Hybrid Constraints in Automated Model Synthesis and Model Processing

K.-U. Leweling, B. Stein

Department of Mathematics and Computer Science—Knowledge-based Systems
University of Paderborn, D–33095 Paderborn, Germany
EMail: {leweling,stein}@uni-paderborn.de

Abstract

Both parametric design tasks and analysis tasks of technical systems have a similar problem setting: The structure of the system to be configured or analyzed is defined already. Within the parametric design task unknown values for component geometries have to be determined, while within the analysis task the system has to be completed with respect to missing physical quantities.

The tasks mentioned form hybrid constraint satisfaction problems, which may be solved by a generic procedure. However, if the no-function-in-structure principle holds, i. e., if the behavior of the entire system can be derived from the behavior of its parts, engineering semantics of model synthesis and simulation apply. As a result, not only domain knowledge can be exploited to solve the constraint satisfaction problem efficiently, but also instances of both types of problems can be tackled by the same problem solving approach: a sequence of intertwined model synthesis and simulation steps.

The paper in hand introduces this problem solving approach as a cycle comprising five generic steps and presents case studies of real life problems from the field of hydraulics, which illustrate its successful application.

Key words: Hybrid constraints, constraint processing, model synthesis, simulation, configuration, design

1 Introduction

Automated model synthesis and model processing are two key issues in supporting parametric design tasks as well as analysis tasks of technical systems, such as required for fault diagnosis or simulation for example. In many cases the structure of the target system is predefined and fixed. The task is to determine unknown values of technical parameters relating to the components' geometries (in case of parametric design) or to deduce a coherent system model by establishing unknown physical state quantities (in case of system analysis).

In both cases, we are confronted with a constraint satisfaction problem (CSP) [16] that is defined by (1) a structural description of the system S to be configured or analyzed, (2) a

Preprint submitted to Elsevier Preprint *October 19, 2007*

model fragment universe $m_1, ..., m_n$ of behavior models of the system's components, and (3) a vector of user requirements in the form of parameter-value pairs.

In the following, the resulting CSP is described in greater detail.

The structural description defines the components of S along with their possible interactions. Such a description can be viewed as a port-and-connector model, where adjacent ports stand for shared properties of the connected components [13]. For instance, a hydraulic cylinder c which is connected to a valve v interacts with this valve solely by means of the hydraulic fluid. Hence, in the related port-and-connector model there are two connections between c and v, representing the fluid's pressure P and flow Q respectively. Figure 1 depicts the example.

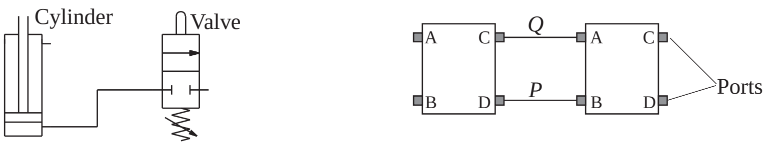

Figure 1: *A cylinder connected to a valve and the related port-and-connector model.*

A model fragment describes the behavior of a component in a particular operational mode or situation. [1] A pressure relief valve, for example, can either be in mode "open", "closed", or "controlling". If a component o has k operational modes, the set $M_o := \{m_{o_1}, ..., m_{o_k}\}$ fully specifies the behavior of o. From a physical point of view, each individual model fragment represents a necessary and complete behavior description for a certain situation, since a component can only be in one operational mode at some point of time. The following example shows a model fragment that describes a simple valve for the operational mode "open"; the first and second subscript in the notation indicates the component and the port respectively.

$$Q_{v_A} = Q_{v_B} \;\; \wedge \;\; P_{v_C} - P_{v_D} = \textit{signum}(Q_{v_A}) \cdot r_v \cdot Q_{v_A}^2$$

The port-and-connector model of the system S defines in which way a behavior model M_S for S is synthesized: If within the port-and-connector model two components o_1 and o_2 are connected, constraints are introduced for the shared properties. In the case of hydraulics, balance equations for the flow Q and unifying equations for the pressure P or the force F realize this job. Respecting the above example, where a cylinder c is connected to a valve v, the following connection constraints are introduced:

$$Q_{c_C} = -Q_{v_B} \;\; \wedge \;\; P_{c_D} = P_{v_B}$$

Observe that the structural description along with the set of model fragments define a *space* of possible models for system S: For each component o in the structural description, one

