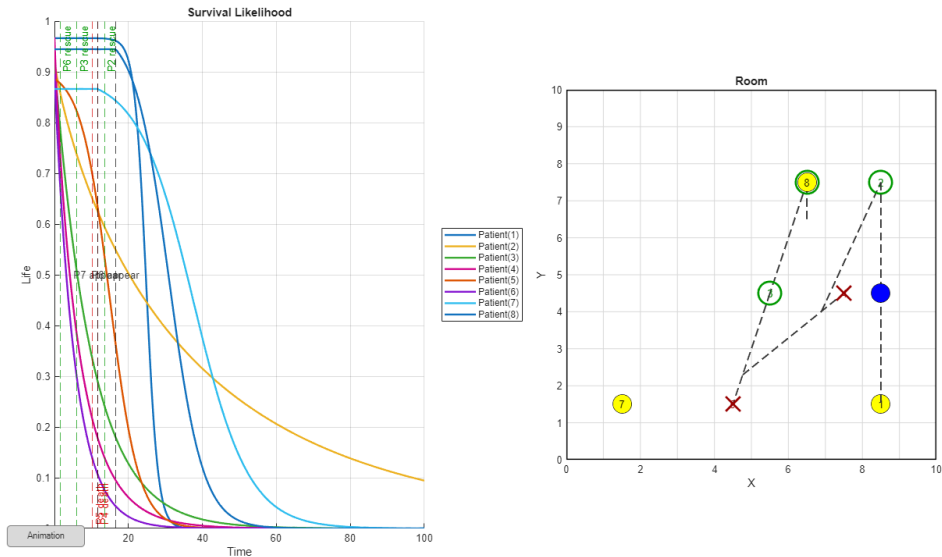


Patient #8 is found in the location of Patient #6!

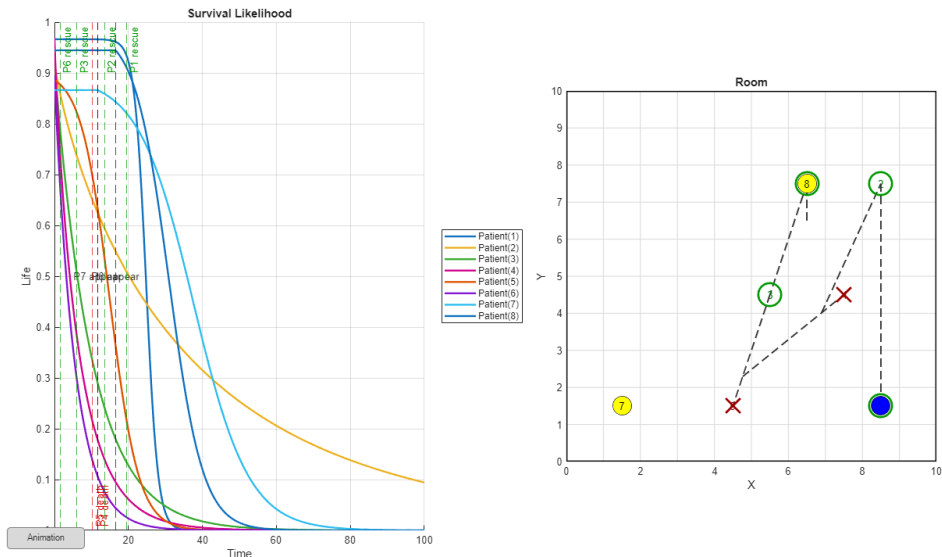


Travels to Patient #1..

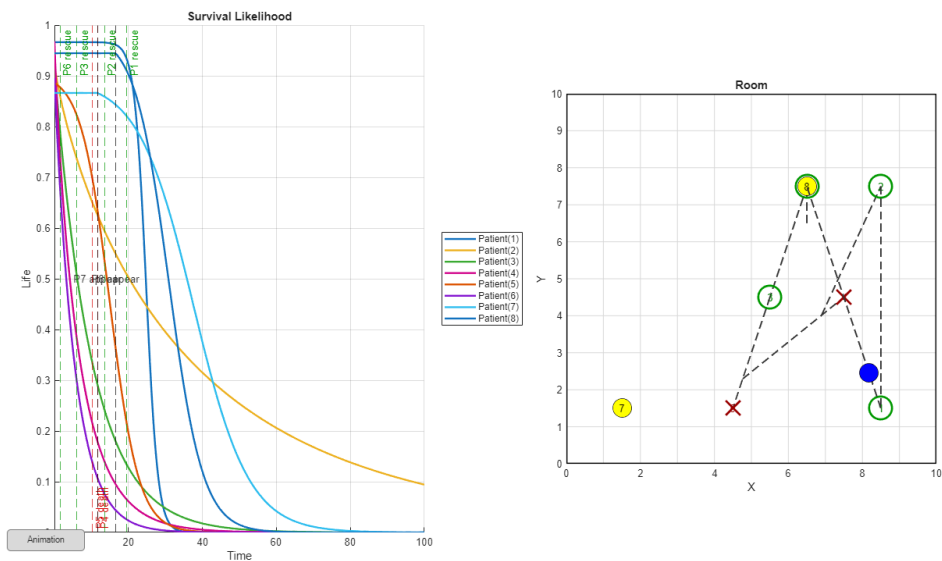
Fig. 11.4: (continued) Snapshot sequence of robot's mission in rescuing initially 6 patients.



Travels to Patient #1..

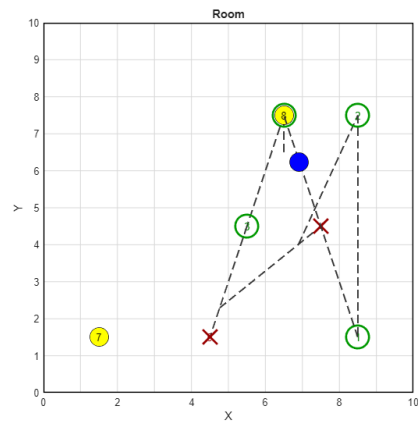
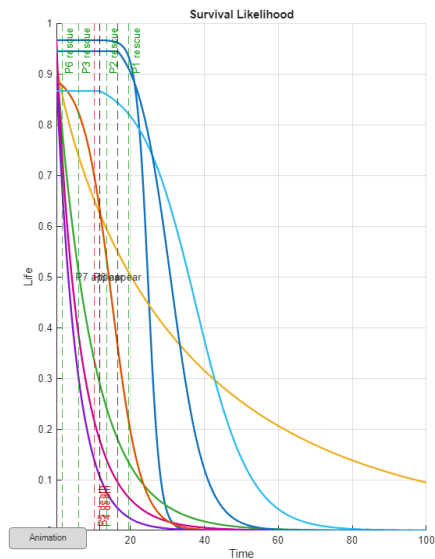


Rescued Patient #1!

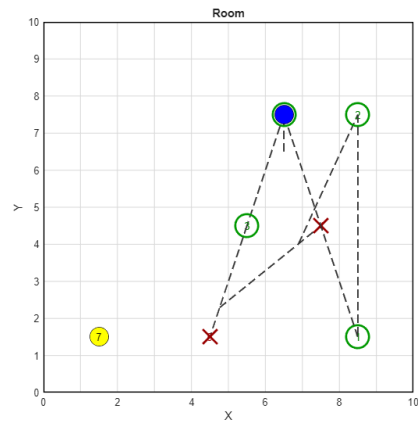
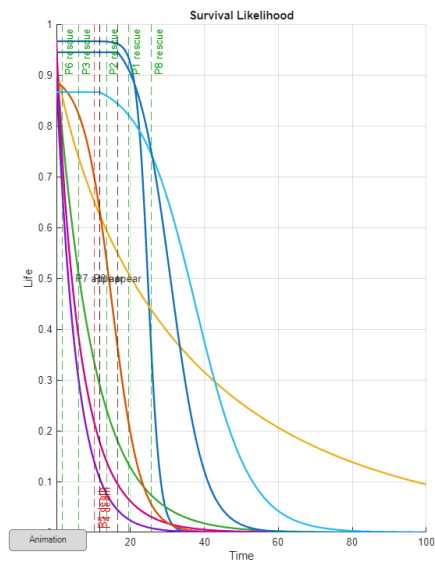


Travels to Patient #8..

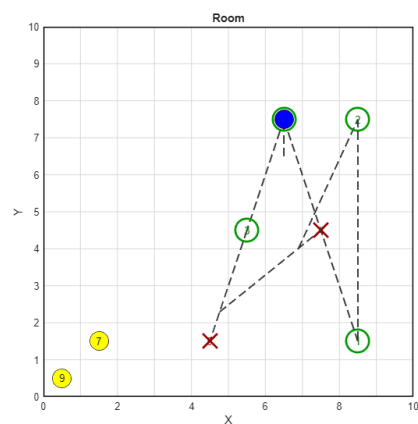
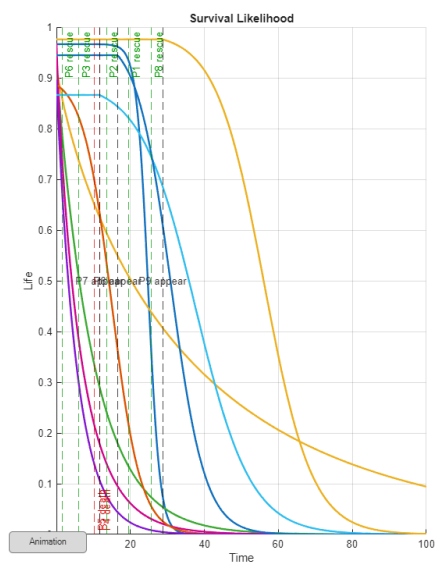
Fig. 11.4: (continued) Snapshot sequence of robot's mission in rescuing initially 6 patients.



Travels to Patient #8..

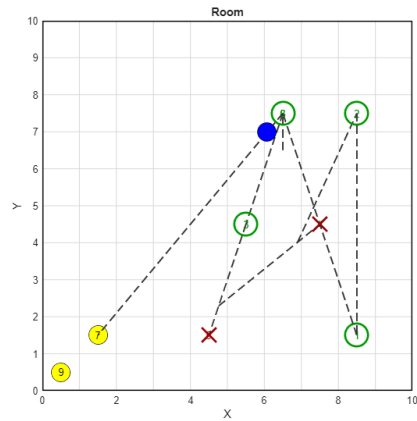
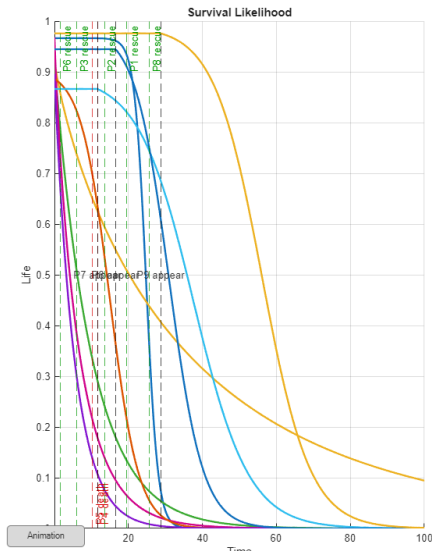


Rescued Patient #8!

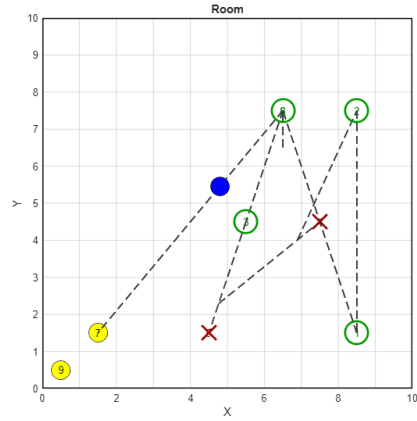
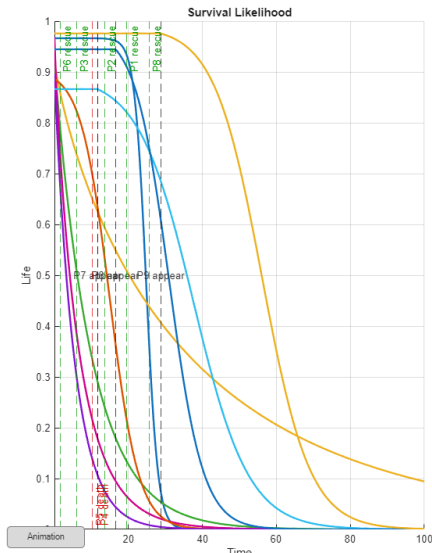


Patient #9 is found!

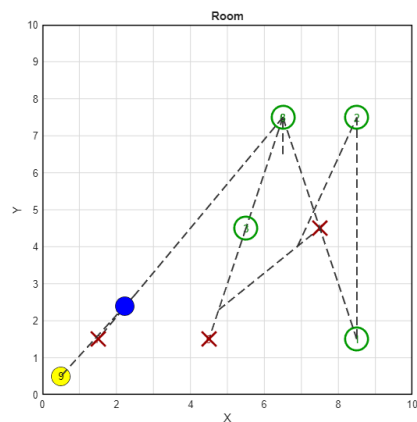
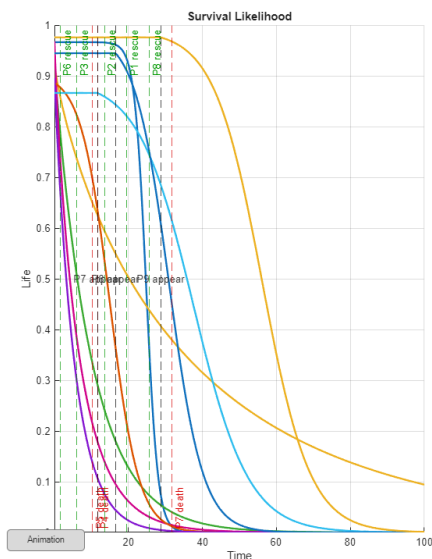
Fig. 11.4: (continued) Snapshot sequence of robot's mission in rescuing initially 6 patients.



Chose Patient #7!

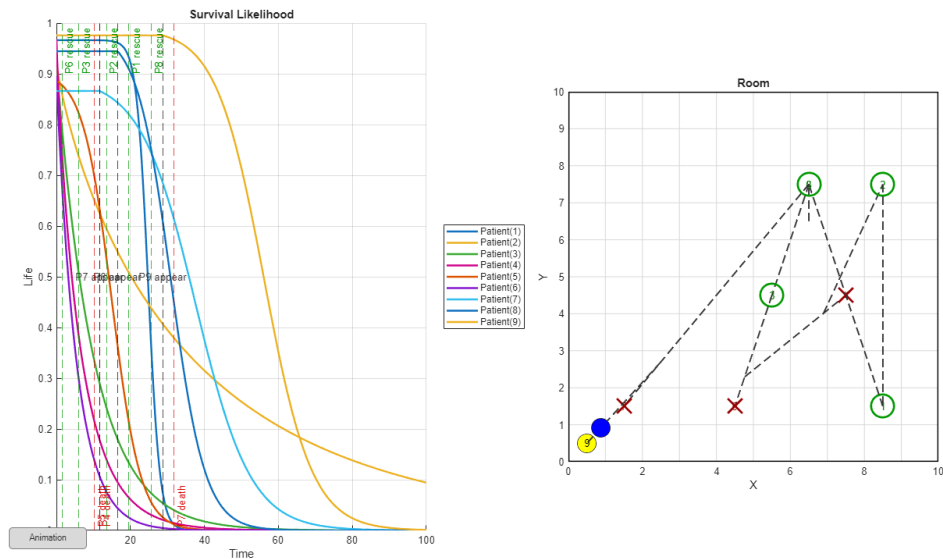


Travels to Patient #7..

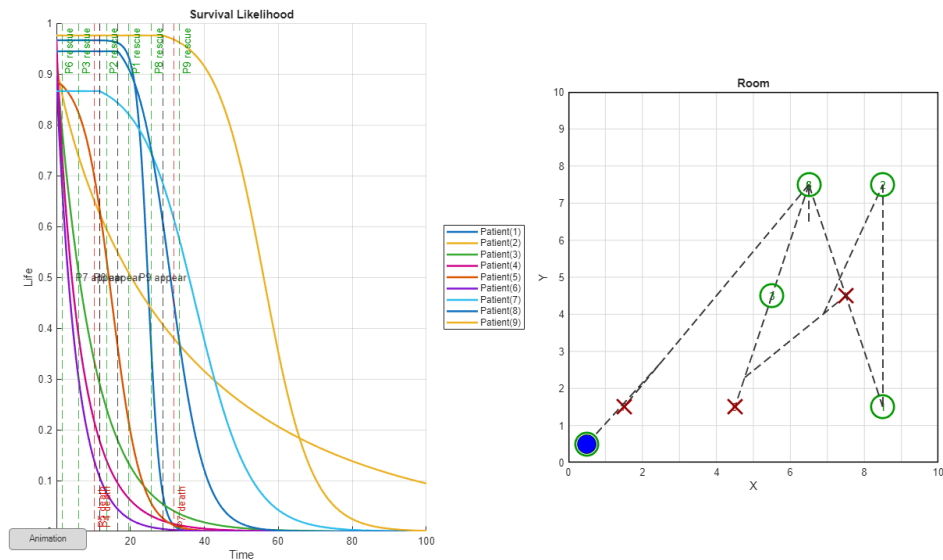


Patient #7 died en-route! Travels to Patient #9..

Fig. 11.4: (continued) Snapshot sequence of robot's mission in rescuing initially 6 patients.



Travels to Patient #9..



Rescued Patient #9! No new patient is found. Mission Completed!
Initial Patients=6, Robot Speed=2, Rescue Time=3

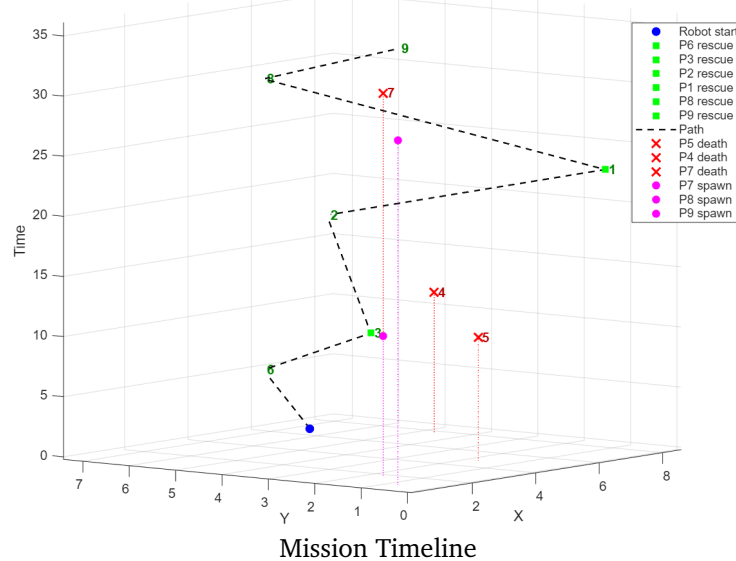


Fig. 11.4: (continued) Snapshot sequence of robot's mission in rescuing initially 6 patients.

