An Interactive Environment for Scientific Model Construction

Javier Nicolás Sánchez and Pat Langley
Computational Learning Laboratory
Center for the Study of Language and Information
Stanford University, Stanford CA 94305 USA
{jsanchez, langley}@apres.stanford.edu

ABSTRACT

Most AI research on scientific model construction aims to automate this process using discovery techniques. In contrast, we describe an interactive environment for model construction that lets the user construct, edit, and visualize scientific models, use them to make predictions, and call on discovery methods to revise them in ways that better fit the available data. The environment relies on a new formalism that embeds mathematical equations, which are familiar to many scientists, within distinct processes, which can encode background knowledge used to constrain model revision. We report initial studies on ecosystem modeling that suggest this environment is more effective than earlier approaches and more transparent to users. In closing, we discuss related work on modeling environments and model revision, then suggest directions for future research.

Categories and Subject Descriptors

I.6 [**Simulation and Modeling**]: Model Development

General Terms

Design, Human Factors, Languages

1. Introduction

Over the past decade, computational approaches to scientific discovery have progressed to the stage where they have contributed to finding new knowledge in a variety of scientific disciplines [8]. However, nearly all research on computational discovery has aimed at generating knowledge automatically, whereas scientists generally prefer computational tools to assist in data analysis and model development, as Schneiderman [17] has proposed. We need more work on software environments that support interaction between domain scientists and discovery algorithms, thus drawing on the strengths of each one. We envision a computational framework that lets a scientist formulate a model, generate its predictions, detect anomalies, and alter the model in response. The scientist would devise the initial model and guide high-level decisions about iterative refinement, with the computer handling predictions, fine-grained search, and other steps that are easily automated.

Another recent trend has been the incorporation of domain knowledge in ways that constrain the discovery process, thus directing their search for models and ensuring their output is more communicable to domain scientists. However, the notations used to encode this background knowledge are not as transparent to domain experts as one would like. For instance, Todorovski and Džeroski's [20] LAGRAMGE states background knowledge as context-free grammars, which few scientists can interpret easily, and Bannai et al.'s [1] 'views' have a similar drawback. A different problem occurs with Saito et al.'s [16] approach, which requires a different error function for each revision task. We need a framework that casts domain constraints in communicable terms and that requires no tuning for different problems.

In this paper we introduce PROMETHEUS, an environment that addresses both of these challenges. As we will describe shortly, the system incorporates a formalism for specifying models and background knowledge in terms of quantitative processes, which play a role in many scientific accounts. The environment also includes tools for constructing, visualizing, and editing such process models, for utilizing them in predictive simulation, and for constrained revision in response to observations, thus supporting the iterative refinement of scientific models. We demonstrate these capabilities in the context of revising a partial model of the Earth's ecosystem, and we compare our results with earlier ones for this problem. We conclude by discussing the related work on simulation and discovery, along with directions for future research in this important area.

K-CAP'03, October 23–25, 2003, Sanibel Island, Florida, USA.

2. A Process Modeling Language

As we have noted, existing computational approaches to scientific discovery generate knowledge in terms of numeric equations or other formalisms familiar to scientists. However, these notations often leave implicit an aspect that plays an important role in many scientific fields – the notion of a *process*. To incorporate this idea into our discovery framework, we designed a language that supports quantitative equations but that organizes them into explicit processes.[1]

Models in our formalism consist of variables and processes that relate them. Each process expresses causal relations between its input and output variables through one or more differential equations (if a process involves change over time) or algebraic equations (if it involves instantaneous effects). Furthermore, a process may include conditions, stated as threshold tests on variables, which describe the preconditions that must be satisfied for the process to be active.

Consequently, a process model consists of a set of processes that link observable variables with each other, possibly through unobserved theoretical terms. Many examples from the literature suggest that scientists often think in this manner. For instance, processes like fluid flow, boiling, and motion occur in many physical models, whereas processes like growth, decay, and predation play an important role in biology. Further evidence that processes are prominent in science comes from Cartier et al. [2], whose study of the structure of scientific models indicated that they typically relate observable or theoretical terms through the processes in which they participate. Moreover, they emphasized that models can be used to explain and predict phenomena, something that our framework also supports.

