Transactions on Information and Communications Technologies vol 8, © 1995 WIT Press, www.witpress.com, ISSN 1743-3517

A semi-quantitative physics compiler

A. Farquhar[a], G. Brajnik[b]

[a]*Knowledge Systems Laboratory, Stanford University, Palo Alto, CA, USA*

[b]*Dipartimento di Matematica e Informatica, Università di Udine, Udine, Italy*

1 Introduction

Consider the problem of water supply control. A lake has a dam with floodgates that can be opened or closed to regulate the water flow through power generating turbines, the water level (stage) of the lake, and the downstream flow. The goal of a controller is to provide adequate reservoir capacity for power generation, consumption, industrial use, and recreation, as well as downstream flow. In exceptional circumstances, the controller must also work to minimize or avoid flooding both above and below the dam. This task is both difficult and vitally important to the residents of surrounding areas. The work of controllers could be substantially eased by sound automatic modeling and simulation tools.

There are several forms of incomplete information that appear in this domain. The precise shape and capacity of lakes or reservoirs is rarely known; the outflow from opening a dam's floodgates is only crudely measured; empirical data on the level/flow–rate curve for rivers becomes less and less accurate when flood conditions approach; few quantities are measured (e.g. flow rates of minor tributaries are not measured at all); the amount of runoff to be expected from a given rainfall depends on difficult to measure surface characteristics such as saturation; the amount of rainfall that actually falls on a lake and surrounding areas is difficult to predict and is imprecisely measured. Nonetheless, both mathematical analysis and observations do provide rough bounds on the quantities involved. Often, rough accurate bounds suffice to select appropriate actions.

This domain is challenging for existing approaches to modeling and simulation. Pure qualitative reasoning techniques [Forbus, 1984; Kuipers, 1986] do not exploit the partial information available and consequently provide insufficiently strong predictions. Traditional numeric methods require much more precise information than is available, forcing modelers to make assumptions which may invalidate results and which may be difficult to evaluate.

This paper describes SQPC (semi–quantitative physics compiler), an implemented approach to modeling and simulation that uses semi–quantitative knowledge. SQPC performs *self–monitoring simulations* of incompletely known, dynamic, continuous systems. It monitors the simulation in order to detect violations of model assumptions; when this happens it modifies the model and resumes the simulation. SQPC is the first compositional modeling system to employ semi–quantitative representation and simulation.

2 Semi–quantitative simulation

SQPC is built on top of the QSIM qualitative simulator [Kuipers, 1994] and takes advantage of its capability to represent and deal with bounds on variable values and functional bounds (envelopes) on otherwise unspecified monotonic functions [Berleant and Kuipers, 1988]. The semi–quantitative simulator augments behavior with the numeric bounds and it also uses the semi–quantitative information to rule out qualitatively possible behaviors. The semi–quantitative technique propagates the bounds throughout each time–point state, and then uses the mean–value theorem to constrain the values across time.

3 Semi–Quantitative Physics Compiler

SQPC is an extension of QPC [Farquhar, 1994], whose modeling language builds on Qualitative Process Theory [Forbus, 1984]. The input to SQPC is a *domain theory* and *scenario* specified in the modeling language. A domain theory consists of a set of quantified definitions, called *model fragments*, each of which describes some aspect of the domain, such as physical laws (e.g. mass conservation), processes (e.g. liquid flows), devices (e.g. pumps), and objects (e.g. containers). Each definition applies whenever there exists a set of participants for whom the stated conditions are satisfied. The specific system or situation being modeled is partially described by the scenario definition, which lists a set of objects that are of interest, some of the initial conditions, and relations that hold throughout the scenario.

SQPC employs a hybrid architecture in which the model building portion is separated from the simulator. The domain theory and scenario induce a set of logical axioms. SQPC uses this database of logical axioms to infer the set of model fragment instances that apply during the time covered by the database (called the *active* model fragments). Inferences performed by SQPC include those concerning structural relationships between objects declared in the scenario, and those aiming at computing the transitive closure of order relationships between quantities. A database with a complete set of model fragment instances defines an initial value problem which is

Transactions on Information and Communications Technologies vol 8, © 1995 WIT Press, www.witpress.com, ISSN 1743-3517

given to the simulator in terms of equations and initial conditions. If any of the predicted behaviors cross the boundary conditions the process is repeated: a new database is constructed to describe the system as it crosses the boundaries of the current model, another complete set of active model fragments is determined, and another simulation takes place.