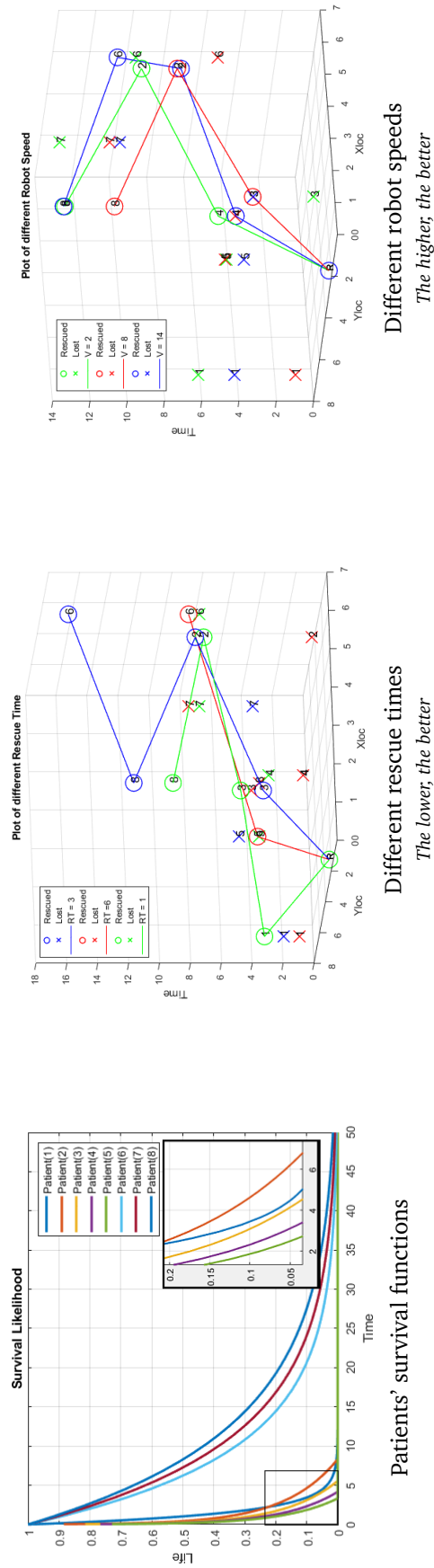


Fig. 11.5: Simulation results of robot's strategy in different rescue-time and robot-speed settings.

Algorithmic Efficiency

The rescue scenario considered here differs substantially from standard routing or scheduling settings, because victim conditions evolve over time, new patients may appear, and planning quality is meaningful only relative to the time required to obtain a usable decision. In the absence of established benchmark suite providing a direct external reference for evaluation, the proposed method is compared with a natural internal reference: the same optimization framework applied to the full patient list without severity-based categorization. This comparison isolates the contribution of the severity grouping itself and allows the effect of that design choice to be assessed directly.

At first glance, such classification may appear to come with a cost. Dividing patients into severity classes and postponing the consideration of part of the set may reduce the amount of information handled simultaneously. Therefore, it can increase the risk of losing global information that would otherwise be available when optimizing over the full list at once. In that sense, the severity-based approach accepts a controlled reduction in the simultaneity of information. However, this is exactly the trade-off under study: whether the information that may be deferred by categorization is outweighed by the computational gain obtained from reducing the effective search burden.

The main idea is that, in a time-critical rescue problem, reducing optimization time can improve overall operational performance even if the full patient set is not processed at once. In a live hazardous incident such as a fire or explosion, the environment changes so rapidly that the relevant trade-off is not only between exploration and exploitation, but more fundamentally between time and cost. Here, the cost of computation contributes directly to the cost of the mission itself, measured in human lives. Fast and reliable decisions, even if they remain valid only for a short period of time, can therefore perform significantly better than highly optimized decisions based on parameters that are no longer valid when the plan is executed.

That trade-off is especially important here because the optimization engine is a genetic algorithm, whose search quality is tightly coupled to the time available for exploration. When the full patient list is treated as one undivided set, the search space grows rapidly and the optimization becomes increasingly expensive, which reduces the ability of the planner to produce strong schedules within the short decision windows imposed by disaster response. Severity-based partitioning addresses this issue by structuring the search and thereby reducing the effective optimization effort required to obtain useful plans earlier. The time recovered in this way can then be reinvested into the mission itself, allowing earlier rescue actions,

faster reaction to newly appearing patients, and better adaptation to uncertainty during execution.

From an operational perspective, this also aligns with how real rescue teams act under pressure. In practice, emergency response rarely proceeds by treating all casualties as a single undifferentiated set. Instead, responders triage, prioritize, and stage their attention so that limited time and resources are directed where they are most consequential. The proposed severity-based variant follows the same principle in algorithmic form. The purpose of the following comparison, therefore, is not only to show that partitioning reduces computational burden, but also to examine whether this structured loss of simultaneity is compensated by improved decision timeliness and better practical rescue yield.

This comparison includes two different means of measurement and two types of horizon prediction for each algorithmic core: our categorized algorithm *Severity* versus the unpartitioned patient list *Plain*. For comparison to be fair and sound, we have only considered a fix set of patients with no new patient appearing as time passes. Additionally, all four means of measure use same seeds for the environment.

We have measured the time required for each algorithmic core to generate the next plan, in addition to the time for the whole iterative strategy to be executed. Since planning time *plan=on* required substantial time leading to many lives lost, we have shown the result of strategy-execution-only as *plan=off* where it is demonstrated that *Severity* algorithm performs similar or better (depending on the random survival functions of patients) than the *Plain* algorithm, even at this highly stochastic environment. This means that a more powerful computation capacity of the device, resulting in less planning time, bridges the gap between these two figures significantly.

Another notion used in the resulting figures is the horizon prediction: *Look up curves* versus *Estimation*. Here, we have utilized two approaches that robot uses to predict the health condition and survival probability of the patients in advance to generate the sequence of rescuing them and maximizing the expected number of patients saved.

First one, *Look up curves*, is where robot simply computes the time of arrival to each patient, looks up in the survival functions and takes the value at $T_{arrival}$ into the consideration. This approach is not time consuming to predict the probability but at the cost of not being realistic. In the real world setting, unless robot's information regarding each patient is updated through means of environment assessment, robot makes decision based on the most recent information and horizon prediction. Therefore, the algorithm should reflect the reality of uncertain information, i.e. the

survival function are the initial estimates of patients health and will not be reliable over time.

This brings us to the second approach, *Estimation*, where robot takes the current available information, extrapolates and generates an exponential function, computes the value of the resulting function as the survival probability at the $T_{arrival}$. This approach takes much higher computation time compared to the *Look up curves* approach while providing a more realistic approach of what robot is required to follow under uncertainty.

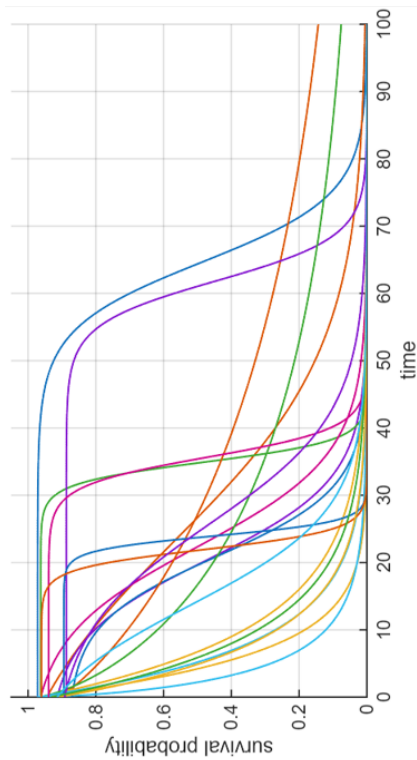
The most important outcome of the comparison across the considered criteria of: *Severity* versus *Plain*, *Planning time included* versus *strategy-execution-only*, and *Lookup curves* versus *Estimation*, is the computational time required by the algorithm. Across both time measurement settings, the *Severity* variant outperforms the *Plain* variant while requiring substantially less total computation time. When computation time required for planning is included, the total time may look surprisingly less than the other variant. The logical explanation hidden in the figures is the fact that, when it takes longer for the robot to plan and compute the next action, less patients survive to be rescued eventually. The wall time (computation plus execution time) therefore corresponds to the number of patients rescued.

Consequently, the reduction in computation time is particularly crucial in the current rescue mission setting, since saved computational effort can be redirected to the mission itself. In practice, this means that more time remains available for handling uncertainty during execution, reacting to newly appearing patients, and increasing the overall rescue yield under dynamic conditions. Ultimately, these results provide clear evidence that the proposed algorithm improves the efficiency of strategic planning in stochastic and time-critical environments.

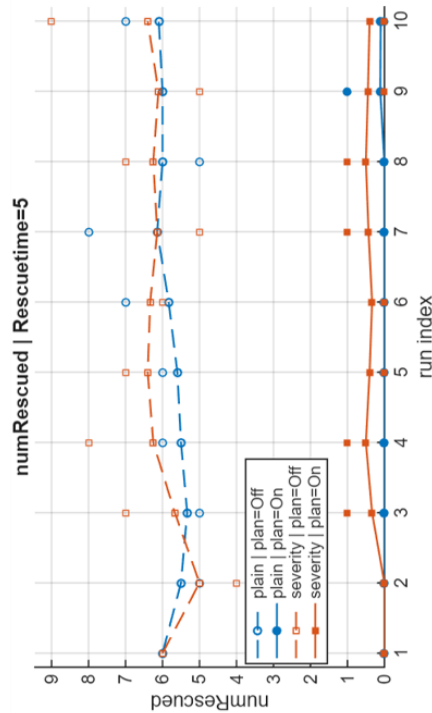
The results are presented in Figures 11.6 to 11.13 for different rescue-time and robot-speed settings, using two independently generated sets of survival functions, each associated with its own patient placement configuration, and evaluated over multiple runs to capture stochastic variability. The reported measures include the number of rescued patients across different combinations of robot speed and rescue time, together with the computational time required for planning and the total execution time.

11.3 Limitations & Considerations

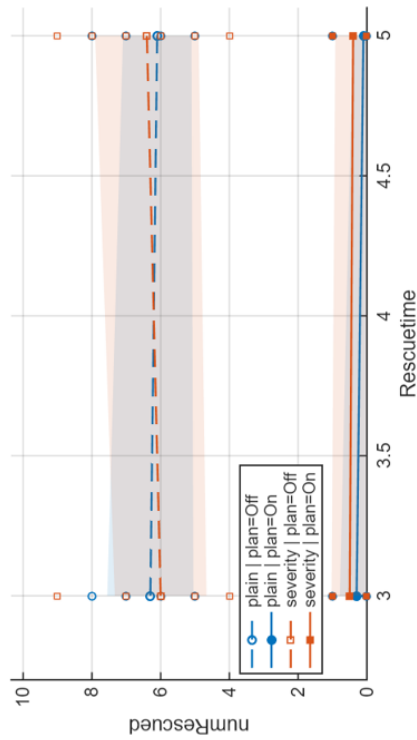
Although an optimal strategy may still be well-defined with respect to a fixed model or an expected-reward objective, there is no guarantee that a plan computed from



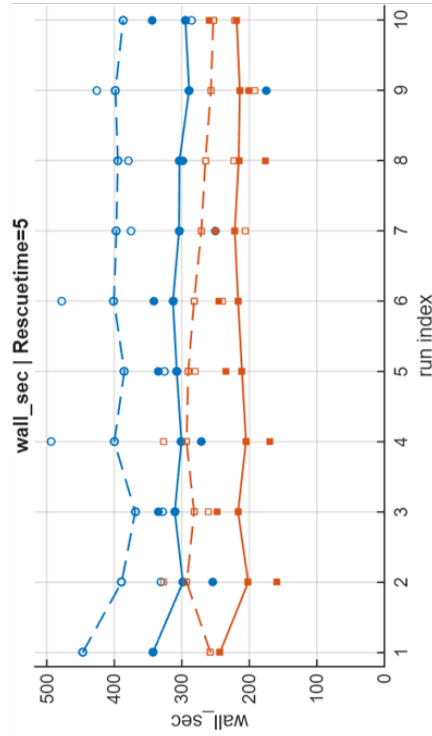
Patients' survival functions



Number of Rescued Patients at rescue time=5 over different runs
The higher, the better

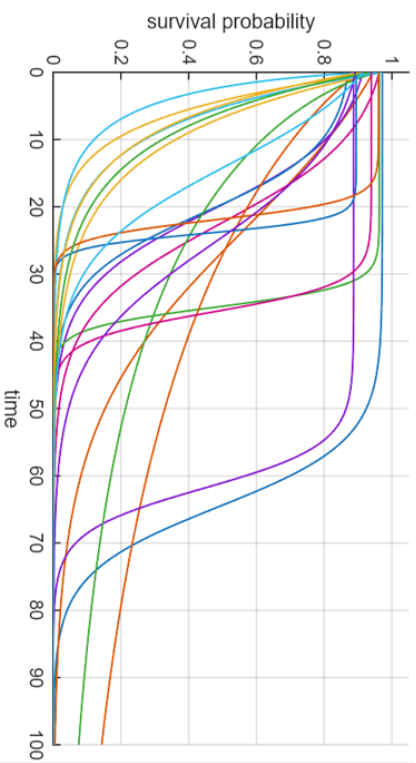


Number of Rescued Patients at different rescuetime=3 and 5
The higher, the better

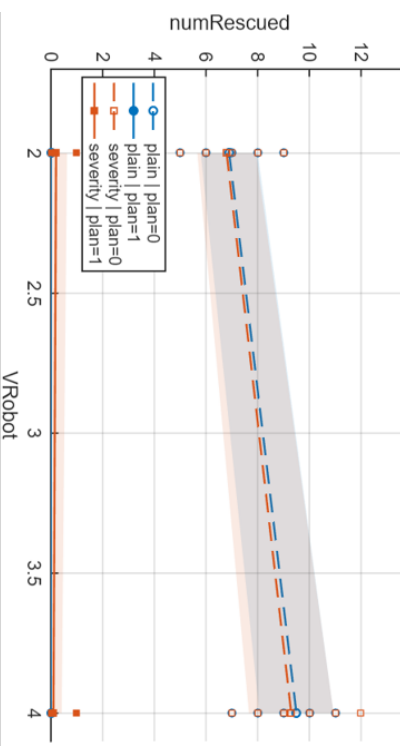


Computation plus execution time (wall) over different runs
The lower, the better

Fig. 11.6: Performance evaluation of Severity algorithm using *Lookup Curves* approach at different survival functions (setting #1) and various rescue times.

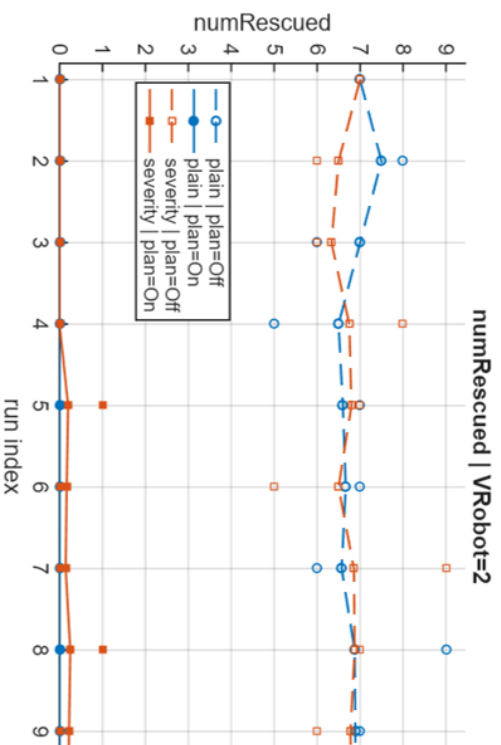


Patients' survival functions



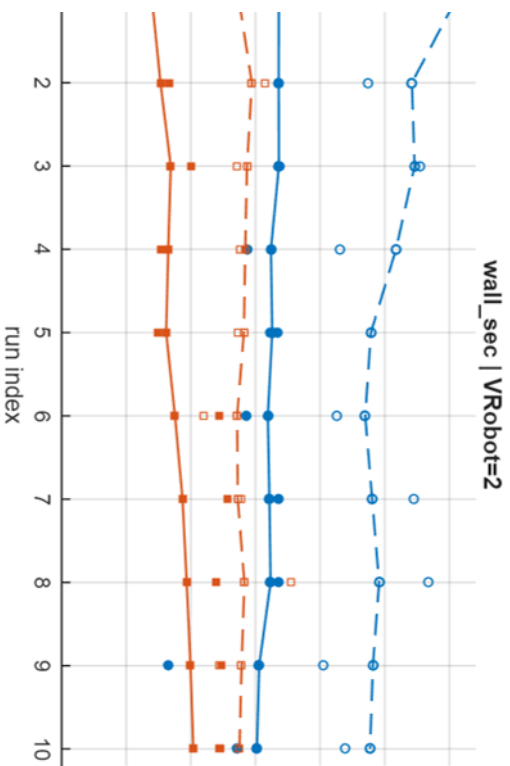
Number of Rescued Patients at different RobotSpeed=2 and 4

The higher, the better



Number of Rescued Patients at RobotSpeed=2 over different runs

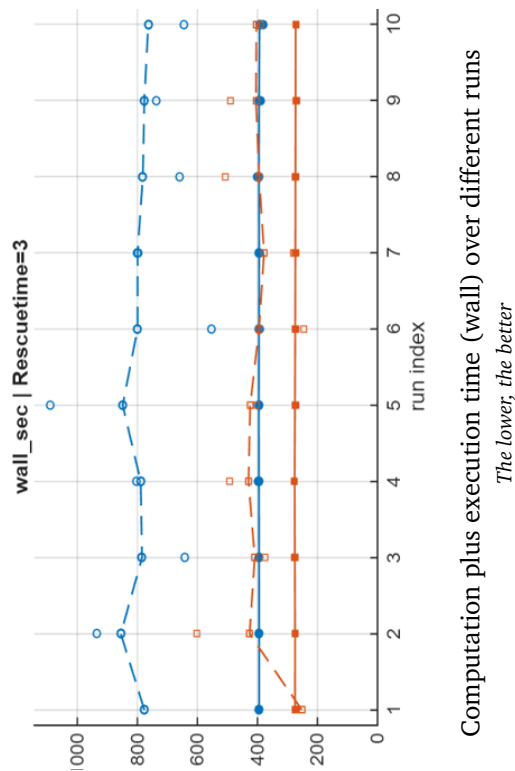
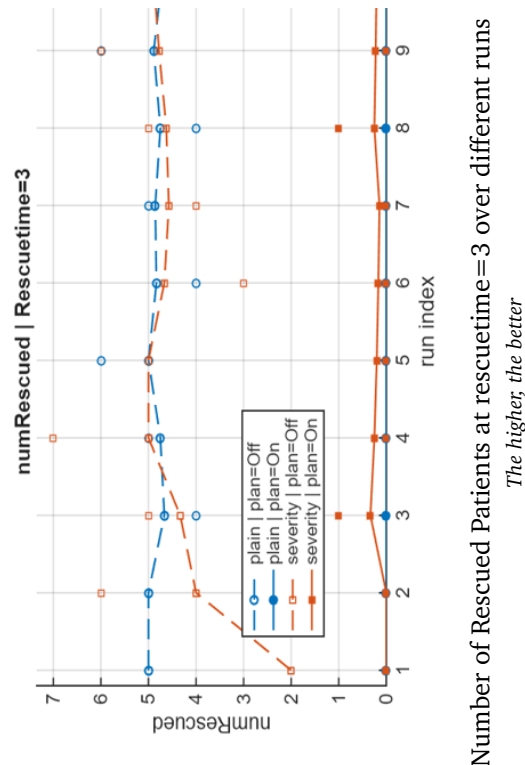
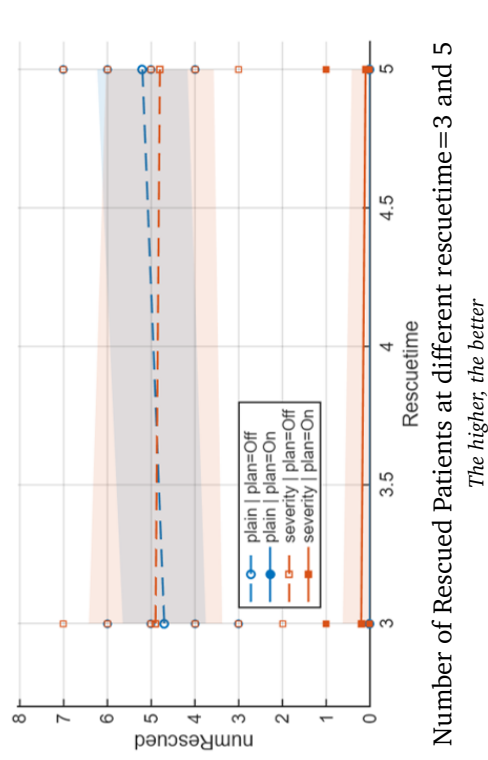
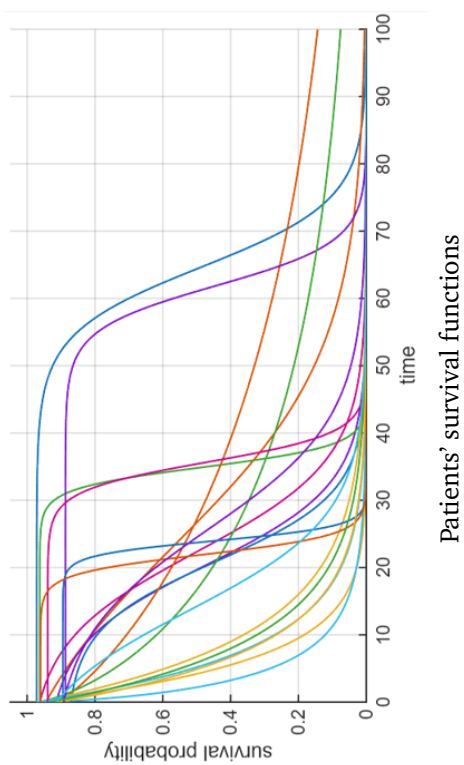
The higher, the better