The notion of process models has also arisen in AI research on qualitative physics. Our formalism borrows many ideas from Forbus' [5] Qualitative Process Theory, which also organized causal relations into processes with activation conditions. However, his framework cast these relations as qualitative influences, whereas ours takes the form of numeric equations, each of which associates a dependent variable with a mathematical formula that involves zero or more influencing variables. Thus, causal dependencies in the model are made explicit through the equations associated with each process. Since the use of an intermediate term inside a process would hide a causal relation, we expressly disallow such 'local variables'.

The PROMETHEUS environment lets a user display the causal structure of a process model, as Figure 1 illustrates. Note that variables are shown as ovals and processes as rectangular boxes, while arrows depict causal influences. An arrow from a process to a variable indicates the latter's value is affected by that process. Similarly, an arrow from a variable to a process shows the process uses that variable as input. The PROMETHEUS environment also lets the user inspect and edit individual processes, as the box in the lower corner reveals.

[1]We focus here on continuous processes from scientific domains, rather than discrete processes, such as those found in manufacturing and business.

Table 1: A quantitative process model of mass and temperature change in an ice-water system.

```
model WaterPhaseChange;
variables    temp, heat, ice_mass, water_mass;
observables temp, heat, ice_mass, water_mass;
process ice_warming;
  conditions ice_mass > 0, temp < 0;
  equations  d[temp,t] = heat/(0.0021 * ice_mass);
process ice_melting;
  conditions ice_mass > 0, temp == 0;
  equations  d[ice_mass,t] = −(18 * heat)/6.02,
             d[water_mass,t] = (18 * heat)/6.02;
process water_warming;
  conditions ice_mass == 0, water_mass > 0,
             temp >= 0, temp < 100;
  equations  d[temp,t] = heat/(0.0042 * water_mass);
```

We will return to this example in Section 5, but we can best illustrate the use of our modeling language with a simpler instance. Table 1 shows a simple process model for the changes in an ice-water system as a function of the heat put into it. The model includes three processes, one (ice_warming) active when the ice mass is nonzero and the system temperature is less than zero degrees Celsius, another (ice_melting) when the ice mass is nonzero and the temperature is zero degrees, and a third (water_warming) when all the ice has melted and the temperature is between zero and 100 degrees. The first and third processes influence only the temperature, whereas the second process affects only the masses of ice and water. The notational convention for equations is similar to that used in mathematical software, with `d[temp,t,1]` denoting the first-order derivative of `temp` with respect to `t` (time).

The previous comments should convince the reader that our representation makes close contact with the conceptualization of many scientific models. Furthermore, the formalism is not only useful for representing scientific knowledge but also serves as the declarative representation for PROMETHEUS' discovery methods. Thus, the system's discoveries are cast in the same process modeling language, making its results communicable to scientists. A final advantage of the framework is that the environment can utilize its models, whether hand crafted or system generated, to carry out simulations, and thus provides integrated support for scientific modeling.

