Testing Hypotheses in an Engineering Domain: Combining Static and Dynamic Analysis of Pneumatic Circuits

Janine Willms[1], Hermann Göhler[2], Claus Möbus[1,2]

[1]FB Informatik, University, D-26111 Oldenburg, Germany
[2]OFFIS Institute, Escherweg 2, D-26121 Oldenburg, Germany
E-Mail: {Janine.Willms,Hermann.Goehler,Claus.Moebus}@informatik.uni-oldenburg.de

Abstract: We want to describe the design of PULSE (Pneumatic Learning and Simulation Environment), an Intelligent Problem Solving Environment (IPSE) for pneumatic circuits based on a cognitive theory of knowledge acquisition (ISP-DL-Theory). PULSE offers tasks given as a textual description and a time-discrete kind of a distance-time diagram. It supports unconstrained design of pneumatic circuits. PULSE offers the possibility to test hypotheses about the correctness of student's proposals. This is achieved by combination of static and dynamic analysis. Dynamic analysis is done by model-checking and static analysis by abductive concept-based explanations. The dynamic analysis method is a complete procedure for checking the correctness of full-functioning circuits. But, because model-checking does not support early problem-solving phases (e.g. deliberation) there is a need for static analysis despite the fact that this is not complete.

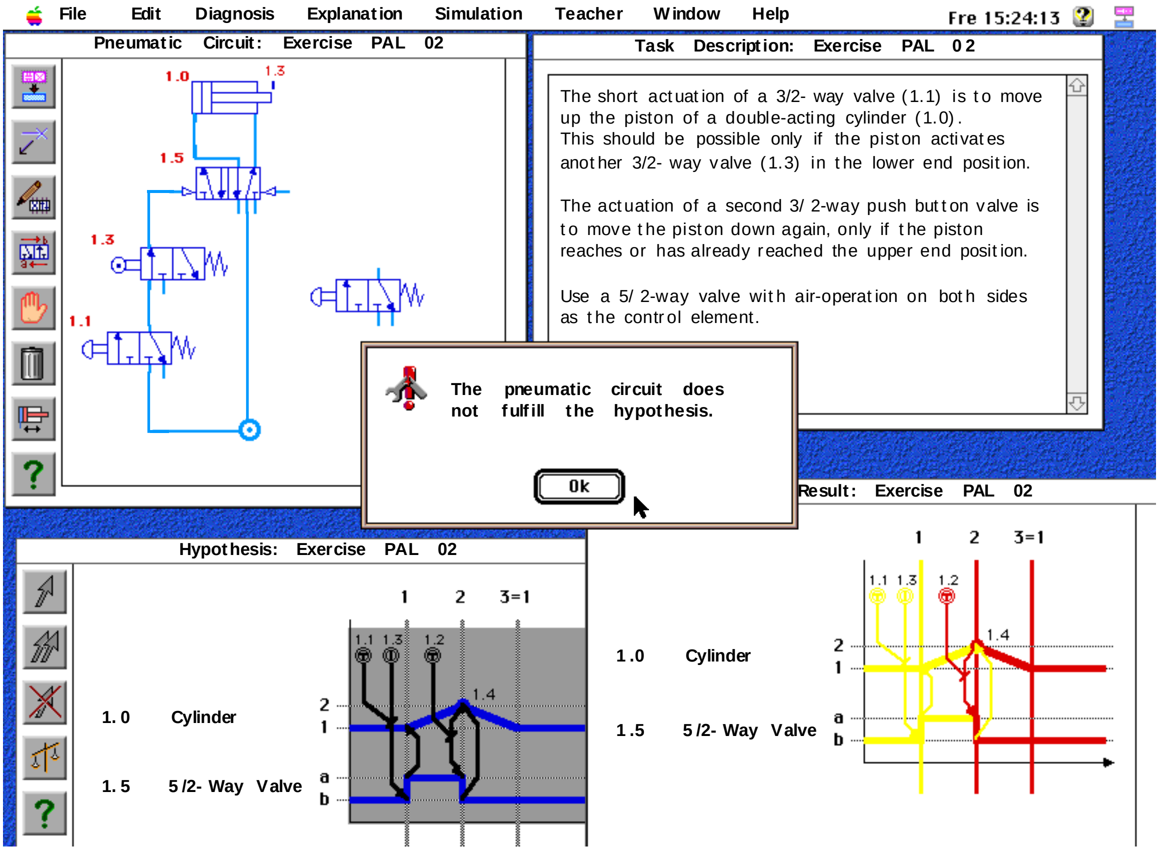
Keywords: intelligent problem solving environment, intelligent design environment, pneumatic circuits, hypotheses testing approach, ISP-DL-Theory, concept-based explanations, model-checking, dynamic and static analysis