Computation plus execution time (wall) over different runs

The lower, the better

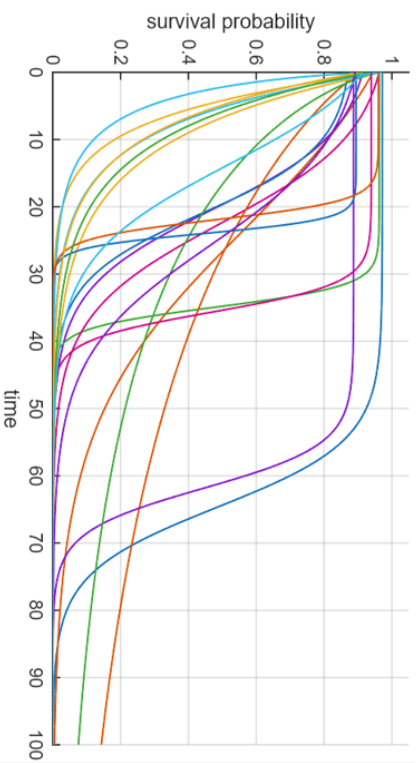
Fig. 11.7: Performance evaluation of Severity algorithm using *Lookup Curves* approach at different survival functions (setting #1) and various robot speeds.



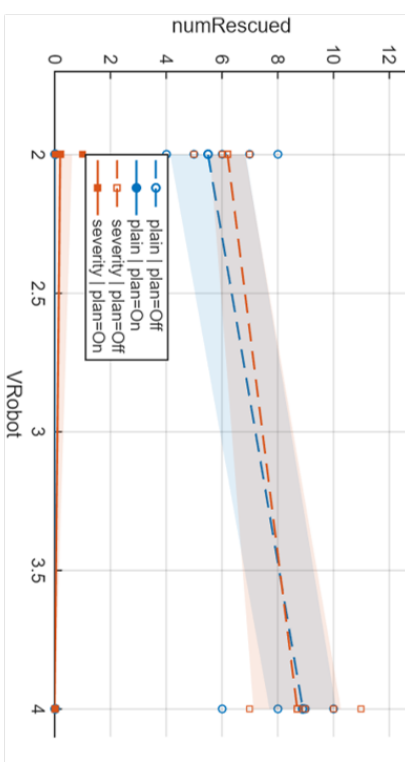
Number of Rescued Patients at rescuetime=3 over different runs

Computation plus execution time (wall) over different runs

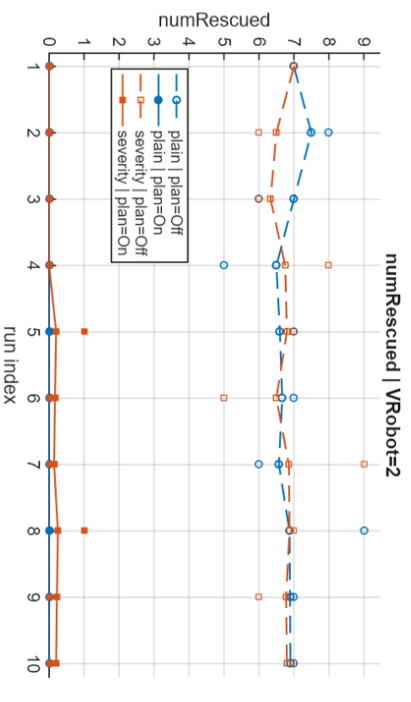
Fig. 11.8: Performance evaluation of Severity algorithm using *Estimation* approach at different survival functions (setting #1) and various rescue times.



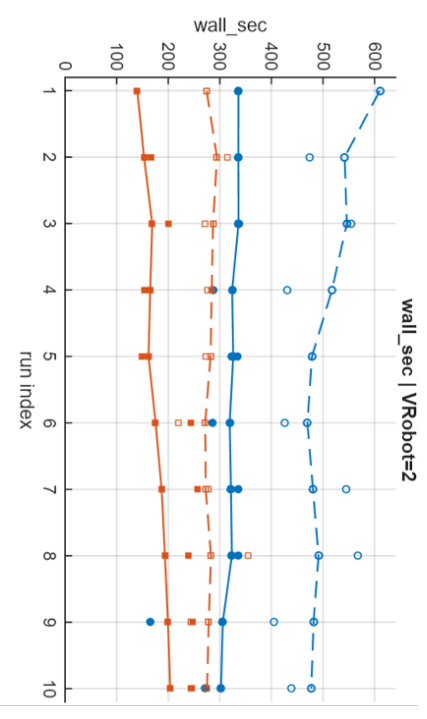
Patients' survival functions



Number of Rescued Patients at different RobotSpeed=2 and 4
The higher, the better

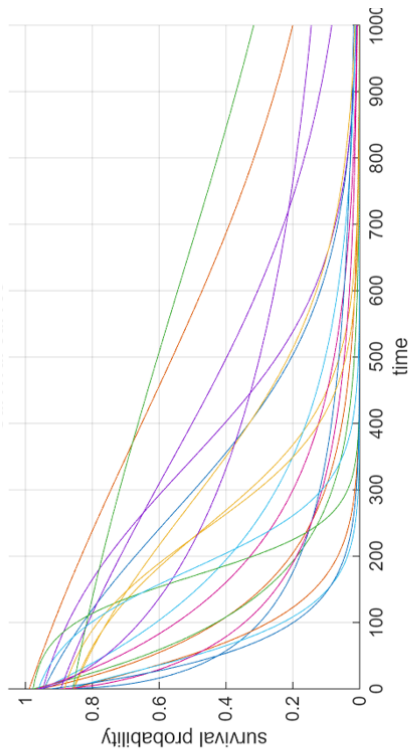


Number of Rescued Patients at RobotSpeed=2 over different runs
The higher, the better

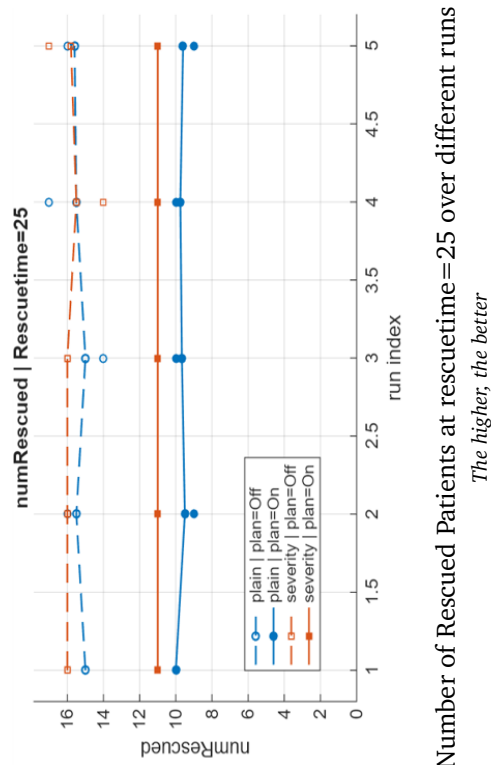


Computation plus execution time (wall) over different runs
The lower, the better

Fig. 11.9: Performance evaluation of *Severity* algorithm using *Estimation* approach at different survival functions (setting #1) and various robot speed.



Patients' survival functions



Number of Rescued Patients at rescutime=25 over different runs

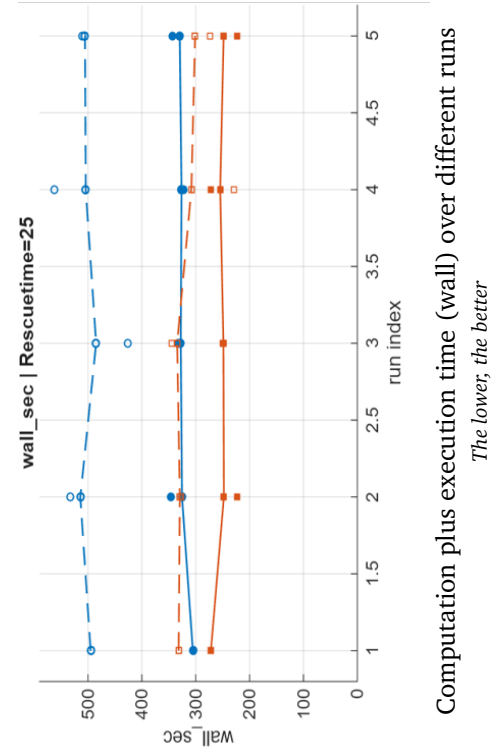
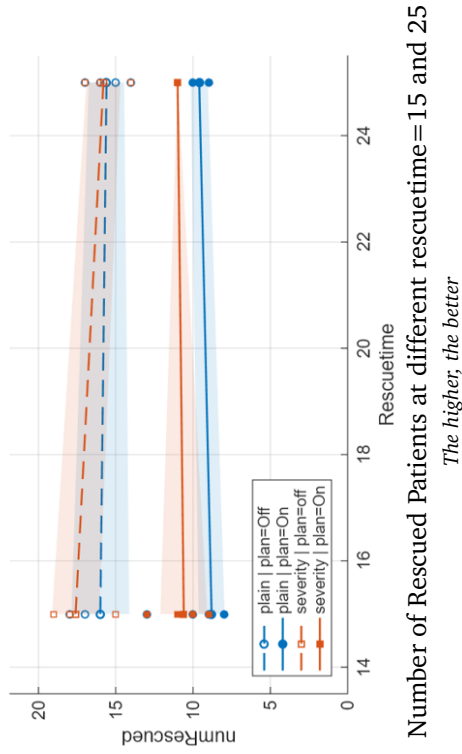
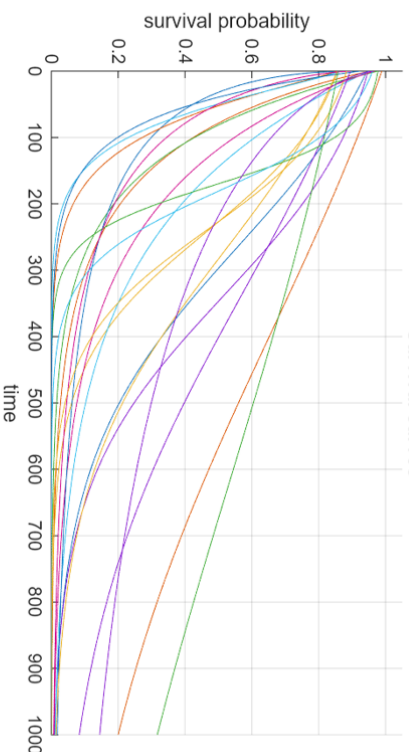
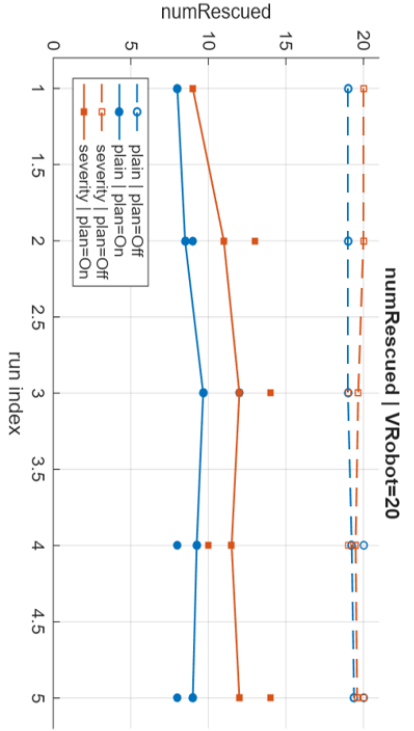


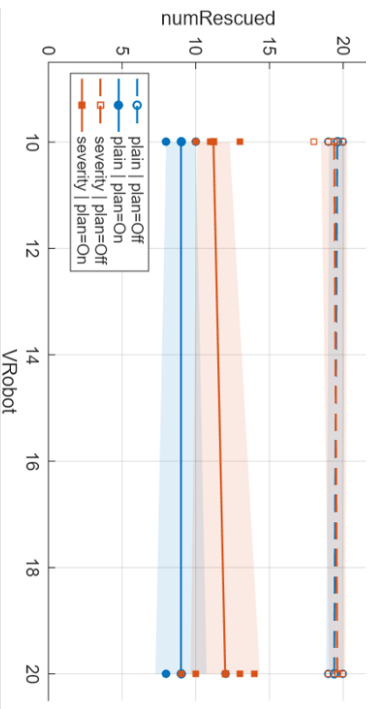
Fig. 11.10: Performance evaluation of Severity algorithm using *Lookup Curves* approach at different survival functions (setting #2) and various rescue times.



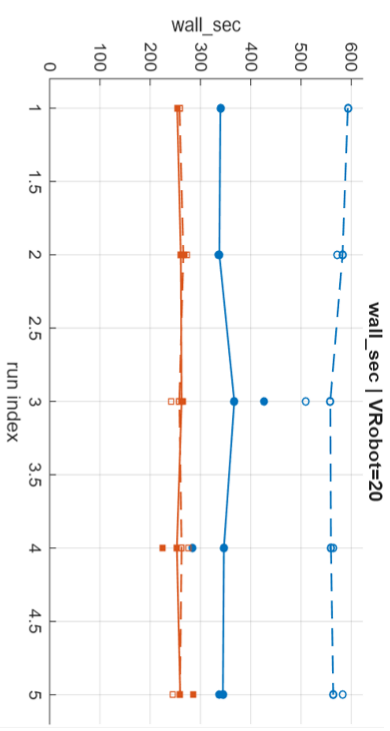
Patients' survival functions



Number of Rescued Patients at RobotSpeed=20 over different runs
The higher, the better



Number of Rescued Patients at different RobotSpeed=10 and 20
The higher, the better



Computation plus execution time (wall) over different runs
The lower, the better

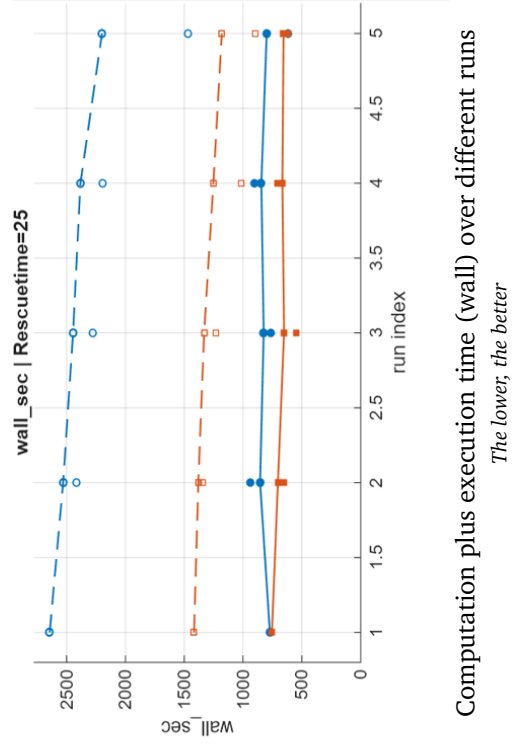
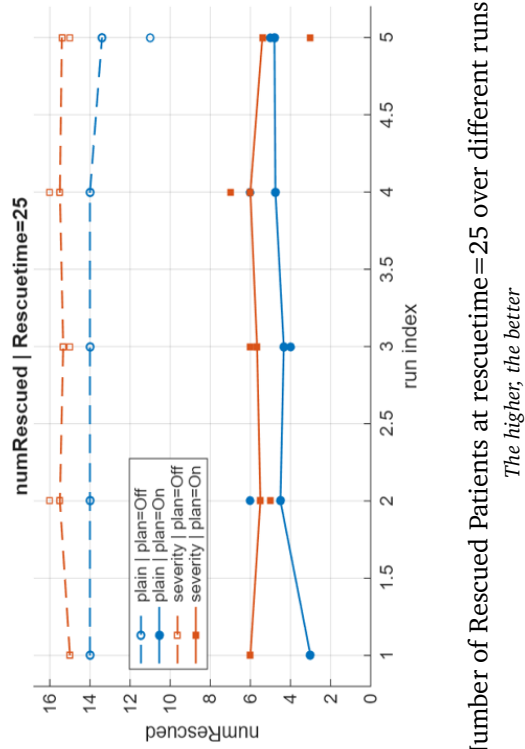
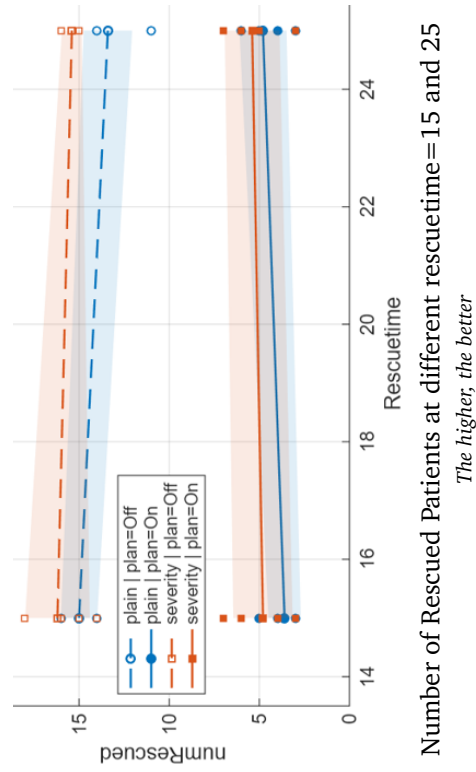
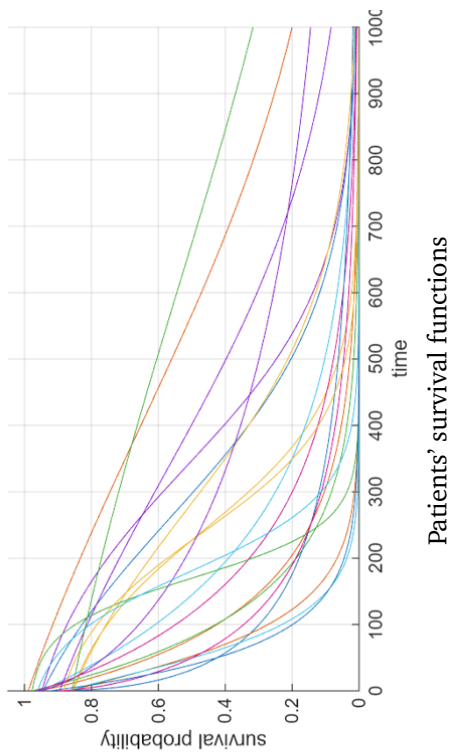
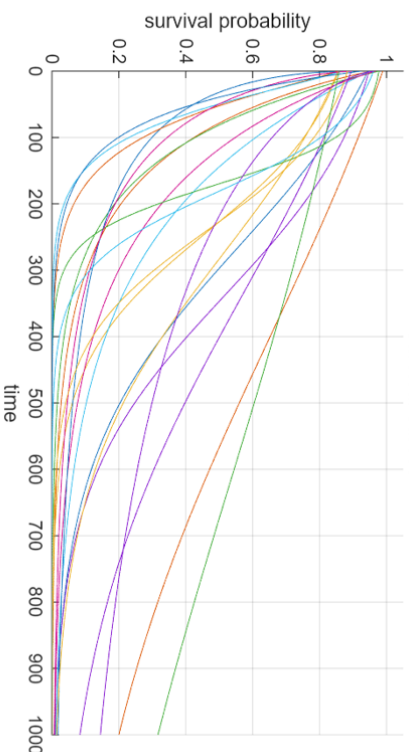
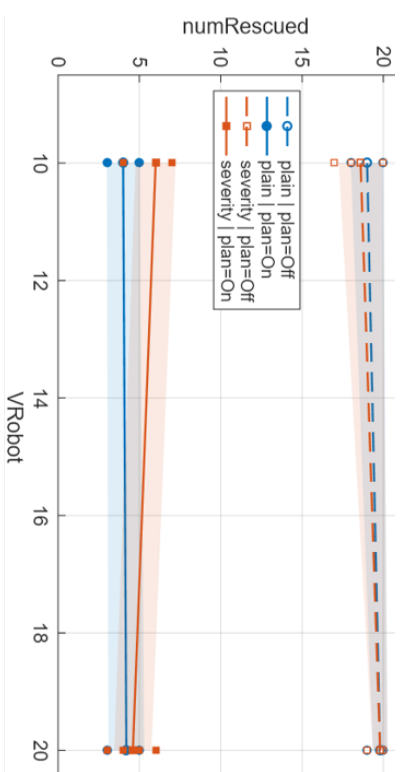