[1] The concept of model fragments has been introduced by Nayak [11]: "A model fragment is a set of independent equations that partially describe some physical phenomena." However, a significant difference to Nayak's work is the following. While Nayak employs model fragments to describe a component from different *views* (e. g. electrical or thermodynamical) or at different *levels of detail*, our concept of model fragments is used to distinguish between the different operational *modes* of a component (at a particular view and a fixed level of detail).

model fragment from the set M_o must be chosen. Exactly this choice point situation renders the synthesis of M_S a combinatorial problem, because

(1) there is rare knowledge that locally constrains the selection of model fragments, and
(2) whether or not a set of synthesized model fragments form a physically correct behavior model for S can only be verified by a simulation of the entire model M_S.

Solving the CSP means to synthesize a system model M_S that completely and coherently specifies the set of system parameters and quantities. In this regard the vector of user requirements prescribes value ranges or concrete values for particular variables.

Figure 2 illustrates the generic model synthesis and model processing approach on the basis of a hybrid constraint processing method for the underlying hybrid CSP.

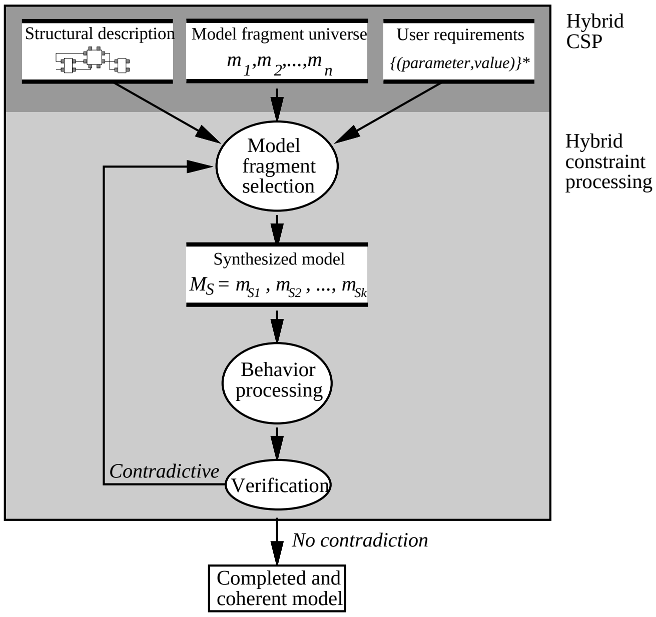

Figure 2: *From the target system we are given its structure, a model fragment universe comprising behavior models, and a set of desired requirements. The resulting CSP is solved by an intertwined loop of selecting, synthesizing, and simulating model fragments.*

Constraint processing, however, happens within two steps, model fragment selection and behavior processing, which is in effect a simulation.

(1) *Model Fragment Selection.* The first step selects and synthesizes a subset of model fragments $m_{i_1}, ..., m_{i_k}$. The model fragments are represented in terms of two kinds of constraints: (a) behavioral constraints, which describe physical relations to hold in that operational mode and which constitute systems of non-linear equations and inequations, (b) qualifying constraints, which describe the conditions under which a model fragment, i. e. a set of behavioral constraints, is valid.
(2) *Behavior Processing.* The second step evaluates the model's coherence and is achieved by a simulation of the selected behavior constraints. Behavior processing is carried out by local (interval) propagation as well as by procedures for solving systems of equations.

If the simulation reveals either a contradiction within the synthesized model or unfulfilled user requirements, the cycle of intertwined model synthesis and simulation is reiterated by modifying the model fragment set with respect to a certain choice point.

Taking that model synthesis and processing tasks form a CSP on mixed numeric and symbolic constraints, they may be solved by a generic constraint processing procedure. Typically, such a generic procedure will be less efficient than the outlined approach.

The outlined approach exploits the engineering view of model formulation and simulation, which provides heuristics and strategic knowledge of how model fragments are selected and processed. To make model synthesis a working concept, the subsumption of constraints to model fragments requires a context-free behavior description of the system components. This characteristic is also called "no-function-in-structure-principle", which says that the behavior of the entire system can be derived from—and solely depends on—the behavior of its individual components [3].