The output of SQPC is a directed rooted graph, whose nodes are either databases or qualitative states. The root of the graph is the initial database, and a possible edge in the graph may: (i) link a database to a refined database (obtained by adding more facts, either derived through inference rules or assumed by SQPC when ambiguous situations are to be solved); (ii) link a complete database to a state (which is one of the possible initial states for the only model derivable from the database); (iii) link a state to a successor state (this link is computed by QSIM); and (iv) link a state to a database (the last state of a behavior which crossed the operating region to the database which describes the situation just after the transition occurred). Each path from the root to a leaf describes one possible temporal evolution of the system being modeled; each model in such paths identifies a distinct operating region of the system. SQPC is proven to construct all possible sequences of initial value problems that are entailed by the domain theory and scenario; thanks to QSIM correctness, it produces also all possible trajectories.

3.1 Semi–quantitative modeling

Numeric values. SQPC represents numeric and qualitative magnitudes in a single framework. Both denote specific real numbers, which might be known only with uncertainty. Numeric magnitudes constrain such a number to lie within a numeric range. Two aspects complicate reasoning on numeric magnitudes. First, two comparable magnitudes constrained by the same range are, in general, not equal (*i.e.*, $Range(m) = [a\ b]$ and $Range(n) = [a\ b]$ do not entail that $m = n$ unless $a = b$). Secondly, range constraints on magnitudes may change during the analysis. This may happen as an effect of the semi–quantitative simulation performed by QSIM. A model might entail $Range(m) = [a\ b]$, while a subsequent model in the behavior graph computed by SQPC might entail $Range(m) = [a'\ b']$ where $[a'\ b'] \subseteq [a\ b]$. That is, as the analysis proceeds, SQPC may tighten the bounds on the numeric range of a magnitude.

Dimensional information. Variables and (symbolic or numeric) magnitudes are partitioned into dimensions. SQPC defines the seven International System dimensions as well as a *null–dimension*, which is provided to represent "pure number" quantities such as the efficiency of a turbine. Explicit representation of dimensions enables SQPC to: (i) perform dimensional analysis and verify that equations and order relations are well formed.

Dimensional errors are common when writing equations and can be easily detected; and (ii) constrain inference about order relations. It is senseless to compare quantities that do not have the same dimension, and a reasoning mechanism not exploiting any dimensional information can produce incorrect inferences such as $x < 5 \wedge 10 < V \vdash x < V$ where a position (x) is being compared to a volume (V).

Bounding envelopes. An *envelope schema* is defined by a form similar to model fragments. It states a set of conditions under which a specific form of monotonic function over a tuple of variables is bounded by a functional envelope. The envelope is specified by a pair of functions. Instantiated envelope schemas are used to enrich a model with suitable envelopes. Since instantiation is automatically performed, envelopes are installed in models as needed, provided an appropriate monotonic constraint has already been included in the model. If the model includes a constraint for which not envelope is applicable, SQPC is still able to infer accurate results (though with degraded precision). However, differently than constraints, envelopes cannot be the result of a composition process (of some sort of "numeric influences"): they must be explicitly provided.

SQPC needs to determine which envelopes to include in the SQDE for each model. This is non–trivial because there are several ways to describe a monotonic relationship among a set of quantities. Because each envelope that can be included is likely to strengthen the predictions, it is important to include all of the applicable ones. For instance, suppose that the model contains the constraint $(M\ (+\ -)\ X\ Y\ Z)$ but there is an envelope defined for the constraint $(M\ (+\ +)\ Y\ Z\ X)$. These two constraints are analytically equivalent, but the second constraint and its envelope enable ranges for X to be computed given ranges for Y and Z. SQPC adds any constraint and envelope into the SQDE that is a permutation of a constraint in the SQDE. Notice that SQPC includes constraints in models after resolving influences (*i.e.*, after assuming a closed world and having determined the complete set of influencing and influenced variables). This strategy makes it possible for the designer of the domain model and scenario to specify the envelopes, or envelope schemas, on the basis of the available data, independently from how influences will get resolved.