Fig. 11.12: Performance evaluation of *Severity* algorithm using *Estimation* approach at different survival functions (setting #2) and various rescue times.

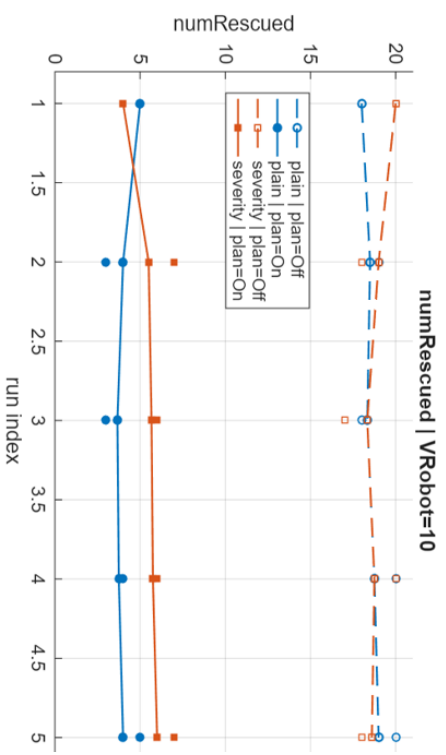


Patients' survival functions



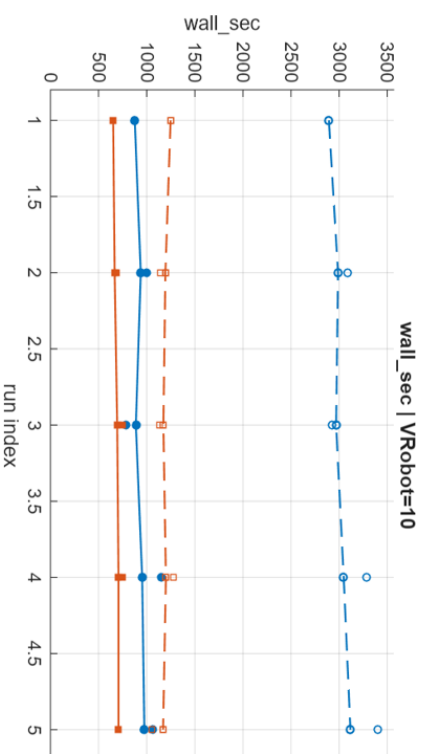
Number of Rescued Patients at different RobotSpeed=2 and 4

The higher, the better



Number of Rescued Patients at RobotSpeed=2 over different runs

The higher, the better



Computation plus execution time (wall) over different runs

The lower, the better

Fig. 11.13: Performance evaluation of Severity algorithm using Estimation approach at different survival functions (setting #2) and various robot speeds.

the current information will remain optimal as the world keeps changing underneath the planner. In life-critical missions, however, a good plan delivered on time usually outweighs a better plan delivered too late. Our approach therefore favors anytime, near-optimal decisions under explicit compute budgets and graceful degradation, rather than pursuing exact optimality that may no longer be relevant by the time the computation finishes.

Genetic Algorithms (GAs) are a natural fit for complex, time-dependent objectives, but they typically need multiple generations to converge. Computational cost grows with both the number of patients (nodes) and the population size; at larger scales (hundreds or thousands of nodes), real-time re-optimization becomes impractical without additional heuristics, specialized operators, and parallel execution. Frequent environmental changes further amplify cost: if re-planning is triggered too often, naive full re-evolution will miss deadlines. Practically, we rely on warm starts, elitism, incremental fitness updates, and bounded-generation “anytime” cycles (optionally GPU/multi-core) to keep latency under control.

Partial observability introduces another layer of difficulty. When a patient’s location is modeled as a probability distribution rather than a known point, fitness evaluation depends on expected travel time/expected distance to a distribution and on expected survival, not simple point-to-point metrics. This significantly increases evaluation complexity. To remain responsive, the GA must integrate sensor updates rapidly (e.g., streaming observations from robots/teams) and replan online with small populations, parallel fitness computation, and event-triggered updates that touch only affected route segments.

Real deployments also demand multi-robot coordination and heterogeneity. Some robots move faster but carry little; others move slower but bring medical supplies or special capabilities (e.g., stabilization, lifting). The problem becomes a coupled task-allocation and routing problem. A GA representation can encode robot–task assignment (with permissible overlap when multiple robots are capable) and route order jointly, but this coupling increases search dimensionality. A pragmatic remedy is a hybrid hierarchy: lightweight allocation (markets/consensus or priority rules) at the team level, with GA-based rescue scheduling per robot, synchronized periodically to avoid conflicts and congestion.

When patient locations are highly uncertain, the problem partially shifts from routing to search. In such cases, a hybrid scheme is appropriate: a search phase that explores high-probability zones to confirm/deny presence, interleaved with rescue execution once a patient is localized. The GA can then focus on the rescue-scheduling core, ordering and assigning confirmed patients, while a complementary search policy updates the belief maps that feed the GA’s fitness function.

Finally, our baseline assumes reasonably accurate travel times and a static set of constraints, which is rarely the case in disasters (blocked roads, collapses, emergent hazards, new casualty reports). These assumptions are a starting point, not a limitation of principle. Each can be relaxed or extended: robust travel-time models and stochastic edges, dynamic no-go zones and safety buffers, late-arrival penalties and triage constraints, and distributional patient models that are continuously refined by incoming data. With these mechanisms, the framework remains flexible enough to operate under dynamic, uncertain conditions while acknowledging that the objective is timely, risk-aware, near-optimal performance, not unattainable global optimality.

From this foundation, it becomes feasible to incrementally introduce new complexity and uncertainty levels. The essence of the solution, a modified GA-based control algorithm that maps a route-encoding to a time-sensitive fitness function, remains highly adaptable to those expanded problem definitions.

In summary, the modified-GA approach is effective for simulating and solving time-critical rescue routing in a simplified setting; However, direct, out-of-the-box deployment is constrained by real-world factors. The next stage includes multiple robots, dynamic events, multi-objective constraints which will be significantly more complex. Our primary objective was to establish a core solution for single-robot multiple-patients rescue scheduling, focusing on solidifying the exponential survival model, refining GA operators and tuning, demonstrating the feasibility of the approach in a clean setting.

Synopsis

Lessons, Limitations, Horizons

Recapitulation & Outlook

” *Human beings, viewed as behaving systems, are quite simple. The apparent complexity of our behavior over time is largely a reflection of the complexity of the environment in which we find ourselves.*

— **Herbert A. Simon**
The Sciences of the Artificial

12.1 Back to the Disaster Scene

Returning to the tragedy of our disaster incident through the lens of our model, the scene is no longer a blur of contingencies but a system with explicit states (Search, Travel, Rescue) and clear transition guards. Hazards arrive as updates to the survival curves, each associated with a limited validity window; data flow in priority lanes, so we know what matters now and what can wait without shifting the strategy. A robot on this scaffold prioritizes correctly; it interrupts only on material deltas, defers low-impact drift, and re-plans with IMCC-aware planning exactly when the thresholds say it should.

In practice this means fewer stalls at choke points and a steady focus on victims who have a chance to be saved. Against deterministic heuristics and static planners, the same hardware rescues more people precisely because the architecture and the algorithm agrees on when to sense, when to compute, and when to move. Seen this way, the vignette is not just a story; it is a worked example of how explicit boundaries, states and transitions, and hazard-aware priorities turn a demonstrator into a credible path to field use, setting the stage for the contributions that follow.

12.2 Synthesis of Contributions

Disaster environments are adversarial by nature: multi-hazard, time-critical, information -poor, and relentlessly non-stationary. Therefore, the problem of autonomous

rescue in dynamic disaster scenes is both operationally critical and formally challenging. Its complexity does not stem only from the number of parameters or the breadth of the state space; it also stems from the continuous change of those parameters, and in some cases from the change in the number of relevant entities over time.

Against this backdrop, robotic rescue is more than a challenging control problem. It is an open-ended strategic planning problem in which uncertainty and dynamics are central components of the decision process. In such a setting, several perspectives may legitimately appear to offer a viable way into the problem, while the central difficulty lies in determining which viewpoint, which parameters, and which sequence of actions preserve the mechanisms that lead to the best attainable outcome: maximizing the expected number of lives saved under changing conditions.

This dissertation turns this problem –autonomous rescue missions in dynamic, multi-hazard disasters– into a tractable, testable decision framework end-to-end: from modeling and parameterization, through formal analysis and verification attempts on small instances, to an optimization engine that produces defensible sequences fast enough to matter. What follows is a consolidated account of what was designed, analyzed, estimated, verified, and engineered.

12.2.1 From Disaster Response to a Formal Decision Problem

The work of this dissertation originates from the broader problem of strategic planning in disaster response. In such environments, the central question is how a sequence of actions should be selected when the conditions that determine the value of those actions evolve during the mission itself. The rescue mission studied throughout this thesis is a concrete instance of this broader problem: a setting in which victims are exposed to time-dependent hazards, rescue operations consume critical time, information may change as the mission unfolds, and the planner must continuously relate present decisions to their future consequences.

At this stage, the contribution is to translate that planning problem from an operational description into a formal object: to determine which system features must be retained so that strategies remain meaningful when information, risks, and feasible actions evolve during execution.

The analysis extracted the mechanisms that make this class of planning problems formally demanding. The environment evolves stochastically and the intensity of critical events changes with time. Several entities remain active simultaneously where their actions and observations affect the state of other components; and the progress of the mission depends on time quantities that must be measured, compared,

and reset. These mechanisms define the modeling requirements of the framework: stochastic evolution, time-dependent rates, concurrent agents, interaction, and explicit clock handling.

These requirements also clarify why the planning problem cannot be evaluated through a static objective alone. The value of a strategy depends on the temporal state in which its decisions are executed, on the stochastic events that may occur before execution is completed, and on the way interactions update the remaining decision space. Consequently, the first major step was to examine whether existing stochastic, timed, probabilistic, and interactive formalisms could provide a common semantic ground for this type of strategic planning problem.

This examination, therefore, revealed a modeling gap. Existing formalisms are highly expressive within their respective domains, yet the rescue problem requires the temporal, stochastic, interactive, and strategic dimensions of the mission to be represented and analyzed within one coherent framework. This gap motivated the introduction of a model class capable of preserving the essential structure of time-critical rescue planning while retaining the non-stationary dynamics of the problem.

12.2.2 A Unified Model of Time-Critical Rescue

The modelling requirements extracted above led to the formal contribution of the thesis: the construction of a model class for time-critical rescue planning. The aim was to obtain a formal structure in which the evolution of risk, the passage of time, the interaction between components, and the strategic resolution of choices could be studied together.

Within the rescue setting considered in this thesis, these requirements become concrete through the survival dynamics of victims and the time-consuming actions of rescue agents. Hazards are represented by time-varying survival functions, which contribute to the quantities of *expected number of rescued patients*, against which planning decisions are evaluated. Each victim carries a survival curve whose behavior reflects the relevant hazard and medical state, while the slope of that curve determines how urgently the victim's state is deteriorating. Thus, the model does not treat rescue targets as static tasks, but as time-dependent entities whose expected contribution to the objective changes during the mission.

To capture this structure, we introduced *Interactive Markov Chains with Clocks*. IMCCs combine the interaction-oriented structure of Interactive Markov Chains

with the clock-based timing discipline of Timed Automata, and extend the stochastic component by allowing autonomous transition rates to depend on clock valuations.

Model Formulation

Formally, the rescue scene is represented as a composed state-transition system whose states encode locations, agents, clock valuations, and mission status. Controllable transitions correspond to strategic choices, such as assigning a rescue agent to a patient or initiating a rescue action. Autonomous stochastic transitions represent events generated by the environment or by the deterioration of patients. Interactive transitions capture mission-level updates, including successful treatment, cancellation of a rescue attempt, or the removal of a patient from the active set. In this way, the model provides a common semantic ground for both the formal analysis of strategies and the later simulation-based evaluation of planning algorithms.

The contribution of this stage is therefore not only the definition of a new model class, but the placement of the disaster-response planning problem inside a formal framework where its essential mechanisms remain explicit. This establishes the foundation on which the later questions of optimality, decidability, approximation, and implementability can be addressed.

12.2.3 Theoretical Boundaries and Approximation Guarantees

The introduction of IMCCs made it possible to formulate the rescue-planning problem with the required semantic precision. Since the model combines timed behavior, stochastic events, clock-dependent rates, interaction, and strategic choices, the next question concerns its theoretical status: whether optimal strategies can be computed, and under which restrictions such computation becomes possible. This question is essential as the usefulness of the model depends not only on its expressive power, but also on the extent to which strategies can be synthesized, compared, and justified.

This thesis therefore examined the theoretical analysis of strategy optimality in IMCCs. We determined that in the unrestricted model, where clocks and horizons are not bounded, exact strategy optimality is not decidable in general. The expressive combination that makes IMCCs suitable for dynamic disaster-response planning also makes optimal strategy requiring restrictions before exact computation can be recovered.

The positive result is obtained by moving to bounded formulations. When the clock domain and mission horizon are restricted, and when the continuous-time behavior

is approximated through a finite granularity, the original problem gives rise to a finite decision model. In this setting, the strategy-synthesis problem becomes tractable in the formal sense: the finite-horizon approximation can be solved by backward induction, yielding an optimal strategy for the approximated structure.

A further contribution is the treatment of the approximation error. The deviation of the discretized model is analyzed explicitly by estimating the accumulated error along the decision process. This thesis, therefore, derives bounds on the loss introduced by the approximation and thereby relates the computed strategy to the continuous-time target. This makes it possible to discuss near-optimality with a quantified margin rather than as an informal heuristic claim.

The theoretical contribution can therefore be summarized as a separation between three levels of strategy analysis: undecidability of strategy optimality is shown in the full unbounded model, exact strategy computation is recovered inside bounded finite approximations; and the distance between the approximated strategy and the continuous-time model is controlled through explicit error analysis. The following stage investigates whether this theoretical construction can be realized and verified through existing verification tools.

12.2.4 Verification and Strategy Synthesis at the Right Scale

After the theoretical analysis, we examined how far existing formal verification tools could support the analysis of the proposed rescue model. While verification concerns the analysis of a given model against specified properties or quantitative objectives, synthesis deals with the construction of a strategy or controller that realizes a desired objective. Our experiments with PRISM and MoDeST did not yield a complete synthesized rescue strategy for the full IMCC model. Instead, they provided restricted encodings, executable traces, and quantitative upper and lower bounds on the expected rescue outcome.

As IMCCs introduce a combination of semantic features that is not directly implemented in current tools, the verification task could not be approached as a direct translation. In PRISM, the rescue problem was approximated through a probabilistic game-oriented encoding in order to represent strategic choices and stochastic outcomes in a form that could be analyzed. In MoDeST, the modeling attempt followed the direction of stochastic timed systems, which are closer to the clock-governed aspects of the mission. Both directions captured meaningful fragments of the rescue scenario, but neither could preserve the simultaneous presence of explicit clocks with resets, urgent interaction, continuously time-dependent autonomous rates, stochastic interruption of planned actions, and evolving strategic choices.

A second limitation was the scale. Even where restricted encodings were possible, the number of states and timed configurations grew rapidly with the number of patients, robots, and possible rescue decisions. This made full exhaustive analysis unsuitable for dynamic rescue instances of realistic size. The difficulty was therefore not only the implementation of IMCCs full semantics, but also that the underlying planning problem remains computationally demanding even after approximation.

This stage therefore contributes a precise account of the current verification boundary. The model checking attempts demonstrate which parts of the model can be represented, which assumptions are necessary for analysis, and where semantic mismatch prevents faithful encoding. In particular, mapping the full model into available tool semantics requires approximations that either simplify the state structure, restrict the interaction pattern, or reduce the temporal behavior to a less expressive form.

They also clarify why existing verification tools cannot serve as the final planning mechanism for the full dynamic rescue problem, which motivates the final algorithmic part of the thesis. Since existing tools can provide formal analysis only for restricted encodings, and do not deliver complete strategy synthesis for the intended rescue model, practical rescue planning requires a separate adaptive optimization method capable of operating under the dynamic conditions represented by the model.

12.2.5 Strategic Planner in Dynamic Environments

The next step is the real world: physics and control for deploying robots in a disaster environment. With the genetic algorithm that we proposed, a foundation for automatic decision-making and planning in dynamic settings has been constructed. In this formulation, changing parameters drive strategy changes, so that the system adapts to save more lives as conditions evolve. The design goal is to behave adaptively under updates and to produce a good sequence fast, then keep improving as uncertainties arrive.