To illustrate the power of the hybrid constraint processing method, we present two popular applications in the field of hydraulics: (a tool for) parametric design of hydraulic drives (see Section 2) and an analysis and simulation tool for hydraulic systems (see Section 3). Though these are two very different applications, it will be clear that both problems can be solved by the same hybrid problem solving approach.

2 Case Study 1: Parametric Design of Hydraulic Drives

Hydraulic drives have a wide variety of applications and are usually tailored to specific customer's demands. Such drives typically have a fixed structure and consist of only a few components (Fig. 3). However, designing hydraulic drives is difficult for two reasons:

(a) Because of the strong physical interactions and interdependencies of the drive's components, the consequences of choosing a particular component variant cannot fully be estimated until the design is completed. For example, the natural frequency of the drive results from the interaction of all hydraulic parts of the system. To proceed with a partial design, certain parameters have to be guessed based on heuristics. Those guesses can turn out wrong and necessitate backtracking to a certain choice point.

(b) For each of the components, there is a large number of possible constructional variants and graduations in the component's geometry to choose from. These variants in fact constitute choice points in the design process. For example, changing the mounting position of the cylinder usually requires a different choice of the remaining components, due to the changed physical behavior of the cylinder.

Since hydraulic components are identified by their technical parameters, (e. g. a hydraulic cylinder is identified by its piston diameter, piston rod diameter, maximum stroke, etc.) the design task consists of determining the parameter sets of the components necessary to build the drive in a way that all of the given requirements are fulfilled. Stated another way, the parameter model of the drive yields an abstract description of a solution to this parametric design problem. Thus, the goal is to select and synthesize a parameter model of the drive from the model fragments which is coherent with all of its parts as well as with the customer's requirements.

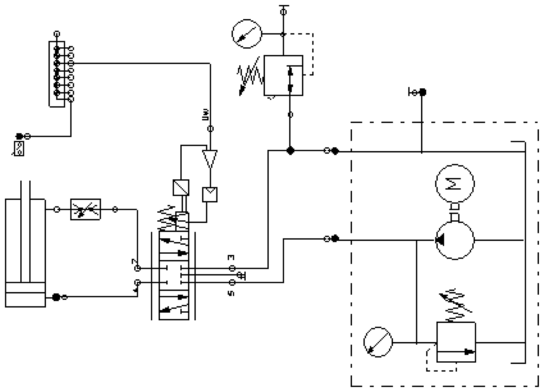

Figure 3: *Circuit schematic of a hydraulic drive.*

For each of a component's different operational modes there is a set of behavior constraints (i. e. a model fragment) stored in the system's knowledge base, along with qualifying constraints describing when which model fragment is to be activated. In conjunction with the constraints imposed to the overall device, the design task poses a hybrid CSP, which is solved by a hybrid constraint processing approach.

2.1 Hybrid Constraint Processing

In hybrid constraint processing within the model synthesis cycle five steps can be distinguished in the parametric design of hydraulic drives:

(1) *Component Selection.* Based on the parameter set derived so far (from the input requirements, from heuristic rules for otherwise unknown parameters), decide which component (e. g. cylinder or valve) is to be modified.

(2) *Choice Point Selection.* Several parameters constitute choice points in the configuration process, for example, the mounting position of the cylinder. Choice point selection means selecting a parameter value which leads to a qualitatively different model of the drive. Choice point selection can take either of the following two forms:

  a) *Geometry Selection.* Select a value for a geometry parameter of the chosen component, e. g. the piston rod diameter for a cylinder. This in effect replaces that component in the model. However, as long as a different component geometry does not entail a qualitative change in its behavior, the current model remains the same. Component geometry selection means constraint value selection and is tried first in the model synthesis process.

  b) *Mode Selection.* Behavior modes are choice points. Mode selection means to choose a definite mode for this component, by selecting a value for some parameter, such as the mounting position of the cylinder. Since different modes define different component behaviors, changing a parameter means changing the model fragment set in the synthesized model. So, mode selection means constraint set selection.