1. Static and Dynamic Analysis

Intelligent Tutoring Systems, Explorative Problem Solving Environments, Intelligent Design Environments... whenever a student constructively explores a domain, the supporting system needs mechanisms for plan analysis. Some systems like SPADE [8] or PROUST [6] use a static analysis based on plans to examine the student's work and give correction or completion proposals. Other systems, like PETRI-HELP [9], which has been created in our working group, concentrate on the dynamic part of a solution proposal. The combination of both kinds of analysis seems to be fruitful though it can rarely be found. Recent developments of the tutoring sytem SYPROS [4,5] show an attempt to add the feature of dynamic analysis similar to PETRI-HELP to an originally static approach. We recognized from our experiences with PETRI-HELP that the model checking approach [1,7] is useful to check dynamic behaviour of a student's solution proposal, because model checking allows feedback on any kind of solution he might offer. Errors are explained by comparison of internal simulation results with the task specification. But this kind of system feedback is low-level and too fine grained. It does not refer to "higher" or more abstract concepts and therefore model checking does not support early planning stages (e.g. deliberation). In the development of PULSE (Pneumatic Learning and Simulation Environment) a static concept-based analysis was added to support the student's construction process with completion and correction proposals even when the circuit was not ready to be run in a simulation.

2. Introduction to PULSE

PULSE is an intelligent problem solving environment (IPSE) [11] in the domain of pneumatic circuits, which has been developed according to a cognitive theory of knowledge acquisition: the ISP-DL-Theory ("Impasse-Success-Problem-Solving-Driven-Learning") [10,11]. It will be used in seminars and courses of the DIHT (German Chamber of Industry and Commerce).



The short actuation of a 3/2- way valve (1.1) is to move up the piston of a double-acting cylinder (1.0). This should be possible only if the piston activates another 3/2- way valve (1.3) in the lower end position.

The actuation of a second 3/ 2-way push button valve is to move the piston down again, only if the piston reaches or has already reached the upper end position.

Use a 5/ 2-way valve with air-operation on both sides as the control element.

The pneumatic circuit does not fulfill the hypothesis.

Fig. 1: The model checker's feedback on a student‘s incorrect hypothesis

PULSE is designed as a learning and problem-solving environment. Tasks have been taken from a published database for paper-and-pencil exercises. Each task consists of a textual description, a time-discrete distance-time diagram reflecting the dynamic behavior of a pneumatic circuit. The distance-time diagram is divided into several time steps showing the activation and deactivation of pneumatic components and how they relate to each other. Emphasis was not put on the simulation aspects of PULSE but on the support of constructing circuits. So explanations based on simulation runs as introduced by the qualitative reasoning community [2,3] are desirable but not necessary for the task at hand. Consequently, it was not necessary to model the dynamics of a pneumatic circuit in detail by differential equations. The main focus was the support of unconstrained design: the hypothesis-testing approach.

3. The Hypothesis-Testing Approach

The student may construct any pneumatic circuit at will by arranging given pneumatic components in order to find a solution for the task. When he thinks that he has constructed an artifact that meets the requirements in the text and those given by the distance-time diagram, he may ask PULSE for an evaluation: he selects those parts of the distance-time diagram that he believes to be correctly represented. The correctness of a hypothesis is proved by calculating the complete state based behavior of the pneumatic circuit by generating a state-case graph and comparing its structure with the distance-time diagram of the task. This is called model-checking. Feedback is given by different coloring of the distance-time diagram. In so far the student may use PULSE as an experimental environment and explore any possible circuit. A visual simulation mode and a trace window further improve the explorative character of the system.

Model checking allows to analyze any solution proposal for correctness with respect to the dynamic parts of the specification. In so far it is complete, but it is not very suitable for generating explanations for the student. Therefore, in addition to the dynamic model checker, a static analysis based on several concept hierarchies was implemented. If the student knows these concepts already when working with PULSE, we hope he will experience less impasses. But, if the student has not acquired the knowledge to understand the task, PULSE allows him to gain this knowledge *just in time*. Therefore PULSE offers a simple introduction to pneumatic terms by texts and pictures and refers to these concepts when errors in the student's solution proposal are explained. In contrast to the dynamic model checker the static analysis is not inherently complete. Missing concepts in the system's knowledge base may cause incorrect feedback. A learning component for new concepts would therefore be necessary, if the structure of the tasks changes.