The rescue-planning task is then treated as an optimization problem in which the objective is to maximize the expected number of saved lives. A strategy is represented by a rescue sequence with its quality depending on travel time, rescue duration, the occurrence of stochastic death events, the current and predicted survival probabilities of the victims, and the time at which each rescue action is expected to occur. This formulation reflects the main insight of the thesis: the value of a rescue action is time-dependent, and therefore the value of a sequence must be evaluated through the survival dynamics induced by its execution.

To solve this problem at a practical scale, we developed a modified genetic algorithm tailored to the rescue setting. The algorithm searches over candidate rescue sequences, evaluates them using survival-based criteria, and updates the plan as the mission evolves. When patients die, when the active set changes, or when new information modifies the expected value of the remaining actions, the planner revises the current sequence instead of treating the initial solution as final. In this sense, the planner is an adaptive scheduling mechanism that repeatedly relates the current state of the mission to the best available continuation.

A central design feature of the proposed algorithm is the use of severity-based structuring. Patients are not only treated individually based on their survival probability, but also are grouped according to the urgency implied by their survival dynamics. In this thesis, severity is understood through the change of the survival curve over time, rather than as a static survival level alone. This makes the partitioning sensitive to how rapidly a patient's condition is deteriorating and allows the planner to focus computational effort on decisions whose timing is most critical. The severity-aware version is then compared against a plain non-partitioned version in order to evaluate whether this structuring improves planning performance under time-critical conditions.

The simulation results provide the practical validation layer of this thesis. They show how the generated strategies depend on survival functions, rescue times, robot speed, stochastic deaths, and the inclusion of computation time in the mission timeline. The comparison between the severity-based and plain approaches demonstrates the importance of organizing the search space in a way that reflects the temporal risk structure of the rescue problem. In a dynamic mission, a strategy is valuable not only because it has a high expected outcome in principle, but because it can be generated, revised, and executed before the opportunity it exploits disappears.

The contribution of this stage is therefore a practical near-optimal planning method that is consistent with the formal view developed earlier. The modified genetic algorithm neither replaces the theoretical results, nor claims unrestricted global optimality. Instead, it provides an implementable response to the limits identified by the theory and the verification experiments: when full exact synthesis is unavailable at realistic scale, adaptive heuristic planning offers a defensible way to produce rescue schedules under explicit time pressure.

12.2.6 Integrated Contribution

Stepping back, the through-line is consistent. The problem we addressed is critical and complex not because it is large, but because it is alive! Its parameters change,

its assumptions move, and its stakes are human. Current thesis, therefore, answers the strategic planning problem at several levels. At the modeling level, it provides a formal language for representing rescue missions with time-dependent stochastic risk, clocks, interaction, and concurrent agents. At the theoretical level, it clarifies where exact optimality is unattainable and where bounded approximations make strategy synthesis possible. At the implementation level, it shows why current model checking tools are insufficient for the full model while still useful for restricted fragments. Eventually, at the algorithmic level, it offers a severity-aware modified genetic algorithm that generates adaptive rescue sequences under realistic computational constraints.

This layered contribution is essential for the kind of problem studied here. In dynamic disaster response, a useful strategy cannot be separated from the model that defines its value, the theory that bounds its computability, the tools that reveal its implementation limits, or the algorithm that produces decisions within operational time. This dissertation brings these layers together and forms an end-to-end route from a disaster-response planning problem to a formal and implementable decision framework as a platform for future systems facing similarly dynamic and uncertain realities.

12.3 Limitations & Assumptions Revisited

We confronted a problem whose dimensionality does not simply grow; it shifts as the scene evolves. New variables appear, known ones drift, and different disciplines reveal different slices of truth. Mechanical engineering cares about what the robot can physically do with its sensors, actuators, and limited degrees of freedom in a fractured environment; electrical engineering sees sensing chains, latencies, and failure modes focusing on hardware reliability, power budgets, and communication fidelity. In another corner of this world, medicine sees triage doctrine and treatment timelines, while psychology tracks stress, attention, and trust.

What formal methods and computer science empower us with, is not a replacement for any of these perspectives. It offers a well-posed decision frame that makes the problem solvable at all by stating what can be proved, what can be computed in time, and where guarantees end. For instance, if an *optimal strategy* is even accessible for a given formulation, formal reasoning is the tool that tells us why and under which constraints.

None of that, however, is possible without abstraction: we must reduce the world until it fits into a model, without sanding off the very features that make disaster response different from ordinary routing or scheduling problems. Everything that

follows is stated in that spirit –transparent choices, their consequences, and why we judged the trade-offs acceptable for a first coherent step.

Modeling Choices that Made the Problem Tractable

Our modeling stance deliberately trades full physical fidelity for a decision frame: time-critical survival under changing access. To get there, we assume piece-wise Lipschitz continuity in survival curves, smooth within short intervals which excludes abrupt medical events such as sudden airway compromise or re-bleeds that produce true jumps. We compress each victim health state into a single scalar survival function, with heterogeneity (age, injury mix and etc) implicitly encoded in these individual curves. Space becomes a graph of locations and corridors with deterministic or interval travel times, and dynamics enter the state via time-decaying survival rates only.

While IMCCs model continuous time passage, their semantics is first defined in a Δ -granular discrete form. This discretization allows autonomous stochastic transitions with continuously varying rates to be expressed as probabilistic branching at discrete instants $t \in \Delta\mathbb{N}$. The continuous-time semantics is then recovered as the limit $\Delta \rightarrow 0$. In IMCC, we have two competing phenomena:

- *Interactive (forced) transitions* that must happen as soon as their guards are true
- *Autonomous (stochastic) transitions* that may happen at any time according to a continuously changing rate.

A purely continuous definition would lack a measurable basis for probability assignment and would break compositionality, as stochastic and urgent (forced) transitions interact at infinitesimal time scales. Hence, discrete Δ -semantics provides the mathematically sound foundation on which continuous behavior is reconstructed. Control acts on a fixed decision step Δ . Survival curves are continuous underneath but evaluated on this grid.

Algorithmic Scope and What It Implies

On the algorithmic side and practical regimes that matter to field rescue team – multiple patients, complex time decay, and real-time changes– the Genetic Algorithm emerged as the most viable option among the methods we studied. The choice is honest about its limits: GA is anytime and lacks worst-case optimality guarantees. Its performance depends on representation and operators and the search space

grows sharply with scene size. Even so, in realistic numerical experiments the GA consistently outperformed simpler heuristics, which validates it as the best compromise between solution quality and computational feasibility under real rescue constraints.

Survival curves are simplified into smooth continuous functions per victim. Field data are noisy and context-specific, and our calibration aims for reasonable rather than site-exact parameters. The interface to other disciplines is necessarily stylized: mechanical limits (payload, endurance), electronics (sensing pipelines), medical doctrine (treatment protocols), and behavioral factors (panic, refusal to evacuate) are referenced rather than fully modeled. The scheduler assumes those realities are reflected in its inputs namely, treatment times and hazard-specific survival decay.

Why These Boundaries Are Necessary at This step

We adopted only those assumptions without which, the problem would be too complex to even interrogate, let alone solve. The dimensionality of rescue under disaster conditions expands with the breadth of the incident itself. Thus, the first responsible step toward saving more lives is to reduce the complexity to a level at which the core question can be analyzed.

Accordingly, we chose a level of abstractions that preserve the soul of the problem—time-critical strategic autonomous rescue missions—while trimming layers that would freeze analysis without changing the central question. Therefore, these boundaries above are deliberate modeling and algorithmic design decisions. They mark where our guarantees apply and where they do not, and they point to the work that remains: richer hazard composition, multi-objective goals that include equity and responder safety, tighter links to perception and motion, and deployment in mixed-initiative settings. The thesis opens those directions while staying honest about the ground it has not yet covered.

12.4 Future Research Avenues

The goal of going forward is not only to widen the scope and bring other possibilities into consideration, but also is to sharpen and deepen the current approach's capabilities. The following outlines concrete steps that keep the thesis's center of gravity while making the system more capable, explainable, and transferable.

Multi-Agent Scheduling Algorithms. The next step is to lift the scheduling engine from one robot to coordinated teams with communication limits, contention for corridors/tools, and collision-avoidance models. The emphasis is on algorithms that remain anytime by producing a usable joint plan quickly and on light guarantees (e.g., bounded suboptimality under bounded communication delay) so behavior stays tractable as the number of agents grows.

Mathematical Cases of the Model Analysis. To complement general heuristics, we will carve out special IMCC cases where clean theorems are possible and aim to prove structural properties (decidability, approximability and exact computations). These results become both sanity checks for the code and fast certifiers we can invoke on solvable subproblems inside larger scenes.

Multi-Objective Optimization. Expected survivors remains the anchor, but real deployments trade it against responder risk and allocation equity. We will further include and implement different simultaneous objectives into current algorithm and add a policy selector that lets incident command fix a defensible operating point. On the theory side, we target approximation/robustness bounds for multi-objective slices (e.g., survivors vs. risk) and study how the frontier moves under updates, so the system can recommend not just a sequence, but a transparent trade-off.

Human-in-the-Loop. The planner should further be able to collaborate, by employing a mixed-initiative layer where commanders can lock constraints, reserve teams or tools, and pose counterfactuals (“why not B first?”) with immediate re-optimization. The objective is a loop that remains fast and auditable under pressure, and whose outputs map cleanly to USAR/medical workflow.

Generalization to CPS Domains. The formal core of our IMCC model extends naturally to other cyber-physical settings where minutes determine outcomes. Further case studies in traffic incident management, industrial safety, and security response, reusing the same IMCC backbone and scheduler, can be investigated with possible yet necessary domain-specific refinements.

12.5 Destination Met, Clock Still Ticking

This dissertation began with the question we ask ourselves when we lose someone: *If we had chosen differently, if we had acted earlier in the course of rescuing them, would their chance of survival have been higher?* In disasters, someone’s life, or many lives, can hinge on our decisions, our tempo, and our planning at that very moment. Whatever strategy we adopt should carry us to a point where, regardless of the

outcome, we can say with intellectual honesty: *given the circumstances and what was knowable, this was the best we could do.*

Although the technical focus here is autonomous rescue in disaster scenes, the motivation is broader: mishandling a disaster is itself a second disaster, with the loss of victims at its peak. The path developed in this work lays a foundation for acting better, with less regret, where uncertainty rules. By mapping hazard into time-dependent survival, making time and budget constraints explicit, and by treating sequencing as a first-class decision, the work turns an overwhelming scene into a structure that can be analyzed, followed by further stress-tests, and explanations.

And yet, the clock is still running. The scope of this problem is as wide as the disasters that trigger it, and closing this document opens new vantage points: richer multi-hazard interactions, mixed-initiative planning with humans in the loop, and shared benchmarks that let others challenge and extend these ideas. For me, as long as scenes like these exist, I intend to work where minutes become decisions; and decisions, when made well, become survivals.

References

- [ABK22] Shaull Almagor, Udi Boker, and Orna Kupferman. „What’s decidable about weighted automata?“ In: *Information and Computation* 282 (2022), p. 104651 (cit. on p. 71).
- [AD94] Rajeev Alur and David L Dill. „A Theory of Timed Automata“. In: *Theoretical Computer Science* 126.2 (1994), pp. 183–235 (cit. on pp. 25, 27, 28, 41, 43, 49, 52, 69, 71).
- [AH97] Sudhir Anand and Kara Hanson. „Disability-adjusted life years: a critical review“. In: *Journal of Health Economics* 16.6 (Dec. 1997), pp. 685–702 (cit. on p. 6).
- [Ahm+20] M. Ahmadi, N. Jansen, B. Wu, and A. Abate. „Control theory meets POMDPs: A hybrid systems approach“. In: *IEEE Transactions on Automatic Control* 65.12 (2020), pp. 5077–5090 (cit. on p. 139).
- [Ait15] A. Aitsi-Selmi. „The Sendai framework for disaster risk reduction: Renewing the global commitment to people’s resilience, health, and well-being“. In: *Springer-Plus* 4.1 (2015), pp. 1–3 (cit. on p. 6).
- [Ala+16] A.Q. Alarhayem, J.G. Myers, D. Dent, et al. „Time is the enemy: Mortality in trauma patients with hemorrhage from torso injury occurs long before the “golden hour”“. In: *The American Journal of Surgery* 212.6 (2016). Southwestern Surgical Congress, pp. 1101–1105 (cit. on p. 12).
- [ALR22] Étienne André, Didier Lime, and Olivier H Roux. „Reachability and liveness in parametric timed automata“. In: *Logical Methods in Computer Science* 18 (2022) (cit. on p. 71).
- [Alu+92] Rajeev Alur, Costas Courcoubetis, Thomas A. Henzinger, and Pei-Hsin Ho. „Hybrid Automata: An Algorithmic Approach to the Specification and Verification of Hybrid Systems“. In: *Hybrid Systems*. Ed. by Robert L. Grossman, Anil Nerode, Anders P. Ravn, and Hans Rischel. Vol. 736. Lecture Notes in Computer Science. Springer, 1992, pp. 209–229 (cit. on p. 44).
- [AM25] Shahad Alqefari and Mohamed El Bachir Menai. „Multi-UAV task assignment in dynamic environments: Current trends and future directions“. In: *Drones* 9.1 (2025), p. 75 (cit. on p. 138).
- [Ama+16] Christopher Amato, George Konidaris, Ariel Anders, et al. „Policy search for multi-robot coordination under uncertainty“. In: *The International Journal of Robotics Research* 35.14 (Dec. 2016), pp. 1760–1778 (cit. on p. 139).

- [Ani85] David J Anick. „Diophantine equations, Hilbert series, and undecidable spaces“. In: *Annals of Mathematics* 122.1 (1985), pp. 87–112 (cit. on p. 71).
- [App+11] David L. Applegate, Robert E. Bixby, Vašek Chvátal, and William J. Cook. „The traveling salesman problem: a computational study“. In: *The Traveling Salesman Problem*. Princeton university press, 2011 (cit. on p. 150).
- [AS16] Walker S. Ashley and Stephen M. Strader. „Recipe for Disaster: How the Dynamic Ingredients of Risk and Exposure Are Changing the Tornado Disaster Landscape“. In: *Bulletin of the American Meteorological Society* 97 (5 2016), pp. 767–786 (cit. on p. 4).
- [ASV22] Shakhnaz Akhmedova, Vladimir Stanovov, and Aleksei Vakhnin. „A hybrid self-adapting multi-swarm algorithm based on PSO and CMA-ES for continuous dynamic optimization“. In: *Proceedings of the genetic and evolutionary computation conference companion*. 2022, pp. 387–390 (cit. on p. 148).
- [Bai+03] Christel Baier, Boudewijn Haverkort, Holger Hermanns, and Joost-Pieter Katoen. „Model-checking algorithms for continuous-time Markov chains“. In: *IEEE Transactions on Software Engineering* 29.6 (June 2003), pp. 524–541 (cit. on p. 98).
- [Bai+07] Christel Baier, Nathalie Bertrand, Patricia Bouyer, Thomas Brihaye, and Marcus Größer. „Probabilistic and Topological Semantics for Timed Automata“. In: *International Conference on Foundations of Software Technology and Theoretical Computer Science*. Springer. 2007, pp. 179–191 (cit. on p. 43).
- [Bay+15] Jamil D. Bayram, Shawki Zuabi, Caitlin M. McCord, et al. „Disaster metrics: evaluation of de Boer’s Disaster Severity Scale (DSS) applied to earthquakes“. In: *Prehospital and Disaster Medicine* 30 (1 2015), pp. 22–27 (cit. on p. 8).
- [BB14] Rajendra Kumar Bhandari and Rajendra Kumar Bhandari. „Disaster Terminology“. In: *Disaster Education and Management: A Joyride for Students, Teachers and Disaster Managers* (2014), pp. 29–53 (cit. on p. 4).
- [BB21] Martin Braquet and Efstathios Bakolas. „Greedy decentralized auction-based task allocation for multi-agent systems“. In: *IFAC-PapersOnLine* 54.20 (2021), pp. 675–680 (cit. on p. 140).
- [BC96] Alexandre Boudet and Hubert Comon. „Diophantine equations, Presburger arithmetic and finite automata“. In: *Trees in Algebra and Programming—CAAP’96: 21st International Colloquium Linköping, Sweden, April 22–24, 1996 Proceedings* 21. Springer. 1996, pp. 30–43 (cit. on p. 71).
- [BCH23] Sander Beckers, Hana Chockler, and Joseph Y. Halpern. „Quantifying Harm“. In: (Aug. 2023). Ed. by Edith Elkind. Main Track, pp. 363–371 (cit. on p. 8).
- [BD16] Sunith Bandaru and Kalyanmoy Deb. „Metaheuristic techniques“. In: *Decision sciences*. CRC Press, 2016, pp. 709–766 (cit. on p. 144).
- [Bea03] Daniele Beauquier. „On Probabilistic Timed Automata“. In: *Theoretical Computer Science* 292.1 (2003), pp. 65–84 (cit. on p. 43).
- [Beh+01] Gerd Behrmann, Ansgar Fehnker, Thomas Hune, et al. „Minimum-cost reachability for priced time automata“. In: *Hybrid Systems: Computation and Control: 4th International Workshop, HSCC 2001 Rome, Italy, March 28–30, 2001 Proceedings* 4. Springer. 2001, pp. 147–161 (cit. on p. 71).

- [Bel57a] Richard Bellman. „A Markovian Decision Process“. In: *Journal of Mathematics and Mechanics* 6.5 (1957), pp. 679–684 (cit. on pp. 42, 72).
- [Bel57b] Richard Bellman. *Dynamic Programming*. Princeton University Press, 1957 (cit. on pp. 42, 72, 88).
- [BG02a] Mario Bravetti and Roberto Gorrieri. „The theory of interactive generalized semi-Markov processes“. In: *Theoretical Computer Science* 282.1 (2002), pp. 5–32 (cit. on p. 43).
- [BG02b] Olli Bräysy and Michel Gendreau. „Tabu search heuristics for the vehicle routing problem with time windows“. In: *Top* 10.2 (2002), pp. 211–237 (cit. on p. 146).
- [BG04] Achim Blumensath and Erich Grädel. „Finite presentations of infinite structures: Automata and interpretations“. In: *Theory of Computing Systems* 37.6 (Dec. 2004), pp. 641–674 (cit. on p. 24).
- [Bha+23] Sushmita Bhattacharya, Siva Kailas, Sahil Badyal, Stephanie Gil, and Dimitri Bertsekas. „Multiagent reinforcement learning: Rollout and policy iteration for pomdp with application to multirobot problems“. In: *IEEE Transactions on Robotics* 40 (2023), pp. 2003–2023 (cit. on p. 139).
- [BHH21] Yuliya Butkova, Arnd Hartmanns, and Holger Hermanns. „A modest approach to markov automata“. In: *ACM Transactions on Modeling and Computer Simulation (TOMACS)* 31.3 (2021), pp. 1–34 (cit. on p. 124).
- [BJM13] Ali Bozorgi-Amiri, Mohamad Saeed Jabalameli, and S. M. J. Mirzapour Al-e-Hashem. „A multi-objective robust stochastic programming model for disaster relief logistics under uncertainty“. In: *OR spectrum* 35.4 (2013), pp. 905–933 (cit. on p. 140).
- [BK08] Christel Baier and Joost-Pieter Katoen. *Principles of model checking*. MIT press, 2008 (cit. on pp. 88, 97).
- [BKW18] Nicolas Basset, Marta Kwiatkowska, and Clemens Wiltsche. „Compositional Strategy Synthesis for Stochastic Games with Multiple Objectives“. In: *Information and Computation* 261 (2018), pp. 536–587 (cit. on p. 42).
- [Bla07] Tim Blackwell. „Particle swarm optimization in dynamic environments“. In: *Evolutionary computation in dynamic and uncertain environments*. 2007, pp. 29–49 (cit. on p. 140).
- [Blo+99] Osbert Blow, Lissa Magliore, Jeffrey A. Claridge, Kathy Butler, and Jeffrey S. Young. „The golden hour and the silver day: detection and correction of occult hypoperfusion within 24 hours improves outcome from major trauma.“ In: *The Journal of trauma* 47 5 (1999), pp. 964–9 (cit. on p. 12).
- [BLR04] Gerd Behrmann, Kim G Larsen, and Jacob I Rasmussen. „Priced timed automata: Algorithms and applications“. In: *International symposium on formal methods for components and objects*. Springer. 2004, pp. 162–182 (cit. on p. 71).
- [BLR11] Santiago R. Balseiro, Irene Loiseau, and Juan Ramonet. „An ant colony algorithm hybridized with insertion heuristics for the time dependent vehicle routing problem with time windows“. In: *Computers & Operations Research* 38.6 (2011), pp. 954–966 (cit. on p. 147).