(3) *Synthesis.* Identify the active model selection constraints. Synthesize the local behavior models into a global model according to the given situation.

(4) *Simulation.* Simulate the synthesized behavior model by evaluating the behavior con-

straints. Simulation is performed in two ways: The first is to do a local propagation. The second comes into play when local propagation does not work. In that case, the underlying equation system is solved directly by algebraic procedures.

(5) *Verification and Modification.* If the synthesized model proves to be inconsistent or does not meet the demands, trace back to a choice point (see Step 2).

2.2 Realization

The system that supports hydraulic drive configuration is composed of two parts: the actual configuration engine, which runs in the background, and a user interface in the form of a standardized spreadsheet environment (Fig. 4). There are spreadsheets corresponding to the customer's requirement sheet and to the desired components' technical parameters.

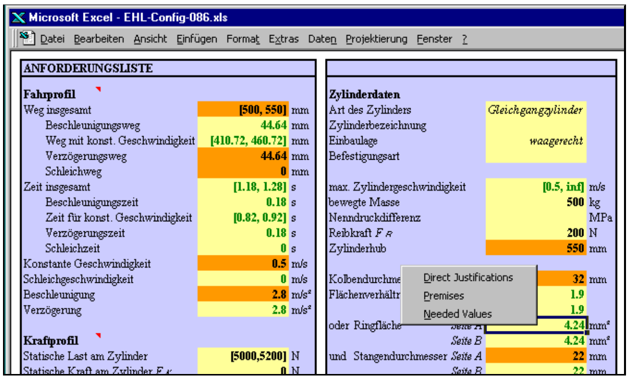

Figure 4: *User interface of the hydraulic drive configuration tool showing the customer's requirement sheet. In an interactive dialog, the system receives (interval) values for the required drive parameters and produces intervals for the unknown parameters. On request, the explanation facility displays for a derived value the grounding inputs as well as the inference chain in terms of involved constraints.*

The configuration process begins with the input of the customer's requirements to the hydraulic drive, for example, parameters concerning the drive's operation profile, force profile and performance demands. The system accepts as input parameters both exact values and intervals. This way, the system can handle inputs which are partially incomplete or only vaguely given by the customer. As a result, the system produces values for the unknown technical parameters of the hydraulic drive and its components.

The design task is a successive process of model synthesis and model modification: Whenever the engineer changes the value for a parameter, the configuration engine is invoked and the effects of this change on the other parameters are displayed instantly. The effects can be examined by the engineer and provide essential information as how to proceed.

The configuration engine realizes the above mentioned hybrid constraint processing approach by applying both interval propagation methods based on interval arithmetic on real-valued variable domains and numerical methods for solving systems of equations [12,5,8].

When developing the system, an important issue has been explainability of the results. For every derived value, the system can identify the underlying input parameters or display the inference chain that led to the derived value. This is achieved by combining constraint processing with a so-called truth maintenance system (TMS), which keeps track of the inference chains by bookkeeping the dependencies during propagation [7,4].

3 Case Study 2: Analysis of Hydraulic Systems

Loosely speaking, hydraulic systems analysis takes a circuit diagram as input and produces a behavior description of the entire circuit. Figure 5 illustrates this task.

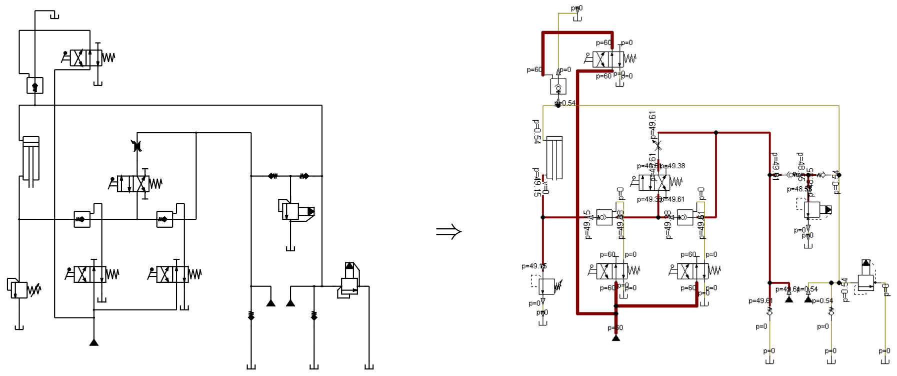

Figure 5: *A small hydraulic system before (left hand side) and during simulation (right hand side).*

Typically, the approach to large-scale circuit analysis is switching logic simulation and stationary simulation relying on nonlinear component models. The term "stationary" relates to the pressure, p, and, equivalently, to the flow, Q, and the first derivative of the piston position, $\dot{x}$.