Tabular functions. Tabular functions provide an important practical extension to the modeling language. A large portion of empirically collected knowledge about time–varying systems is represented and summarized in tabular form. The SQPC language permits numeric functions (used to specify envelopes) to be defined by data in a multi–dimensional table. SQPC assumes that these tables are coarse descriptions of the continuous reasonable functions that satisfy monotonic constraints. Currently SQPC provides two mechanisms for interpolating tabular data: *stepwise* functions,

Transactions on Information and Communications Technologies vol 8, © 1995 WIT Press, www.witpress.com, ISSN 1743-3517

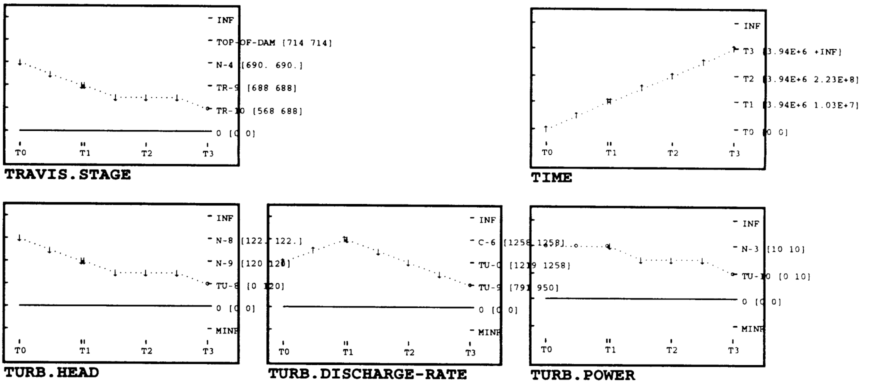

Figure 1: Behavior plot for several variables in the scenario. The power output by the turbine after T1 is below the desired level.

providing piecewise constant upper and lower bounds, or *piecewise linear* functions, providing tighter, but possibly less accurate, interpolations. In this way it is possible to define two or more envelope schemas from the same underlying tabular data. One envelope schema might use a linear interpolation method in a region where this approximation is known to introduce no significant error; in other regions a safer, but less precise, envelope schema using the more conservative interpolation method based on stepwise bounding functions, might be used. Of course the set of interpolation methods being used for computing tabular functions is open ended. The current version of SQPC provides the two mentioned above.

4 An Example

We demonstrate SQPC on a problem from the domain of water supply control. We consider a portion of the system of lakes and rivers to be found in the scenic hill country surrounding Austin, Texas. The Colorado river flows into Lake Travis; the Mansfield Dam on Lake Travis produces hydroelectric power, controls the level of the lake, and the flow into the downstream leg of the Colorado.

The problem is to evaluate a "what if" scenario. We are given an initial level for Lake Travis (a typical value between 690.2 and 690.3 feet) and a rough projected inflow from the Colorado river (between 791 and 950 cfs). The task is to determine what happens to the lake level and evaluate how long the hydroelectric plant can deliver power at the requested rate of 10 Mw.

All of the semi–quantitative information in this domain theory is specified in the form of tables. The tables reflect both observations and engineering estimates about the relationships between important variables.

Solving this problem is made slightly more complex because of the behavior of the turbines. The turbines are controlled by a servo–mechanism designed to generate the desired amount of power regardless of the hydraulic pressure, which is determined by the *head* at the turbine. This is possible as long as there is sufficient *head*. When the *head* drops below the minimum threshold for a given power output, then less power is released. The domain theory captures this accurately. The domain theory also includes model fragments for conservation laws (e.g. of mass and energy), basic hydraulic principles (e.g. flow is proportional to head), and so on.

Figure 1 shows the SQPC output for this scenario. Under the specified conditions, the desired power level can be maintained until time `T1`, at least 45 days ($3.94 * 10^6$ seconds) after the start time. After `T1`, there will be insufficient hydraulic pressure to provide the full power output, the discharge rate from the turbine will decrease until it reaches equilibrium with the inflow at a rate between 791 and 950 cfs, and the lake level will stabilize between 568' and 688'. Notice that at `T1` the lake system is entering a new operating region because the turbine is no longer servo–controlled (*i.e.*, the model fragment `NORMAL-TURBINE-MF` is no longer active).

These predictions are strong enough to be useful to a system controller, even though the problem statement is very imprecise: the flow rate was very coarse; there are no semi–quantitative bounds for the relationship between *power* and *head* in the low–head situation after `T1`; the table relating *stage* and *capacity* becomes very coarse below 600'.

More precise information in the domain theory or scenario will result in more precise predictions. This is the strength of the semi–quantitative inference methods. We illustrate this by first strengthening the initial conditions of the scenario and then by strengthening the domain theory. If the upper bound on the inflow rate is reduced from 950 cfs to 800 cfs, then the upper bound on `T1`, the time that power generation drops below the desired rate, is reduced to 76 days, a 58% improvement. The domain theory can be strengthened by tightening the envelopes by using a linear interpolation for the stage–capacity curve instead of a step function. This tightens the range for `T1` to 50–58 days, an improvement of 89% from the original. Increased precision in the input or model leads to increased precision in the output.