- [BMR17] Damien Busatto-Gaston, Benjamin Monmege, and Pierre-Alain Reynier. „Optimal reachability in divergent weighted timed games“. In: *International Conference on Foundations of Software Science and Computation Structures*. Springer. 2017, pp. 162–178 (cit. on p. 72).
- [BMR21] Damien Busatto-Gaston, Benjamin Monmege, and Pierre-Alain Reynier. „Optimal controller synthesis for timed systems“. In: *CoRR* abs/2104.12577 (2021). arXiv: 2104.12577 (cit. on p. 72).
- [Boh+06] Henrik Bohnenkamp, Pedro R D’Argenio, Holger Hermanns, and J-P Katoen. „MoDeST: A compositional modeling formalism for hard and softly timed systems“. In: *IEEE Transactions on Software Engineering* 32.10 (2006), pp. 812–830 (cit. on p. 124).
- [Brá+12] Tomás Brázdil, Holger Hermanns, Jan Krcál, Jan Kreťinsky, and Vojtech Reháť. „Verification of open interactive Markov chains“. In: *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2012)*. Schloss Dagstuhl–Leibniz-Zentrum für Informatik. 2012, pp. 474–485 (cit. on p. 72).
- [BS23] Paul C. Bell and Pavel Semukhin. „Decision Questions for Probabilistic Automata on Small Alphabets“. In: *Logical Methods in Computer Science* 19 (4 Dec. 2023) (cit. on p. 72).
- [BTT18] Peter Baumgartner, Sylvie Thiébaux, and Felipe W. Trevizan. „Heuristic Search Planning With Multi-Objective Probabilistic LTL Constraints“. In: *KR*. 2018, pp. 415–424 (cit. on p. 99).
- [Bud+20] Carlos E. Budde, Pedro R. D’Argenio, Arnd Hartmanns, and Sean Sedwards. „An efficient statistical model checker for nondeterminism and rare events“. In: *International journal on software tools for technology transfer* 22.6 (Dec. 2020), pp. 759–780 (cit. on pp. 100, 124).
- [Bud+24] Carlos E. Budde, Pedro R. D’Argenio, Juan A. Fraire, Arnd Hartmanns, and Zhen Zhang. „Modest Models and Tools for Real Stochastic Timed Systems“. In: *Principles of Verification: Cycling the Probabilistic Landscape: Essays Dedicated to Joost-Pieter Katoen on the Occasion of His 60th Birthday, Part II*. Ed. by Joost-Pieter Katoen. Cham: Springer Nature Switzerland, 2024, pp. 115–142 (cit. on p. 124).
- [Bul+12] Peter Bulychyev, Alexandre David, Kim Gulstrand Larsen, et al. „UPPAAL-SMC: Statistical model checking for priced timed automata“. In: *arXiv preprint arXiv:1207.1272* (2012) (cit. on p. 99).
- [Cap21] Ellen Cappel. „Data-driven multi-robot planning for online user-directed coordination“. PhD thesis. Carnegie Mellon University, 2021 (cit. on p. 148).
- [Car+12] Giulio Caravagna, Alex Graudenzi, Marco Antoniotti, Giancarlo Mauri, and Alberto d’Onofrio. „Effects of delayed immune-response in tumor immune-system interplay“. In: *arXiv preprint arXiv:1208.3855* (2012) (cit. on p. 73).
- [Cas+08] Alberto Casagrande, Pietro Corvaja, Carla Piazza, and Bud Mishra. „Decidable Compositions of O-Minimal Automata“. In: *Automated Technology for Verification and Analysis*. Ed. by Sungdeok (Steve) Cha, Jin-Young Choi, Moonzoo Kim, Insup Lee, and Mahesh Viswanathan. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 274–288 (cit. on p. 71).

- [CE81] Edmund M. Clarke and E. Allen Emerson. „Design and Synthesis of Synchronization Skeletons Using Branching-Time Temporal Logic“. In: *Logic of Programs, Workshop*. Berlin, Heidelberg: Springer-Verlag, 1981, pp. 52–71 (cit. on p. 97).
- [CFH16] M. Chen, E. Frazzoli, and D. Hsu. „POMDP-lite for robust robot planning under uncertainty“. In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. 2016, pp. 1000–1007 (cit. on p. 139).
- [CG04] Frank Ciesinski and Marcus Größer. „On probabilistic computation tree logic“. In: *Validation of stochastic systems: a guide to current research*. 1. Berlin, Heidelberg: Springer Berlin Heidelberg, July 2004, pp. 162–170 (cit. on p. 98).
- [CG87] Edmund M. Clarke and Orna Grumberg. „Avoiding the state explosion problem in temporal logic model checking“. In: *Proceedings of the sixth annual ACM Symposium on Principles of distributed computing*. 1987, pp. 294–303 (cit. on p. 97).
- [Che+13a] Taolue Chen, Vojtěch Forejt, Marta Kwiatkowska, David Parker, and Aistis Simaitis. „PRISM-games: A model checker for stochastic multi-player games“. In: *Tools and Algorithms for the Construction and Analysis of Systems: 19th International Conference, TACAS 2013*. Springer. 2013, pp. 185–191 (cit. on p. 106).
- [Che+13b] Taolue Chen, Vojtěch Forejt, Marta Kwiatkowska, Aistis Simaitis, and Clemens Wiltsche. „On Stochastic Games with Multiple Objectives“. In: *Mathematical Foundations of Computer Science, MFCS 2013*. Springer. 2013, pp. 266–277 (cit. on p. 42).
- [CL89] J. Carroll and D. Long. *Theory of finite automata: with an introduction to formal languages*. philarchive.org, 1989 (cit. on pp. 24, 25).
- [CLP20] Lorenzo Clemente, Sławomir Lasota, and Radosław Piórkowski. „Determinisability of one-clock timed automata“. In: *arXiv preprint arXiv:2007.09340* (2020) (cit. on p. 34).
- [CLV07] Carlos A. Coello Coello, Gary B. Lamont, and David A. Van Veldhuizen. *Evolutionary algorithms for solving multi-objective problems*. Boston, MA: Springer US, 2007 (cit. on p. 146).
- [CR06] A. E. Carter and C. T. Ragsdale. „A new approach to solving the multiple traveling salesperson problem using genetic algorithms“. In: *European Journal of Operational Research* 175.1 (2006), pp. 246–257 (cit. on p. 150).
- [CRJ24] Vladimir M. Cvetković, Renate Renner, and Vladimir Jakovljević. „Industrial disasters and hazards: From causes to consequences—A holistic approach to resilience“. In: *International Journal of Disaster Risk Management* 6.2 (2024), pp. 149–168 (cit. on p. 5).
- [CT00] Costas Courcoubetis and Stavros Tripakis. „Probabilistic Model Checking: Formalisms and Algorithms for Discrete and Real-Time Systems“. In: *Verification of Digital and Hybrid Systems*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 183–219 (cit. on p. 98).
- [CW24] H Jithamala Caldera and SC Wirasinghe. „Evolution of natural disaster terminologies, with a case study of the covid-19 pandemic“. In: *Scientific Reports* 14.1 (2024), p. 14616 (cit. on p. 5).

- [CZ15] Xiang Chu and QiuYan Zhong. „Post-earthquake allocation approach of medical rescue teams“. In: *Natural hazards* 79.3 (2015), pp. 1809–1824 (cit. on p. 13).
- [DAr+01] Pedro R D’Argenio, Holger Hermanns, Joost-Pieter Katoen, and Ric Klaren. „Modest—a modelling and description language for stochastic timed systems“. In: *Joint International Workshop von Process Algebra and Probabilistic Methods, Performance Modeling and Verification*. Springer Berlin Heidelberg. 2001, pp. 87–104 (cit. on p. 124).
- [Dav+15] Alexandre David, Kim G. Larsen, Axel Legay, Marius Mikucionis, and Danny Bøgsted Poulsen. „Uppaal SMC tutorial“. In: *Int. J. Softw. Tools Technol. Transf.* 17.4 (2015), pp. 397–415 (cit. on p. 43).
- [Dav91] Yuval Davidor. *Genetic Algorithms and Robotics: A heuristic strategy for optimization*. Vol. 1. World Scientific Publishing Company, 1991 (cit. on p. 145).
- [DD20] Yaser Damchi and Mohammad Dolatabadi. „Hybrid VNS–LP algorithm for online optimal coordination of directional overcurrent relays“. In: *IET Generation, Transmission Distribution* 14.23 (2020), pp. 5447–5455 (cit. on p. 146).
- [Deb15] Kalyanmoy Deb. „Multi-objective evolutionary algorithms“. In: *Springer handbook of computational intelligence*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 995–1015 (cit. on p. 146).
- [Deh+17] Christian Dehnert, Sebastian Junges, Joost-Pieter Katoen, and Matthias Volk. „A storm is coming: A modern probabilistic model checker“. In: *International Conference on Computer Aided Verification*. Cham: Springer International Publishing, 2017, pp. 592–600 (cit. on p. 99).
- [Des+15] Sachin Desale, Akhtar Rasool, Sushil Andhale, and Priti Rane. „Heuristic and meta-heuristic algorithms and their relevance to the real world: a survey“. In: *Int. J. Comput. Eng. Res. Trends* 351.5 (2015), pp. 2349–7084 (cit. on p. 144).
- [DGT12] Gaifang Dong, William W. Guo, and Kevin Tickle. „Solving the traveling salesman problem using cooperative genetic ant systems“. In: *Expert systems with applications* 39.5 (2012), pp. 5006–5011 (cit. on p. 150).
- [DKB97] Pedro R D’Argenio, Joost P Katoen, and Hendrik Brinksma. „A stochastic automata model and its algebraic approach“. In: *5th International Workshop on Process Algebra and Performance Modeling, PAPM 1997*. Centre for Telematics and Information Technology (CTIT). 1997, pp. 1–16 (cit. on pp. 43, 52).
- [Duc+14] František Duchoň, Andrej Babinec, Martin Kajan, et al. „Path planning with modified a star algorithm for a mobile robot“. In: *Procedia engineering* 96 (2014), pp. 59–69 (cit. on p. 149).
- [EHZ10] Christian Eisentraut, Holger Hermanns, and Lijun Zhang. „On Probabilistic Automata in Continuous Time“. In: *2010 25th Annual IEEE Symposium on Logic in Computer Science*. IEEE. 2010, pp. 342–351 (cit. on pp. 43, 72).
- [ENT11] Murugappan Elango, Subramanian Nachiappan, and Manoj Kumar Tiwari. „Balancing task allocation in multi-robot systems using K-means clustering and auction based mechanisms“. In: *Expert Systems with Applications* 38.6 (2011), pp. 6486–6491 (cit. on p. 147).

- [Fed19] Federal Emergency Management Agency. *National Response Framework*. Tech. rep. Washington, DC: U.S. Department of Homeland Security, 2019 (cit. on p. 8).
- [Fed25] Federal Emergency Management Agency (FEMA). *FEMA Hazards Portal*. <https://hazards.fema.gov/>. Accessed: 26 Sep 2025. 2025 (cit. on pp. 7, 8).
- [FH03] Norman Ferrier and C. Emdad Haque. „Hazards risk assessment methodology for emergency managers: A standardized framework for application“. In: *Natural hazards* 28.2 (Mar. 2003), pp. 271–290 (cit. on p. 5).
- [Fis+18] Jaime F. Fisac, Anayo K. Akametalu, Melanie N. Zeilinger, et al. „A general safety framework for learning-based control in uncertain robotic systems“. In: *IEEE Transactions on Automatic Control* 64.7 (2018), pp. 2737–2752 (cit. on p. 141).
- [FKK24] Martin Fränzle, Rabeaeh Kiaghadi, and Paul Kröger. „Rescuing Catastrophe Victims by Interactive Markov Chains with Clocks“. In: *Principles of Verification: Cycling the Probabilistic Landscape: Essays Dedicated to Joost-Pieter Katoen on the Occasion of His 60th Birthday, Part III*. Springer, 2024, pp. 374–395 (cit. on pp. 69, 73, 82).
- [Foe+21] Daniel Foead, Alifio Ghifari, Marchel Budi Kusuma, Novita Hanafiah, and Eric Gunawan. „A systematic literature review of A* pathfinding“. In: *Procedia Computer Science* 179 (2021), pp. 507–514 (cit. on p. 149).
- [Frä+11] Martin Fränzle, Ernst Moritz Hahn, Holger Hermanns, Nicolás Wolovick, and Li-jun Zhang. „Measurability and safety verification for stochastic hybrid systems“. In: *Proc. of the 14th Int. Conf on Hybrid Systems: Computation and Control*. ACM, 2011, pp. 43–52 (cit. on p. 44).
- [Frä+22] Martin Fränzle, Mahsa Shirmohammadi, Mani Swaminathan, and James Worrell. „Costs and rewards in priced timed automata“. In: *Inf. Comput.* 282 (2022), p. 104656 (cit. on p. 71).
- [GA13] Thomas Glade and David E. Alexander. „Classification of Natural Disasters“. In: *Encyclopedia of Natural Hazards*. Ed. by Peter T. Bobrowsky. Dordrecht: Springer Netherlands, 2013, pp. 78–82 (cit. on p. 5).
- [Gau+19] Tobias Gauss, François-Xavier Ageron, Marie-Laure Devaud, et al. „Association of Prehospital Time to In-Hospital Trauma Mortality in a Physician-Staffed Emergency Medicine System“. In: *JAMA Surgery* 154.12 (2019), p. 1117 (cit. on p. 13).
- [GD05] Piotr J. Gmytrasiewicz and Prashant Doshi. „A framework for sequential planning in multi-agent settings“. In: *Journal of Artificial Intelligence Research* 24 (2005), pp. 49–79 (cit. on p. 139).
- [GL23] Mitsuo Gen and Lin Lin. „Genetic algorithms and their applications“. In: *Springer handbook of engineering statistics*. London: Springer London, 2023, pp. 635–674 (cit. on p. 145).
- [Gmi+21] Maha Gmira, Michel Gendreau, Andrea Lodi, and Jean-Yves Potvin. „Tabu search for the time-dependent vehicle routing problem with time windows on a road network“. In: *European Journal of Operational Research* 288.1 (2021), pp. 129–140 (cit. on p. 146).

- [GO24] Quentin Guilmant and Joël Ouaknine. „Inapproximability in Weighted Timed Games“. In: *35th International Conference on Concurrency Theory (CONCUR 2024)*. Ed. by Rupak Majumdar and Alexandra Silva. Vol. 311. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024, 27:1–27:15 (cit. on p. 72).
- [Goy10] Sanchit Goyal. „A survey on travelling salesman problem“. In: *Midwest instruction and computing symposium*. 2010, pp. 1–9 (cit. on p. 150).
- [GPL20] C. R. Garrett, C. Paxton, and T. Lozano-Perez. „Online replanning in belief space for partially observable task and motion problems“. In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. 2020, pp. 1000–1007 (cit. on p. 139).
- [GSB25] Manya Goel, Juhi Singh, and Amit Kumar Bairwa. „Performance Evaluation of Heuristic and Meta-Heuristic Algorithms on Large-Scale Travelling Salesman Problem Instances“. In: *IEEE Access* (2025) (cit. on p. 150).
- [Guc+12] Dennis Guck, Tingting Han, Joost-Pieter Katoen, and Martin R. Neuhäuser. „Quantitative timed analysis of interactive markov chains“. In: *Proceedings of the 4th International Conference on NASA Formal Methods*. NFM’12. Norfolk, VA: Springer-Verlag, 2012, pp. 8–23 (cit. on pp. 34, 72).
- [Guc+14] Dennis Guck, Hassan Hatefi, Holger Hermanns, Joost-Pieter Katoen, and Mark Timmer. „Analysis of timed and long-run objectives for Markov automata“. In: *Logical Methods in Computer Science* 10 (2014) (cit. on p. 72).
- [Guo+22] Zhen Guo, Zelin Wan, Qisheng Zhang, et al. „A survey on uncertainty reasoning and quantification for decision making: Belief theory meets deep learning“. In: *arXiv preprint arXiv:2206.05675* (2022) (cit. on p. 141).
- [Hah+13] Ernst Moritz Hahn, Arnd Hartmanns, Holger Hermanns, and Joost-Pieter Katoen. „A compositional modelling and analysis framework for stochastic hybrid systems“. In: *Formal Methods in System Design* 43.2 (2013), pp. 191–232 (cit. on p. 44).
- [Har12] Arnd Hartmanns. „Modest-a unified language for quantitative models“. In: *Proceedings of the 2012 Forum on Specification and Design Languages (FDL)*. IEEE. 2012, pp. 1–8 (cit. on p. 124).
- [Har15] Arnd Hartmanns. „On the analysis of stochastic timed systems“. PhD thesis. Faculty of Natural Science and Technology I of Saarland University, 2015 (cit. on pp. 44, 48, 124).
- [Har22] Arnd Hartmanns. „An overview of Modest models and tools for real stochastic timed systems“. In: *arXiv preprint arXiv:2203.09881* (2022) (cit. on p. 124).
- [HD23] James R. Heselden and Gautham P. Das. „Heuristics and rescheduling in prioritised multi-robot path planning: a literature review“. In: *Machines* 11.11 (2023), p. 1033 (cit. on p. 140).
- [He+21] Meiling He, Zhixiu Wei, Xiaohui Wu, and Yongtao Peng. „An adaptive variable neighborhood search ant colony algorithm for vehicle routing problem with soft time windows“. In: *Ieee Access* 9 (2021), pp. 21258–21266 (cit. on p. 147).