Stationary simulation means to identify those points of the system in which all derivatives are equal to zero, the so-called steady-state points. Until a steady-state point is reached, the components of the system may incur several changes in their operational modes. These changes are triggered when $\dot{p} \neq 0$ or $\dot{Q} \neq 0$ and physical thresholds of particular components are passed. Examples of such thresholds are the switching pressure of relief valves (mode `open` or `closed`), the piloting pressure at way valves (mode `left`, `zero`, or `right`), or the flow velocity within throttle valves (mode `laminar_flow` or `turbulent_flow`).

Hence, to *directly* compute the desired stationary behavior of a fluidic circuit, assumptions that reason about the components' behavior modes must be made. [2]

Example. Given is a circuit as drawn in Figure 6, consisting of three cylinders with different loads, three pressure relief valves, `rv_1`, `rv_2`, and `rv_3`, and the necessary supply elements. The task is to determine (a) which of the cylinders will extend if the pump is switched on, and (b) the pressure p_x, at the bottom of the rightmost cylinder. The geometries, resistances, and pressure thresholds are given. The Figure shows the first steady-state point of the circuit after the pump has been switched on: Among others, the rightmost cylinder extends, and all relief valves are closed.

Figure 7 takes a closer look on what has happened until the first steady-state point is reached (at time t_6). In fact, while the relief valve `rv_1` remains closed when the pump is switched on, the relief valves `rv_2` and `rv_3` open and close, resulting in pressure drops as well

[2] Of course all steady-state points can be found by tracing the derivatives of the state variables. Note, however, that there are crucial points bound up with the necessary dynamic simulation: efficiency requirements, trade-offs between correct behavior and model precision, or transparency issues of the analysis.

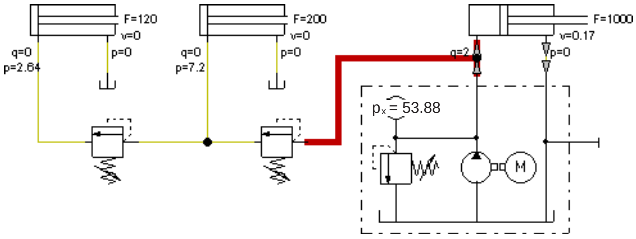

Figure 6: *Circuit with three coupled cylinders. The schematic indicates the first steady-state point after the pump is switched on.*

as stalled pressures between the middle cylinder and `rv_2` and the left cylinder and `rv_3` respectively. In all, five operational mode changes may be passed until the first stationary time interval is entered, presumed a more detailed dynamic simulation.

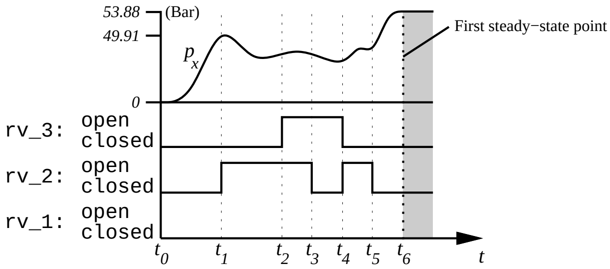

Figure 7: *A look at the dynamical development of* p_x.

Figure 8 shows the operational mode space of the three relief valves from the example. Here, m_{rv_a} and m_{rv_b} stand for the behavior models of a closed and open relief valve respectively; e. g. the set $\{m_{rv1_a}, m_{rv2_a}, m_{rv3_a}\}$ indicates that all valves are closed, which corresponds to the correct mode assignment for the first steady-state point at t_6 in the example. There exist 8 operational mode combinations relating to the relief valves.