PULSE uses concepts for the following explanations:

- Object and functional concepts are introduced and described. The student can get the information needed *just in time* in the problem solving situation.
- Constraints for the current task hidden in the text or distance-time diagram are presented explicitely to the student. Furthermore it is explained by which rules and heuristics these constraints could be discovered.
- PULSE is able to analyze the student‘s hypothesis in order to refer to the concepts in the student‘s pneumatic circuit.
- PULSE compares the concepts of the user‘s hypothesis with the constraints of the current task and identifies missing or mixed up concepts and gives detailed information to the student, if he asks for it.
- PULSE contains a module to synthesize a pneumatic circuit as an internal reference that can be used for correction and completion proposals.

4. Summary

The essence of the development of PULSE was that model checking is valuable in a design environment for analyzing the dynamic aspects of a system because of its completeness. A mainly static analysis always struggles with typical problems known from incomplete plan recognition, e.g. missing or mixed up plans. On the other hand static analysis allows to comment on a student's solution in earlier planning stages and on a much higher level than model checking. Concept-based explanations may be used to criticize a students solution proposal even when it is not yet ready to be simulated by the model checker. A learning environment which is powerful to offer unconstrained design and support should therefore integrate both approaches.

PULSE is available as a product under WINDOWS 3.1, Windows 95 and Macintosh. It will be used in courses of the DIHT in near future.

See *http://www.offis.uni-oldenburg.de/projekte/pulse/* for further information.

References

[1] Clarke, E.M., Emerson, F.A. & Sistla, A.P, Automatic Verification of Finite-State Concurrent Systems Using Temporal Logic Specifications. ACM Transactions on Programming Languages and Systems, Vol. 8, No. 2, 244 - 263, 1986.

[2] de Kleer, J., Brown, J.S., A physics based on confluences. In: Bobrow, D.G. (ed.) Qualitative reasoning about Physical Systems. MIT Press, Cambridge, Massachusetts, 1984.

[3] Forbus, K., Stevens, A.L., Using qualitative simulation to generate explanations. BBN Report 4490. Bolt Beranek and Newman Inc. Cambridge, Massachusetts, 1981.

[4] Henning, U., Analyse eines Modells paralleler Prozesse für die Anwendung in einem intelligenten Lehrsystem, Dissertation, TU München, Institut für Informatik, 1994.

[5] Herzog, C., Syntaxorientierte vs. ablauforientierte Diagnose in intelligenten Programmier- umgebungen als Beispiel für den Einsatz konkurrierender Problemlöser. In: M. Thielscher, S.-E. Bornscheuer (eds.), Fortschritte der Künstlichen Intelligenz (KI 96), Dresden: Dresden University Press, 1996.

[6] Johnson, W.L., Intention-based Diagnosis of Novice Programming Errors. Los Altos: Morgan Kaufmann, 1986.

[7] Josko, B.,Veryfying the correctness of AADL modules using model checking. In: J.W. de Bakker, W.P. de Roever, G. Rozenberg (eds.), Proceedings REX-Workshop on stepwise refinement of distributed systems: models, formalisms, correctness. Berlin: Springer, LNCS 430, 386-400, 1990.

[8] Miller, M.L., Goldstein, I.P., SPADE: a grammar bases editor for planning and debugging programs. AI Lab Memo 386. Massachusetts Institute of Technology, Cambridge, Massachusetts, 1976.

[9] Möbus, C., Pitschke, K., Schröder, O., Towards the Theory-Guided Design of Help Systems for Programming and Modelling Tasks. In C. Frasson, G. Gauthier, G.I. McCalla (eds), Intelligent Tutoring Systems, Proceedings of the Second International Conference ITS 92, Montreal, Berlin: Springer, LNCS 608, 294 - 301, 1992.

[10] Möbus, C., Towards an Epistemology of Intelligent Problem Solving Environments: The Hypothesis Testing Approach. In: J. Greer (ed.), Artificial Intelligence in Education, Proceedings of AI-ED 95, Washington, D.C., August 16-19, 1995, Charlottesville: AACE, 138-145, 1995.

[11] Möbus, C., Towards an Epistemology on Intelligent Problem Solving Environments: The Hypothesis Testing Approach. In: Proceedings of EuroAIED 96, Lisbon, Portugal, Sept. 30 - Oct. 2, 1996