- [Her02] Holger Hermanns. *Interactive Markov Chains: The Quest for Quantified Quality*. Vol. 2428. Lecture Notes in Computer Science. Springer, 2002 (cit. on pp. 27, 34, 42, 52, 69).
- [Hes04] Joao P Hespanha. „Stochastic hybrid systems: Application to communication networks“. In: *International Workshop on Hybrid Systems: Computation and Control*. Springer. 2004, pp. 387–401 (cit. on pp. 43, 73).
- [Hes06] Joao P Hespanha. „Modelling and analysis of stochastic hybrid systems“. In: *IEEE Proc. -Control Theory and Applications* 153.5 (2006), pp. 520–535 (cit. on p. 25).
- [HH15a] Arnd Hartmanns and Holger Hermanns. „In the Quantitative Automata Zoo“. In: *Science of Computer Programming* 112 (2015), pp. 3–23 (cit. on pp. 42, 44, 49, 124).
- [HH15b] Hassan Hatefi and Holger Hermanns. „Improving time bounded reachability computations in interactive Markov chains“. In: *Science of Computer Programming* 112 (2015), pp. 58–74 (cit. on p. 72).
- [HH19] Arnd Hartmanns and Holger Hermanns. „A modest Markov automata tutorial“. In: *Reasoning Web. Explainable Artificial Intelligence: 15th International Summer School 2019, Bolzano, Italy, September 20–24, 2019, Tutorial Lectures*. Springer, 2019, pp. 250–276 (cit. on p. 124).
- [HHH14] E Moritz Hahn, Arnd Hartmanns, and Holger Hermanns. „Reachability and reward checking for stochastic timed automata“. In: *EASST electronic communications* 70 (2014) (cit. on pp. 44, 52).
- [Hil00] David Hilbert. „Mathematische probleme“. In: *Nachrichten von der Koniglichen Gesellschaft der Wissenschaften zu Gottingen* (1900) (cit. on p. 75).
- [HK09] Holger Hermanns and Joost-Pieter Katoen. „The How and Why of Interactive Markov Chains“. In: *International Symposium on Formal Methods for Components and Objects*. Springer. 2009, pp. 311–337 (cit. on pp. 34, 42, 69).
- [HKK13] Holger Hermanns, Jan Krčál, and Jan Křetínský. „Compositional verification and optimization of interactive markov chains“. In: *International Conference on Concurrency Theory*. Springer. 2013, pp. 364–379 (cit. on p. 72).
- [HKM08] Tingting Han, Joost-Pieter Katoen, and Alexandru Mereacre. „Compositional Modeling and Minimization of Time-inhomogeneous Markov Chains“. In: *Hybrid Systems: Computation and Control, HSCC 2008*. Springer. 2008, pp. 244–258 (cit. on p. 43).
- [HLS00] Jianghai Hu, John Lygeros, and Shankar Sastry. „Towards a theory of stochastic hybrid systems“. In: *International Workshop on Hybrid Systems: Computation and Control*. Springer. 2000, pp. 160–173 (cit. on pp. 43, 73).
- [HMU01] John E. Hopcroft, Rajeev Motwani, and Jeffrey D. Ullman. „Introduction to automata theory, languages, and computation“. In: *Acm Sigact News* 32.1 (2001), pp. 60–65 (cit. on p. 24).
- [HN16] Reza Hassanzadeh and Zorica Nedovic-Budic. „Where to go first: prioritization of damaged areas for allocation of Urban Search and Rescue (USAR) operations (PI-USAR model)“. In: *Geomatics, Natural Hazards and Risk* 7.4 (2016), pp. 1337–1366 (cit. on p. 157).

- [HS87] Peter J Haas and Gerald S Shedler. „Regenerative generalized semi-Markov processes“. In: *Stochastic Models* 3.3 (1987), pp. 409–438 (cit. on p. 43).
- [HU69] John E. Hopcroft and Jeffrey D. Ullman. *Formal languages and their relation to automata*. Addison-Wesley Longman Publishing Co., Inc., 1969 (cit. on p. 24).
- [HYT21] Wen-Jing Hong, Peng Yang, and Ke Tang. „Evolutionary computation for large-scale multi-objective optimization: A decade of progresses“. In: *International Journal of Automation and Computing* 18.2 (2021), pp. 155–169 (cit. on p. 145).
- [Jha10] Madan Kumar Jha. „Natural and Anthropogenic Disasters: An Overview“. In: *Natural and Anthropogenic Disasters: Vulnerability, Preparedness and Mitigation*. Ed. by Madan Kumar Jha. Dordrecht: Springer Netherlands, 2010, pp. 1–16 (cit. on p. 5).
- [JKR17] Steven James, George Konidaris, and Benjamin Rosman. „An analysis of monte carlo tree search“. In: 2017 (cit. on p. 148).
- [JM18] Yaping Jing and Andrew S. Miner. „Action and state based computation tree measurement language and algorithms“. In: *International Conference on Quantitative Evaluation of Systems*. Cham: Springer International Publishing, 2018, pp. 190–206 (cit. on p. 99).
- [Joh25] Richard Johnson. *Search Algorithms and Systems: Definitive Reference for Developers and Engineers*. HiTeX Press, 2025 (cit. on p. 146).
- [Jua+23] A. A. Juan, P. Keenan, R. Martí, S. McGarraghy, and J. Panadero. „A review of the role of heuristics in stochastic optimisation: From metaheuristics to learnheuristics“. In: *Annals of Operations Research* 323.1 (2023), pp. 1–38 (cit. on p. 141).
- [KA11] András Király and János Abonyi. „Optimization of Multiple Traveling Salesmen Problem by a Novel Representation Based Genetic Algorithm“. In: *Intelligent Computational Optimization in Engineering: Techniques and Applications*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 241–269 (cit. on p. 150).
- [Kam+24] Afshin Kamyabniya, Antoine Sauré, F. Sibel Salman, Noureddine Bénichou, and Jonathan Patrick. „Optimization models for disaster response operations: a literature review“. In: *Or Spectrum* 46.3 (2024), pp. 737–783 (cit. on p. 140).
- [KBM12] T. Kawano, F. Bouteau, and S. Mancuso. „Finding and defining the natural automata acting in living plants: toward the synthetic biology for robotics and informatics in vivo“. In: *Communicative Integrative Biology* 5.5 (2012), pp. 409–412 (cit. on p. 25).
- [KHH18] M. Kaviyani-Charati, F. Heidarzadeh Souraki, and M. Hajiaghahi-Keshteli. „A robust optimization methodology for multi-objective location-transportation problem in disaster response phase under uncertainty“. In: *International Journal of Engineering* 31.11 (Nov. 2018), pp. 1953–1961 (cit. on p. 140).
- [Kia23] Rabeaeh Kiaghadi. „Optimal Rescue Sequences in Disastrous Incidents“. In: *European Conference on Multi-Agent Systems*. Springer. 2023, pp. 534–539 (cit. on p. 164).

- [KJ23] Christopher de Koning and Anahita Jamshidnejad. „Hierarchical integration of model predictive and fuzzy logic control for combined coverage and target-oriented search-and-rescue via robots with imperfect sensors“. In: *Journal of Intelligent Robotic Systems* 107.3 (2023), p. 40 (cit. on p. 139).
- [KNP02] Marta Kwiatkowska, Gethin Norman, and David Parker. „PRISM: Probabilistic symbolic model checker“. In: *International Conference on Modelling Techniques and Tools for Computer Performance Evaluation*. Springer. 2002, pp. 200–204 (cit. on p. 99).
- [KNP06] M. Kwiatkowska, G. Norman, and D. Parker. „Quantitative analysis with the probabilistic model checker PRISM“. In: *Elsevier Electronic Notes in Theoretical Computer Science* 153.2 (2006), pp. 101–122 (cit. on p. 105).
- [KNP11] Marta Kwiatkowska, Gethin Norman, and David Parker. „PRISM 4.0: Verification of Probabilistic Real-Time Systems“. In: *International Conference on Computer Aided Verification*. Springer Berlin Heidelberg, 2011, pp. 585–591 (cit. on p. 105).
- [Kon10] Savas Konur. „Real-time and probabilistic temporal logics: An overview“. In: *arXiv preprint arXiv:1005.3200* (2010) (cit. on pp. 98, 99).
- [KPG25] P. Kumar, K. Pal, and M. Govil. „Comprehensive review of path planning techniques for unmanned aerial vehicles (uavs)“. In: *ACM Computing Surveys* 57.1 (2025), pp. 1–35 (cit. on p. 139).
- [Kur21] Hanna Kurniawati. „Partially observable markov decision processes (pomdps) and robotics“. In: *arXiv preprint arXiv:2107.07599* (2021) (cit. on p. 139).
- [KV00] Orna Kupferman and Moshe Y. Vardi. „An automata-theoretic approach to reasoning about infinite-state systems“. In: *International Conference on Computer Aided Verification*. Springer Berlin Heidelberg, 2000, pp. 36–52 (cit. on p. 25).
- [Kwi03] Marta Kwiatkowska. „Model checking for probability and time: from theory to practice“. In: *18th Annual IEEE Symposium of Logic in Computer Science, 2003. Proceedings*. 2003, pp. 351–360 (cit. on p. 98).
- [Kwi07] Marta Kwiatkowska. „Quantitative verification: models techniques and tools“. In: *Proceedings of the the 6th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering*. 2007, pp. 449–458 (cit. on p. 99).
- [LA11] Gabriel Luque and Enrique Alba. *Parallel genetic algorithms: Theory and real world applications*. Vol. 367. Springer, 2011 (cit. on p. 145).
- [Lea94] John Leach. *Survival psychology*. Springer, 1994 (cit. on p. 13).
- [LMS18] Helena Ramalhinho Lourenço, Olivier C. Martin, and Thomas Stützle. „Iterated Local Search: Framework and Applications“. In: *Handbook of Metaheuristics*. Cham: Springer International Publishing, 2018, pp. 129–168 (cit. on p. 146).
- [LMS22] Hoang Thanh Le, Martin Middendorf, and Yuhui Shi. „An Improvement Heuristic Based on Variable Neighborhood Search for Dynamic Orienteering Problems with Changing Node Values and Changing Budgets“. In: *SN Computer Science* 3.4 (2022), p. 326 (cit. on p. 146).

- [LP10] John Lygeros and Maria Prandini. „Stochastic hybrid systems: a powerful framework for complex, large scale applications“. In: *European Journal of Control* 16.6 (2010), pp. 583–594 (cit. on p. 43).
- [LS13] L. Liu and D. A. Shell. „Optimal Market-based Multi-Robot Task Allocation via Strategic Pricing“. In: *Robotics: Science and Systems*. 2013 (cit. on p. 147).
- [Mac11] Kathryn Macarthur. „Multi-agent coordination for dynamic decentralised task allocation“. PhD thesis. University of Southampton, 2011 (cit. on p. 140).
- [Mac96] Adrian Mackenzie. „Undecidability: The history and time of the universal Turing machine“. In: *Configurations* 4.3 (1996), pp. 359–379 (cit. on p. 24).
- [Mah+22] A. Mahmoodi, M. Jasemi Zergani, L. Hashemi, and M. Jafari. „Analysis of optimized response time in a new disaster management model by applying metaheuristic and exact methods“. In: *Smart and Resilient Transportation* 1.1 (2022), pp. 1–18 (cit. on p. 140).
- [Mai+16] Holger R. Maier, Joseph H. A. Guillaume, Hedwig van Delden, et al. „An uncertain future, deep uncertainty, scenarios, robustness and adaptation: How do they fit together?“. In: *Environmental modelling software* 81 (2016), pp. 154–164 (cit. on p. 140).
- [Mal+25] Vineetha Malathi, Pramod Sreedharan, Rthuraj P R, et al. „Decision-Making for Path Planning of Mobile Robots Under Uncertainty: A Review of Belief-Space Planning Simplifications“. In: *Robotics* 14.9 (2025) (cit. on p. 138).
- [Mao+20] Weichao Mao, Kaiqing Zhang, Erik Miehling, and Tamer Başar. „Information state embedding in partially observable cooperative multi-agent reinforcement learning“. In: *2020 59th IEEE Conference on Decision and Control (CDC)*. 2020, pp. 6124–6131 (cit. on p. 139).
- [Mat96] Yuri Matiyasevich. „HILBERT’S TENTH PROBLEM: What can we do with Diophantine equations?“. In: (1996) (cit. on p. 75).
- [MBS06] Anthony G. Macintyre, Joseph A. Barbera, and Edward R. Smith. „Surviving collapsed structure entrapment after earthquakes: a “time-to-rescue” analysis“. In: *Prehospital and disaster medicine* 21.1 (Feb. 2006), pp. 4–17 (cit. on p. 13).
- [Min66] Marvin Lee Minsky. *Computation: Finite and Infinite Machines*. Engle. 1966 (cit. on p. 71).
- [MM24] David Mansfield and Allahyar Montazeri. „A survey on autonomous environmental monitoring approaches: towards unifying active sensing and reinforcement learning“. In: *Frontiers in Robotics and AI* 11 (2024), p. 1336612 (cit. on p. 138).
- [Moh07] Ibrahim Mohamed Shaluf. „Disaster types“. In: *Disaster Prevention and Management: An International Journal* 16.5 (2007), pp. 704–717 (cit. on p. 5).
- [MPR25] Benjamin Monmege, Julie Parreaux, and Pierre-Alain Reynier. „Decidability of one-clock weighted timed games with arbitrary weights“. In: *Logical Methods in Computer Science* 21 (2025) (cit. on p. 72).
- [MS23] Samantha Montano and Amanda Savitt. „Revisiting emergencies, disasters, & catastrophes: Adding duration to the hazard event classification“. In: *International Journal of Mass Emergencies & Disasters* 41.2-3 (2023), pp. 259–278 (cit. on p. 5).

- [Nat25] National Oceanic and Atmospheric Administration (NOAA), Office of Response and Restoration. *Office of Response and Restoration*. <https://response.restoration.noaa.gov/>. Accessed: 26 Sep 2025. 2025 (cit. on p. 7).
- [NB11] Eric Neumayer and Fabian Barthel. „Normalizing economic loss from natural disasters: A global analysis“. In: *Global Environmental Change* 21.1 (Feb. 2011), pp. 13–24 (cit. on p. 6).
- [Nea97] David M. Neal. „Reconsidering the phases of disaster“. In: *International Journal of Mass Emergencies Disasters* 15.2 (1997), pp. 239–264 (cit. on p. 5).
- [New+10] Craig D. Newgard, Robert H. Schmicker, Jerris R. Hedges, et al. „Emergency medical services intervals and survival in trauma: assessment of the “golden hour” in a North American prospective cohort“. In: *Annals of Emergency Medicine* 55.3 (Mar. 2010), pp. 235–246 (cit. on p. 12).
- [New+15] Craig D. Newgard, Eric N. Meier, Eileen M. Bulger, et al. „Revisiting the “Golden Hour”: An Evaluation of Out-of-Hospital Time in Shock and Traumatic Brain Injury“. In: *Annals of Emergency Medicine* 66.1 (2015), 30–41.e3 (cit. on p. 13).
- [Noo+25] Erfaun Noorani, Zachary Serlin, Ben Price, and Alvaro Velasquez. „From abstraction to reality: DARPA’s vision for robust sim-to-real autonomy“. In: *AI Magazine* 46.2 (2025), e70015 (cit. on p. 140).
- [Nor98] James R. Norris. *Markov Chains*. Cambridge Series in Statistical and Probabilistic Mathematics 2. Cambridge University Press, 1998 (cit. on p. 42).
- [NPS13] Gethin Norman, David Parker, and Jeremy Sproston. „Model checking for probabilistic timed automata“. In: *Formal methods in system design* 43 (2013), pp. 164–190 (cit. on p. 71).
- [NY16] Ilan Noy and Rio Yonson. „A survey of the theory and measurement of economic vulnerability and resilience to natural hazards“. In: *Research Papers in Economics* (2016) (cit. on pp. 4, 5).
- [Oli09] J. M. de Oliveira Mendes. „Social vulnerability indexes as planning tools: beyond the preparedness paradigm“. In: *Journal of Risk Research* 12.3 (2009), pp. 445–460 (cit. on p. 7).
- [Omi+17] Shayegan Omidshafiei, Jason Pazis, Christopher Amato, Jonathan P. How, and John Vian. „Deep decentralized multi-task multi-agent reinforcement learning under partial observability“. In: *International Conference on Machine Learning*. 2017, pp. 2681–2690 (cit. on p. 139).
- [Ong+10] Sylvie CW Ong, Shao Wei Png, David Hsu, and Wee Sun Lee. „Planning under uncertainty for robotic tasks with mixed observability“. In: *The International Journal of Robotics Research* 29.8 (2010), pp. 1053–1068 (cit. on p. 139).
- [Ozi+22] Leonid Ozirkovskyy, Bohdan Volochiy, Oleksandr Shkiliuk, Mykhailo Zmysnyi, and Pavlo Kazan. „Functional safety analysis of safety-critical system using state transition diagram“. In: *Radioelectronic and computer systems* 2 (2022), pp. 145–158 (cit. on p. 25).
- [Ped+09] P. Peduzzi, H. Dao, C. Herold, and F. Mouton. „Assessing global exposure and vulnerability towards natural hazards: the Disaster Risk Index“. In: *Natural Hazards and Earth System Sciences* 9.4 (2009), pp. 1149–1159 (cit. on p. 3).

- [Ped19] Pascal Peduzzi. „The Disaster Risk, Global Change, and Sustainability Nexus“. In: *Sustainability* 11 (4 2019), p. 957 (cit. on p. 4).
- [Pen21] Neiler Medina Pena. „Adaptive Disaster Risk Assessment Combining Multi-Hazards with Socioeconomic Vulnerability and Dynamic Exposure“. In: (2021) (cit. on p. 4).
- [Pet+22] Geoffrey Pettet, Hunter Baxter, Sayyed Mohsen Vazirizade, et al. „Designing Decision Support Systems for Emergency Response: Challenges and Opportunities“. In: (2022). arXiv: 2202.11268 [cs.AI] (cit. on p. 9).
- [Pet99] Paul Pettersson. *Modelling and verification of real-time systems using timed automata: theory and practice*. Department of Computer systems, Univ., 1999 (cit. on p. 25).
- [PGM17] Victor Pillac, Christelle Guéret, and Andrés-L. Medaglia. „A fast reoptimization approach for the dynamic technician routing and scheduling problem“. In: *Recent developments in metaheuristics*. Cham: Springer International Publishing, 2017, pp. 347–367 (cit. on p. 146).
- [PP15] Lucy Pearson and Mark Pelling. „The UN Sendai framework for disaster risk reduction 2015–2030: Negotiation process and prospects for science and practice“. In: *Journal of Extreme Events* 2.01 (2015), p. 1571001 (cit. on p. 6).
- [PR18] David Pisinger and Stefan Ropke. „Large neighborhood search“. In: *Handbook of metaheuristics*. Cham: Springer International Publishing, 2018, pp. 99–127 (cit. on p. 146).
- [PR20] R. Patel and E. Rudnick. „Decentralized task allocation in multi-agent systems using a decentralized genetic algorithm“. In: *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. 2020, pp. 1–8 (cit. on p. 140).
- [Pu+25] F. Pu, Z. Li, Y. Wu, C. Ma, and R. Zhao. „Recent Advances in Disaster Emergency Response Planning: Integrating Optimization, Machine Learning, and Simulation“. In: *arXiv preprint arXiv:2505.03979* (2025) (cit. on p. 139).
- [Put05] Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 2005 (cit. on pp. 42, 72, 88).
- [QSW17] Karin Quaas, Mahsa Shirmohammadi, and James Worrell. „Revisiting reachability in timed automata“. In: *2017 32nd Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*. 2017, pp. 1–12 (cit. on p. 34).
- [RBL21] Hafiz Tayyab Rauf, Waqas Haider Khan Bangyal, and M. Ikramullah Lali. „An adaptive hybrid differential evolution algorithm for continuous optimization and classification problems“. In: *Neural Computing and Applications* 33.17 (2021), pp. 10841–10867 (cit. on p. 148).
- [Rob+15] Mark Roberts, Swaroop Vattam, Ronald Alford, et al. „Goal reasoning to coordinate robotic teams for disaster relief“. In: *Proceedings of ICAPS-15 PlanRob Workshop*. Citeseer, 2015, pp. 127–138 (cit. on p. 140).
- [Roy+25] Rajarshi Roy, Yash Pote, David Parker, and Marta Kwiatkowska. „Learning probabilistic temporal logic specifications for stochastic systems“. In: *arXiv preprint arXiv:2505.12107* (2025) (cit. on p. 99).