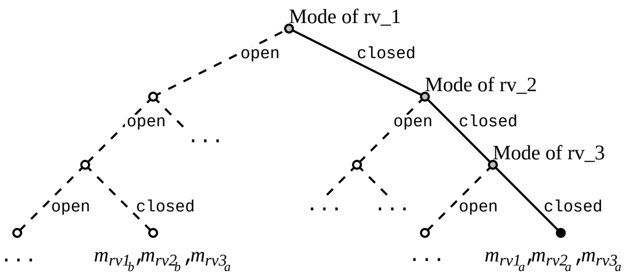

Figure 8: *Mode space of the three relief valves.*

Also the cylinders provide for two distinct modes—(a) an equilibrium state, m_{cyl_a}, where the balance of forces holds between the pressure, $\Delta p \cdot A$, the load, F, and the piston velocity, $k \cdot \dot{x}$, and (b) a stop mode, m_{cyl_b}, where the piston touches one side of the cylinder.

At time t_6 in the example, the extending cylinder is in mode m_{cyl_b} while the others are in mode m_{cyl_a}. Hence, the set $\{m_{rv1_a}, m_{rv2_a}, m_{rv3_a}, m_{cyl1_b}, m_{cyl2_a}, m_{cyl3_a}\}$ completely defines the model in the first steady-state point. The total mode space of the circuit contains $2^6 = 64$ elements.

The next subsection shows how the synthesis of completed and coherent model is tackled.

3.1 Hybrid Constraint Processing

The previous example has shown that even though a circuit diagram S has a correct and definite physical interpretation, not all physical quantities can be derived in an ad-hoc manner: Most components of S are not defined by a single model but a set of model fragments, from which the relevant one must be selected. This problem, called model synthesis here, consists of all steps that are necessary to set up a global behavior model which is both correct in a physical sense and *locally unique* (cf. [15]). [3]

Stated precisely, for each component o in S let $M_o = \{m_{o_1}, m_{o_2}, \ldots, m_{o_k}\}$ be comprised of the k model fragments of o. If a component o has a locally unique model, say, a pipe for instance, $|M_o| = 1$. Let $\mathcal{M}_S$ be the Cartesian product of the sets $M_o, o \in S$. $\mathcal{M}_S$ comprises models of the circuit S that can be synthesized at all, and thus, $\mathcal{M}_S$ defines the total synthesis search space.

As described at the outset, meta constraints, the so-called qualifying constraints, are used to reason about the operational modes and the respective model fragments m_{o_i} of a component o. Example:

IF x is of type relief_valve
  AND x is in state *open*
  THEN $m_{rv_a} := \{Q_A = Q_B, \ldots\}$ is valid

The IF-clause constitutes a qualifying constraint or model fragment qualifier, m_{rv_a} is one of the model fragments of the relief valve, and "$Q_A = Q_B$" is a single constraint of the model fragment m_{rv_a}. A qualifying constraint is called active if its conditions are fulfilled.

Given the concept of qualifying constraints, the search for a coherent model can be realized as a cycle of the following five steps:

(1) *Component Selection.* Select a component that possesses several operational modes, that is to say, model fragments.
(2) *Mode Selection.* Choose some mode for this component.
(3) *Model Synthesis.* Identify and evaluate active model qualifying constraints and synthesize the model fragments to a global model M_S. [4]
(4) *Simulation.* Simulate the synthesized model M_S by processing its behavior constraints.
(5) *Verification and Modification.* In case of physical inconsistencies, trace back to a choice point, formulate additional synthesis restrictions in the form of nogoods (see

[3] There is particular research in connection with model composition problems (cf. [10], [6], [9]). Note that the mentioned as well as related work focuses on the construction or selection of adequate models with respect to different tasks (simulation, diagnosis) or different levels of granularity. This is not the case here: Although both the task and the level of granularity are given, there is a synthesis problem, which results from the indeterminacy of local behavior descriptions.

[4] This type of inference is sometimes called "constraint inference", as opposed to a "value inference" process that is performed during simulation, [2].

Subsection 3.2), and set up a new M_S.

Figure 9 illustrates the search process graphically: The circuit S defines $\mathcal{M}_S$, from which a subset is selected ($= M_S$), simulated ($\Rightarrow B_S$, the behavior of S under M_S), and compared to $\mathcal{B}_{Hyd}$, which stands for the universal behavior laws of hydraulics.