5 Related work

In recent years, several research efforts have worked towards the development of self–explanatory simulators that construct numerical simulations and use a qualitative representation to help explain the results. Unlike SQPC, they do not use semi–quantitative information. Their predictions are either precise numeric ones, or purely qualitative.

Transactions on Information and Communications Technologies vol 8, © 1995 WIT Press, www.witpress.com, ISSN 1743-3517

SIMGEN [Forbus and Falkenhainer, 1990] computes a total envisionment of the scenario and then, for each envisionment state it builds a numerical simulator, monitors the simulation and, at the end of the analysis, interprets numerical results in terms of the envisionment graph. SIMGEN requires precise and complete numerical equations, initial and boundary conditions for the simulation. SIMGEN must be capable of building a numerical model for *each* envisionment state touched during the simulation; to this end it must be supplied with a library of numeric procedures for *every* possible combination of influences. SIMGEN is incapable of performing a simulation when a qualitative relation is quantitatively underspecified or when precise knowledge unavailable for any initial conditions.

DME (the Device Modeling Environment) [Iwasaki and Low, 1991] is an incremental compositional modeling system capable of generating self–explanatory simulations. DME can work in two exclusive modes: qualitative or numeric. In the former case DME constructs qualitative states, and uses QSIM to generate successors; in the latter case, DME builds numerical models for simulation. In both modes, crossing an operating region triggers remodeling. DME is highly interactive and provides sophisticated explanation capabilities [Gruber and Gautier, 1993]. DME requires precise numerical equations, initial and boundary conditions. Therefore, DME does not integrate qualitative and quantitative information in prediction.

Pika [Amador *et al.*, 1993] builds a numerical model for each operating region of the system as soon as this is needed. Pika monitors the numerical simulation and, at the end of the analysis, is capable of engaging in a simple question/answering dialogue. Pika requires precise equations, complete initial conditions (unlike the other systems), and complete specification of boundary conditions (in particular inequalities are not allowed). Compared to SQPC Pika performs limited inferences: no structural inferences are possible (this limits the expressive power of the modeling language) and influences are limited to indirect and algebraic ones: no provision is made for handling more general monotonic influences.

6 Conclusion

We have presented SQPC, the first system to unify compositional modeling techniques with semi–quantitative simulation. This is crucial for automatically building models of systems whose dynamics cross several operating regions. SQPC automatically constructs semi–quantitative models and produces useful predictions with imprecise knowledge. We argued that semi–quantitative knowledge is crucial to many applied engineering domains like the one chosen for demonstrating SQPC, water supply control. Coupled with compositional modeling, the semi–quantitative techniques have the promise of achieving one of the major goals of qualitative reasoning: to make strong predictions about behavior, given the strongest model available.

References

[Amador *et al.*, 1993] F. Amador, A. Finkelstein, and D. Weld. Real–time self–explanatory simulation. In *Proc. of the Eleventh National Conference on Artificial Intelligence.* AAAI Press/MIT Press, 1993.

[Berleant and Kuipers, 1988] D. Berleant and B. Kuipers. Using incomplete quantitative knowledge in qualitative reasoning. In *Proc. of the Sixth National Conference on Artificial Intelligence*, pages 324–329, 1988.

[Farquhar, 1994] A. Farquhar. A qualitative physics compiler. In *Proc. of the 12th National Conference on Artificial Intelligence*, pages 1168–1174. AAAI Press / The MIT Press, 1994.

[Forbus and Falkenhainer, 1990] K. Forbus and B. Falkenhainer. Self–explanatory simulations: an integration of qualitative and quantitative knowledge. In *Proc. of the Eight National Conference on Artificial Intelligence*, pages 380–387. AAAI Press / The MIT Press, 1990.

[Forbus, 1984] K. Forbus. Qualitative process theory. *Artificial Intelligence*, 24:85–168, 1984.

[Gruber and Gautier, 1993] T. Gruber and P. Gautier. Machine–generated explanations of engineering models: a compositional modeling approach. In *Proc. International Joint Conference on Artificial Intelligence*, pages 1502–1508, Chambery, F., 1993.

[Iwasaki and Low, 1991] Y. Iwasaki and C. M. Low. Model generation and simulation of device behavior with continuous and discrete changes. Technical Report KSL 91–69, Knowledge Systems Laboratory — Stanford University, November 1991.

[Kuipers, 1986] Benjamin Kuipers. Qualitative simulation. *Artificial Intelligence*, 29:289–338, 1986.

[Kuipers, 1994] B. Kuipers. *Qualitative Reasoning: modeling and simulation with incomplete knowledge.* MIT Press, Cambridge, Massachusetts, 1994.