- [RR10] Mauricio G. C. Resende and Celso C. Ribeiro. „Greedy Randomized Adaptive Search Procedures: Advances, Hybridizations, and Applications“. In: *Handbook of Metaheuristics*. Boston, MA: Springer, 2010, pp. 283–319 (cit. on p. 147).
- [RR13] Mauricio GC Resende and Celso C. Ribeiro. „GRASP: Greedy Randomized Adaptive Search Procedures“. In: *Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques*. Boston, MA: Springer US, 2013, pp. 287–312 (cit. on p. 147).
- [Sch01] Lothar M. Schmitt. „Theory of genetic algorithms“. In: *Theoretical Computer Science* 259.1-2 (2001), pp. 1–61 (cit. on p. 145).
- [She+05] Mikhail Sheremet, Dmitry Tishkovsky, Frank Wolter, and Michael Zakharyashev. „Comparative similarity, tree automata, and diophantine equations“. In: *Logic for Programming, Artificial Intelligence, and Reasoning: 12th International Conference, LPAR 2005, Montego Bay, Jamaica, December 2-6, 2005. Proceedings 12*. Springer, 2005, pp. 651–665 (cit. on p. 71).
- [ŞM22] Elif Dilara Şeker and Michelle Maris. „Equivalent hazard magnitude scale“. In: *Natural Hazards and Earth System Sciences* 22 (12 2022), pp. 4103–4118 (cit. on p. 5).
- [Spa12] M. T. J. Spaan. „Partially observable Markov decision processes“. In: *Planning and Learning in Games*. Ed. by Springer. Springer, 2012, pp. 387–415 (cit. on p. 139).
- [SPG21] Jose Ricardo Sanchez-Ibanez, Carlos J Perez-del-Pulgar, and Alfonso García-Cerezo. „Path planning for autonomous mobile robots: A review“. In: *Sensors* 21.23 (2021), p. 7898 (cit. on p. 158).
- [Spr01] Jeremy Sproston. „Model Checking of Probabilistic Timed and Hybrid Systems (PhD Thesis)“. English. PhD thesis. Jan. 2001 (cit. on pp. 43, 54, 73).
- [Spr17] Jeremy Sproston. „Probabilistic Timed Automata with Clock-Dependent Probabilities“. In: *Reachability Problems*. Ed. by Matthew Hague and Igor Potapov. Springer, 2017, pp. 144–159 (cit. on pp. 27, 32, 43, 52, 72, 73).
- [Spr21] Jeremy Sproston. „Probabilistic timed automata with one clock and initialised clock-dependent probabilities“. In: *Logical Methods in Computer Science* 17 (2021) (cit. on p. 35).
- [Spr24] Jeremy Sproston. „Clock-Dependent Probabilistic Timed Automata with One Clock and No Memory“. In: *Formal Methods and Software Engineering*. Ed. by Kazuhiro Ogata, Dominique Mery, Meng Sun, and Shaoying Liu. Singapore: Springer Nature Singapore, 2024, pp. 70–84 (cit. on pp. 35, 72).
- [SS04] Tobias Schuele and Klaus Schneider. „Bounded model checking of infinite state systems: Exploiting the automata hierarchy“. In: *Proceedings. Second ACM and IEEE International Conference on Formal Methods and Models for Co-Design, 2004. MEMOCODE'04*. 2004, pp. 17–26 (cit. on p. 25).
- [SSP17] Eric Schneider, Elizabeth I. Sklar, and Simon Parsons. „Mechanism selection for multi-robot task allocation“. In: *Conference Towards Autonomous Robotic Systems*. Cham: Springer International Publishing, 2017, pp. 421–435 (cit. on p. 147).

- [SSW24] Andrew Scoones, Mahsa Shirmohammadi, and James Worrell. „Reachability for Multi-Priced Timed Automata with Positive and Negative Rates“. In: *arXiv preprint arXiv:2407.18131* (2024) (cit. on p. 71).
- [Ste+24] Shirly Stephen, Mark Schildhauer, Krzysztof Janowicz, et al. „The HIP Ontology: a formal framework to support disaster risk reduction and management“. In: *14th International Conference on Formal Ontology in Information Systems (FOIS 2024) (FOIS Ontology showcase Track)*. 2024 (cit. on p. 8).
- [Świ+23] Maciej Świechowski, Konrad Godlewski, Bartosz Sawicki, and Jacek Mańdziuk. „Monte Carlo tree search: A review of recent modifications and applications“. In: *Artificial Intelligence Review* 56.3 (Mar. 2023), pp. 2497–2562 (cit. on p. 148).
- [Tal09] El-Ghazali Talbi. *Metaheuristics: from design to implementation*. John Wiley Sons, 2009 (cit. on p. 141).
- [Tha+24] N. K. Thakre, D. Nimma, A. V. Turukmane, et al. „Dynamic Path Planning for Autonomous Robots in Forest Fire Scenarios Using Hybrid Deep Reinforcement Learning and Particle Swarm Optimization“. In: *International Journal of Advanced Computer Science Applications* 15.9 (2024) (cit. on p. 139).
- [TK21] David Teh and Tehmina Khan. „Types, Definition and Classification of Natural Disasters and Threat Level“. In: Springer, Cham, 2021, pp. 27–56 (cit. on p. 5).
- [TS15] Pan Tang and Geoffrey Qiping Shen. „Decision-making model to generate novel emergency response plans for improving coordination during large-scale emergencies“. In: *Knowledge-Based Systems* 90 (Dec. 2015), pp. 111–128 (cit. on p. 9).
- [TTK23] Oguz Emrah Turgut, Mert Sinan Turgut, and Erhan Kirtepe. „A systematic review of the emerging metaheuristic algorithms on solving complex optimization problems“. In: *Neural Computing and Applications* 35.19 (2023), pp. 14275–14378 (cit. on p. 144).
- [Tur37] A. M. Turing. „On Computable Numbers, with an Application to the Entscheidungsproblem“. In: *Proceedings of the London Mathematical Society* s2-42.1 (Jan. 1937), pp. 230–265 (cit. on p. 71).
- [UN23] UN. „Understanding risk to people, the planet and prosperity“. In: United Nations, 2023, pp. 16–73 (cit. on p. 4).
- [Uni22] United Nations Office for Disaster Risk Reduction. *Global Assessment Report on Disaster Risk Reduction 2022*. Geneva, 2022 (cit. on pp. 3, 5, 7–9).
- [Uni25] United Nations Office for Disaster Risk Reduction (UNDRR). *PreventionWeb — Knowledge platform for disaster risk reduction*. <https://www.preventionweb.net/>. Accessed: 26 Sep 2025. 2025 (cit. on pp. 6, 7).
- [VB94] Johan Van Benthem and Jan A. Bergstra. „Logic of transition systems“. In: *Journal of Logic, Language and Information* 3.4 (1994), pp. 247–283 (cit. on p. 26).
- [VBB14] Christian Von Lüken, Benjamin Barän, and Carlos Brizuela. „A survey on multi-objective evolutionary algorithms for many-objective problems“. In: *Computational optimization and applications* 58.3 (2014), pp. 707–756 (cit. on p. 145).

- [Vij22] Rishika Vij. „Types of Disasters“. In: *Management of Animals in Disasters*. Ed. by Subhash Verma and Hansen Thambi Prem. Singapore: Springer Nature Singapore, 2022, pp. 3–14 (cit. on p. 5).
- [Vik16] P. A. Vikhar. „Evolutionary algorithms: A critical review and its future prospects“. In: *2016 International Conference on Global Trends in Signal Processing, Information Computing and Communication (ICGTSPICC)*. Dec. 2016 (cit. on p. 150).
- [Wan+22] Shengli Wang, Youjiang Liu, Yongtao Qiu, and Jie Zhou. „Consensus-based decentralized task allocation for multi-agent systems and simultaneous multi-agent tasks“. In: *IEEE Robotics and Automation Letters* 7.4 (Nov. 2022), pp. 12593–12600 (cit. on p. 140).
- [Wei00] Klaus Weihrauch. *Computable Analysis — An Introduction*. Texts in Theoretical Computer Science. Springer, 2000 (cit. on p. 59).
- [Wex+14] Felix Wex, Guido Schryen, Stefan Feuerriegel, and Dirk Neumann. „Emergency response in natural disaster management: Allocation and scheduling of rescue units“. In: *European Journal of Operational Research* 235.3 (2014), pp. 639–651 (cit. on p. 12).
- [Wol02] Karl Erich Wolff. „Interpretation of Automata in Temporal Concept Analysis“. In: *International Conference on Conceptual Structures*. Springer Berlin Heidelberg, 2002, pp. 341–353 (cit. on p. 24).
- [Wor25] World Health Organization. *World Health Organization (WHO)*. <https://www.who.int/>. Accessed: 26 Sep 2025. 2025 (cit. on p. 6).
- [WRÁ23] Lisa Willemsen, Anne Remke, and Erika Ábrahám. „Comparing Two Approaches to Include Stochasticity in Hybrid Automata“. In: *International Conference on Quantitative Evaluation of Systems*. Springer. 2023, pp. 238–254 (cit. on p. 49).
- [Wu+21] J. Wu, C. Song, J. Ma, J. Wu, and G. Han. „Reinforcement learning and particle swarm optimization supporting real-time rescue assignments for multiple autonomous underwater vehicles“. In: *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 51.10 (2021), pp. 6400–6412 (cit. on p. 140).
- [Wu+22] Jiehong Wu, Chengxin Song, Jian Ma, Jinsong Wu, and Guangjie Han. „Reinforcement Learning and Particle Swarm Optimization Supporting Real-Time Rescue Assignments for Multiple Autonomous Underwater Vehicles“. In: *IEEE Transactions on Intelligent Transportation Systems* 23.7 (2022), pp. 6807–6820 (cit. on p. 140).
- [WW22] Remy Wehbe and Ryan K. Williams. *Probabilistically Resilient Multi-Robot Informative Path Planning*. 2022. arXiv: 2206.11789 [cs.R0] (cit. on p. 55).
- [XDI03] Gaoyan Xie, Zhe Dang, and Oscar H Ibarra. „A solvable class of quadratic diophantine equations with applications to verification of infinite-state systems“. In: *Automata, Languages and Programming: 30th International Colloquium, ICALP 2003 Eindhoven, The Netherlands, June 30–July 4, 2003 Proceedings 30*. Springer. 2003, pp. 668–680 (cit. on p. 71).
- [Yan+20] Zhou Yang, Long Nguyen, Jiazhen Zhu, et al. „Coordinating disaster emergency response with heuristic reinforcement learning“. In: *2020 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*. 2020, pp. 565–572 (cit. on p. 141).

- [Yan+23] Liwei Yang, Ping Li, Song Qian, et al. „Path planning technique for mobile robots: A review“. In: *Machines* 11.10 (Oct. 2023), p. 980 (cit. on p. 140).
- [Yew+19] Ying Ying Yew, Rafael Castro Delgado, David James Heslop, and Pedro Arcos González. „The Yew Disaster Severity Index: a new tool in disaster metrics“. In: *Prehospital and disaster medicine* 34.1 (2019), pp. 8–19 (cit. on p. 5).
- [YJY20] Yusen Ye, Wen Jiao, and Hong Yan. „Managing relief inventories responding to natural disasters: Gaps between practice and literature“. In: *Production and Operations Management* 29.4 (2020), pp. 807–832 (cit. on p. 140).
- [YY11] Bin Yu and Zhong Zhen Yang. „An ant colony optimization model: The period vehicle routing problem with time windows“. In: *Transportation Research Part E: Logistics and Transportation Review* 47.2 (2011), pp. 166–181 (cit. on p. 147).
- [Zha+22] Y. Zhang, R. Bai, R. Qu, C. Tu, and J. Jin. „A deep reinforcement learning based hyper-heuristic for combinatorial optimisation with uncertainties“. In: *European Journal of Operational Research* 300.1 (2022), pp. 1–15 (cit. on p. 141).
- [Zho+18] H. Zhou, A. Taal, S. Koulouzis, et al. „Dynamic real-time infrastructure planning and deployment for disaster early warning systems“. In: *Computational Science – ICCS 2018*. Ed. by P. M. A. Sloot, G. D. van Albada, A. G. Hoekstra, and P. Coveney. Springer International Publishing, 2018, pp. 543–556 (cit. on p. 5).
- [Zho+19] Xin Zhou, Weiping Wang, Tao Wang, Yonglin Lei, and Fangcheng Zhong. „Bayesian reinforcement learning for multi-robot decentralized patrolling in uncertain environments“. In: *IEEE Transactions on Vehicular Technology* 68.12 (2019), pp. 11691–11703 (cit. on p. 139).
- [ZN10] Lijun Zhang and Martin R Neuhäuser. „Model checking interactive Markov chains“. In: *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*. Springer. 2010, pp. 53–68 (cit. on p. 72).

List of Figures and Tables

4.1	A simple Deterministic Finite Automaton over $\{0, 1\}$ accepting strings that end with 1.	24
4.2	A simple state-transition diagram for a rescue robot.	26
5.1	Formal Models' Comparison in terms of Expressiveness required by Rescue Mission	45
5.1	Hierarchy of a subset of the modeling frameworks related to our IMCCs, ordered by expressiveness (based on a similar hierarchy taken from [Har15]).	48
6.1	An example of an Interactive Markov Agent with Clocks.	52
7.1	An IMCC encoding a Diophantine equation problem	76
9.1	Patients' IMCC model, to be instantiated for $i \in \{1, 2\}$	102
9.2	Robot's IMCC model in a 2-Patient Rescue Mission	103
9.3	Rescue mission for one robot and two patients, IMCC model	104
9.4	Time progress module in PRISM.	109
9.5	Transition diagram for the single-robot, two-patient decision process in PRISM.	110
9.6	State-transition diagram for a single robot in PRISM.	111
9.7	State-transitions for patients in PRISM	112
9.8	Random priority assignment as decision-making conflict resolution in PRISM.	114
9.9	PRISM synthesis for a mission with one robot and four patients	116
9.10	PRISM synthesis for a mission with two robots and four patients	117
9.11	PRISM interactive simulation (Trace 1) — 1 Robot, 4 Patients	118
9.12	PRISM interactive simulation (Trace 1) — continuation	119
9.13	PRISM interactive simulation (Trace 1) — continuation	120
9.14	PRISM interactive simulation (Trace 2) — 2 Robot, 4 Patients	121
9.15	PRISM interactive simulation (Trace 2) — continuation	122
9.16	PRISM interactive simulation (Trace 2) — continuation	123
9.17	State-transitions for patients in MoDeST	127
9.18	State-transition diagram for a single robot in MoDeST	128
9.19	MoDeST Synthesis for one Robot and three Patients' Mission (at 50 & 400 runs)	132

9.20	MoDeST Synthesis for two Robot and three Patients' Mission(at 50 & 400 runs)	133
11.1	Rescue mission decision-making approach in disastrous incident	155
11.2	Random location of Robot and Patients in MATLAB	157
11.3	An example of patients' survival Probability functions ($f_{Survival_{P_i}}$)	158
11.4	Snapshot sequence of robot's mission in rescuing initially 6 patients.	165
11.5	Simulation results of robot's strategy in different rescue-time and robot-speed settings.	173
11.6	Performance evaluation of <i>Severity</i> algorithm using <i>Lookup Curves</i> approach at different survival functions (setting #1) and various rescue times.	177
11.7	Performance evaluation of <i>Severity</i> algorithm using <i>Lookup Curves</i> approach at different survival functions (setting #1) and various robot speeds.	178
11.8	Performance evaluation of <i>Severity</i> algorithm using <i>Estimation</i> approach at different survival functions (setting #1) and various rescue times.	179
11.9	Performance evaluation of <i>Severity</i> algorithm using <i>Estimation</i> approach at different survival functions (setting #1) and various robot speed.	180
11.10	Performance evaluation of <i>Severity</i> algorithm using <i>Lookup Curves</i> approach at different survival functions (setting #2) and various rescue times.	181
11.11	Performance evaluation of <i>Severity</i> algorithm using <i>Lookup Curves</i> approach at different survival functions (setting #2) and various robot speeds.	182
11.12	Performance evaluation of <i>Severity</i> algorithm using <i>Estimation</i> approach at different survival functions (setting #2) and various rescue times.	183
11.13	Performance evaluation of <i>Severity</i> algorithm using <i>Estimation</i> approach at different survival functions (setting #2) and various robot speeds.	184

Acronyms

Acronyms	Meaning
ACO	Ant Colony Optimization
ALNS	Adaptive Large Neighborhood Search
CCTV	Closed-circuit Television
cd-PTA	Clock-dependent Probabilistic Timed Automata
CMA-ES	Covariance Matrix Adaptation Evolution Strategy
CPS	Cyber Physical Systems
CSG	Concurrent Stochastic Game
CSL	Continuous Stochastic Logic
CTL	Computation Tree Logic
CTMC	Continuous-Time Markov Chain
DALYs	Disability-adjusted Life Years
DE	Differential Evolution
DSS	Disaster Severity Scale
DTMC	Discrete-Time Markov chain
ETA	Estimated Time of Arrival
FEMA	Federal Emergency Management Agency
GA	Genetic Algorithm
GRASP	Greedy Randomized Adaptive Search Procedure
GSMP	Generalized Semi-Markov Processes
HA	Hybrid Automata
HAZUS	Hazards U.S.
HIPs	Hazard Information Profiles
I ² MC	Inhomogeneous Interactive Markov Chains
ICTMC	Time-Inhomogeneous Continuous-Time Markov Chains
ICU	Intensive Care Unit
IGSMP	Interactive Generalized Semi-Markov Processes
ILS	Iterated Local Search
IMC	Interactive Markov Chain
IMCC	Interactive Markov Chain with Clocks
IMDP	Interval Markov Decision Processes

LNS	Large Neighborhood Search
LTL	Linear Temporal Logic
LTS	Labelled Transition System
MA	Markov Automata
MARL	Multi Agent Reinforcement Learning
MC	Markov Chain
MCTS	Monte-Carlo Tree Search
MDP	Markov Decision Processes
MOEA	Multi-Objective Evolutionary Algorithms
MRTA	Multi-Robot Task Allocation
ODE	Ordinary Differential Equation
PCTL	Probabilistic Computation Tree Logic
PHA	Probabilistic Hybrid Automata
pLTL	Probabilistic Linear Temporal Logic
POMDP	Partially Observable Markov Decision Processes
PSO	Particle Swarm Optimization
PTA	Probabilistic Timed Automata
QoS	Quality of Service
RGSMF	Regenerative Generalized Semi-Markov Processes
RL	Reinforcement Learning
rPATL	Reward Probabilistic Alternating-time Temporal Logic
SA	Stochastic Automata
SAR	Search and Rescue
SHA	Stochastic Hybrid Automata
SHS	Stochastic Hybrid System
SMC	Statistical Model Checking
SoVI	Social Vulnerability Index
STA	Stochastic Timed Automata
TA	Timed Automata
TS	Tabu Search
TSP	Traveling Salesman Problem
UCT	Upper Confidence bounds applied to Trees
UNDRR	United Nations Office for Disaster Risk Reduction
VNS	Variable Neighborhood Search

Erklärung

Hiermit erkläre ich, dass ich diese Arbeit eigenständig verfasst und keine anderen als die angegebenen Hilfsmittel und Quellen benutzt habe. Ebenso versichere ich, dass diese Dissertation weder in ihrer Gesamtheit noch in Teilen einer anderen wissenschaftlichen Hochschule zur Begutachtung in einem Promotionsverfahren vorgelegen hat.

Declaration

I declare that I have written this thesis independently and that I have used only the resources indicated. I also affirm that this dissertation has not been submitted by me, in whole or in part, to any other university for assessment in a doctoral procedure.

Oldenburg, 19 January 2026

Rabeaeh Kiaghadi