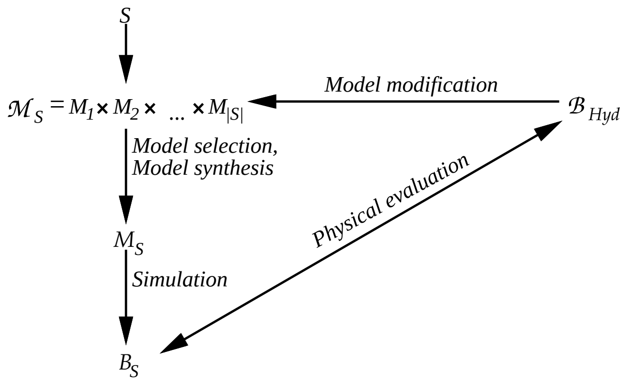

Figure 9: *Exploring the synthesis search space* $\mathcal{M}_S$

The search comes to an end if either a coherent global behavior model is found or if no further choice point exists. Note that different components constrain the model synthesis process in a different manner. Hence, the order by which undetermined components are processed plays a crucial role.

3.2 Realization

The outlined approach has been realized within the drawing and simulation environment Fluid*SIM*.[5] Fluid*SIM* operationalizes the philosophy of a graphic problem specification. While the circuit diagram of a system is drawn, a knowledge base containing the topological and the physical connections is created. I. e., the model synthesis process as well as complex physical dependencies are made transparent: Nearly all information obligatory for the analysis is derived from the technical drawing. Figure 10 depicts a snapshot when working in Fluid*SIM*'s simulation mode.

During simulation also the user is allowed to trigger events by operating components like switches or valves. The related models are updated immediately in the background, thus providing the feeling of interacting with a running system.

A strength of Fluid*SIM* comes with its powerful concepts to efficiently solve the model synthesis problem. Aside from the deployment of domain knowledge, there are two mechanism to *automatically* create synthesis restrictions that cut down the search space: (a) a topological circuit analysis, (b) a dependency recording which is based on the operational mode assumptions.

E. g., in the circuit of Figure 6 all variable assignments containing the subsets

[5] Fluid*SIM* originates from a DFG research project where the institute MSRT, University of Duisburg and the working group Knowledge-based Systems, University of Paderborn were involved. The Fluid*SIM* system in its actual form has been developed by D. Curatolo, M. Hoffmann, and B. Stein.

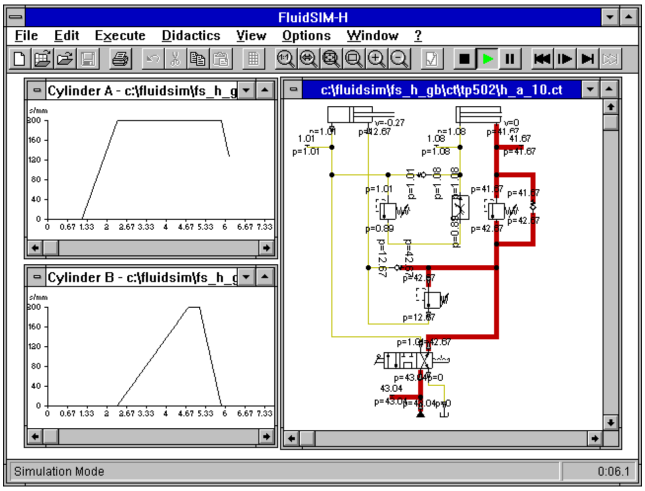

Figure 10: *The snapshot shows a simulation run and the cylinders' related distance/time diagrams.*

$\{m_{rv\,n_a}, m_{cyl\,n_b}\}$ or $\{m_{rv\,n_b}, m_{cyl\,n_a}\}$, $n \in \{2,3\}$, can be discarded at the outset: Cylinder 2 (3) can only extend if `rv_2` (`rv_3`) is open. Variable assignments that define *physically* contradictory mode combinations are called *nogoods*. [6]

Constraint processing in Fluid*SIM* encloses standard numerical procedures as well as inference methods for value propagation or rule processing. In this text we will not engage into constraint processing details; additional information may be found in [1,15] and [14].

4 Discussion

We have presented case studies of two different applications in the hydraulic domain. The associated CSPs could be solved by a uniform hybrid constraint processing procedure, which shows its usability and practicability.

The presented constraint processing procedure is coined by an engineering view to technical domains: A portion of the constraints is viewed as model synthesizing constraints, the rest is interpreted as a set of behavior constraints.

To estimate the benefits of this approach consider the alternative: Each of the two problems is modeled into a CSP on a flat and undifferentiated set of algebraic constraints, which then is solved by a generic procedure. However, this often results in a cumbersome and relatively inefficient solution mechanism and is bound up with the following disadvantages.

- Algebraic methods poorly handle constraint alternatives $\Rightarrow$ dynamically retracting or adding of model fragments as a consequence of Step 2b of the hybrid constraint processing is impossible.
- *Case Study 1.*

(1) No interactive configuration $\Rightarrow$ no direct user feedback, user is confronted with skillfully selecting a correct subset of input parameters.

---

[6] There still exist further nogoods, such as $\{m_{rv\,n_b}, m_{rv3_a}\}$, $n \in \{2,3\}$: `rv_2` or `rv_3` can be open only if `rv_1` is open as well.

(2) No local cause-effect chains ⇒ no explanation facility.
(3) Systems of nonlinear equations and inequations must be handled ⇒ ambiguous solutions, poor convergence, additional conditions for physical quantities cannot be easily incorporated.

- *Case Study 2.*

(1) The physical knowledge w. r. t. a backtracking is not exploited.
(2) Topological knowledge for the generation of nogood gets lost.

References

[1] D. Curatolo. *Wissensbasierte Methoden zur effizienten Simulation fluidtechnischer Systeme*. Dissertation, University of Paderborn, Department of Mathematics and Computer Science, 1996.

[2] E. Davis. Constraint Propagation with Interval Labels. *Artificial Intelligence*, 32:281–331, 1987.

[3] J. de Kleer and J. Brown. A Qualitative Physics Based on Confluences. *Artificial Intelligence*, 24:7–83, 1984.

[4] J. Doyle. A Truth Maintenance System. *Artificial Intelligence*, 12:231–272, 1979.

[5] P. V. H. F. Benhamou, D. Mc Allester. CLP(Intervals) Revisited. Technical Report CS-94-18, Université Aix-Marseille, Apr. 1994.

[6] B. Falkenhainer and K. Forbus. Compositional Modeling: Finding the Right Model for the Job. *Artificial Intelligence*, 51:95–143, 1991.

[7] K. Forbus and J. de Kleer. *Building Problem Solvers*. MIT Press, Cambridge, MA, 1993.

[8] E. Hyvönen. Constraint reasoning based on interval arithmetic: the tolerance propagation approach. *Artificial Intelligence*, 58:71–112, 1992.

[9] Y. Iwasaki and A. Y. Levy. Automated Model Selection for Simulation. Knowledge Systems Laboratory, Stanford University (QS93), 1993.

[10] P. P. Nayak. *Automated Model Selection*. Dissertation, Stanford University, 1992.

[11] P. P. Nayak. Causal Approximations. *Artificial Intelligence*, 70:277–334, 1994.

[12] F. B. Pascal Van Hentenryck, Laurent Michel. Newton—Constraint programming over nonlinear constraints. *Science of Computer Programming, special issue on concurrent constraint programming, Elsevier*, 30:83–118, 1998.

[13] M. Stefik. *Introduction to Knowledge Systems*. Morgan Kaufmann Publishers, Inc., 1995. pp. 670–771.

[14] B. Stein. *Functional Models in Configuration Systems*. Dissertation, University of Paderborn, Institute of Computer Science, 1995.

[15] B. Stein, D. Curatolo, and M. Hoffmann. Enriching Engineering Education in Fluidics. *Proc. WMC 98, Western Multiconference featuring ICSEE '98, International Conference on Simulation and Multimedia in Engineering Education, San Diego, California*, Jan. 1998.

[16] E. Tsang. *Foundations of Constraint Satisfaction*. Computation in Cognitive Science. Academic Press, 1993.