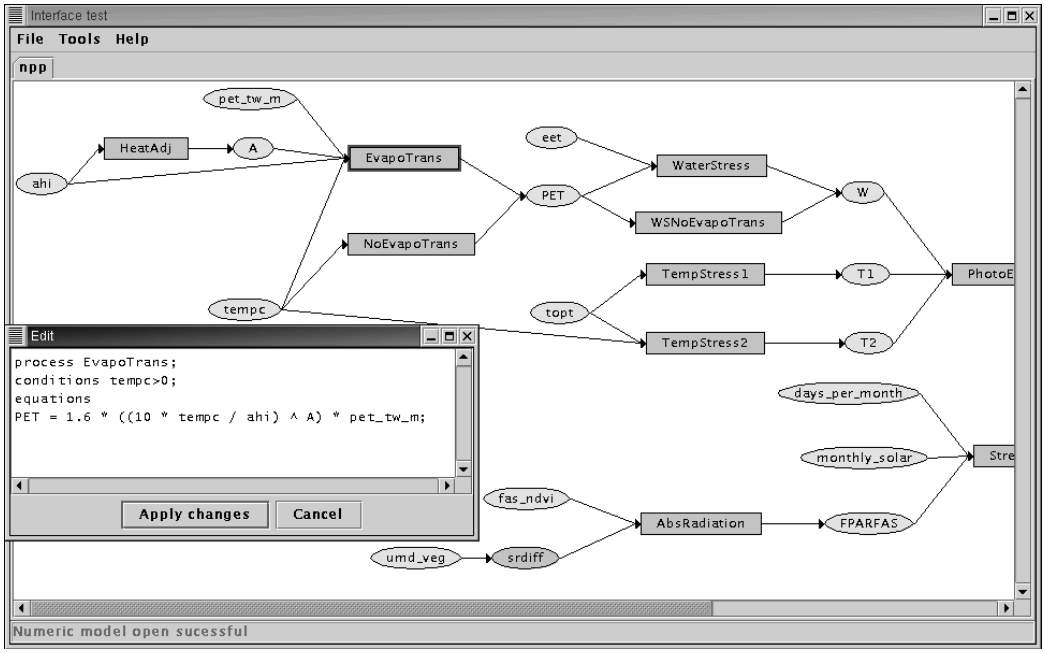

Figure 1: Screen shot of a process model displayed graphically in PROMETHEUS.

3. Simulation and Prediction in PROMETHEUS

Scientists devise models not only to understand phenomena but also to use them for prediction and simulation. Moreover, when refining a model, the ability to analyze its behavior often gives valuable insight into ways in which the model might be improved. For this reason, PROMETHEUS includes a module for numerical simulation and analysis that works directly on models expressed in its modeling language. This provides the support necessary to evaluate the models' fit to observations and to generate predictions for new situations.

Simulation in PROMETHEUS involves translating a model from its encoding in the modeling language into a C program. As in some earlier systems (e.g., [6], [7], [18]), the translation occurs automatically, without user intervention, thus freeing the scientist from error-prone implementation details. Moreover, a scientist's models can change frequently during their construction and revision. Thus, the ability to alter the high-level specification of the model, without worrying about implementation details, should aid productivity considerably.

The environment supports models that involve only algebraic equations and ones that include both algebraic and differential equations. Since these have different requirements, PROMETHEUS generates tailored code for each type. For static models, the system generates a program that relies on the standard mathematical library from the C language, since this is sufficient for making predictions. For dynamic models with differential equations, the environment instead generates code that invokes CVODE [3], a well-known package for solving systems of ordinary differential equations.[2] Simulation of static models proceeds by reading values for the input variables from a stream of data and computing values for dependent variables for each set of inputs. Simulation of dynamic models requires solution of an initial value problem, so the system reads only the starting values of input variables from the data file, then predicts a trajectory over time for each variable.

Finally, PROMETHEUS includes visualization tools to inspect the results of simulation. One facility lets the user plot the model's predictions for a given variable against the observed values to assist in detecting anomalies. Another visualization module modulates a display like Figure 1 to highlight which processes are active at each time step in the simulation, thus letting the user track the model's qualitative behavior as conditions change.

4. Interactive Revision of Scientific Models

The construction of scientific models is not a one-step affair. A model's developer may revise it many times after its initial creation, as new data or knowledge become available. For this reason, PROMETHEUS includes a module to support the interactive revision of an existing model. The key idea is that the user provides constraints on possible revisions that define a restricted search space, which the system explores to find an improved model with a better fit to the data.

[2]Because conditions in processes change the equations in the model dynamically, our simulator adapts which equations are solved according to which processes are active.

The model revision module relies on the notion of *generic processes* that encode relevant background knowledge. These differ from specific processes (described previously) in that they do not commit to particular variables or parameter values, although they do specify variable types and ranges of allowed values. Each generic process denotes an entire family of specific processes that refer to different variables and constants, and thus encodes a declarative bias on the space of equations.

We claim that generic processes and variable types are powerful tools for stating domain content that one can easily extract from available knowledge. For example, they have proved useful in our development of population dynamics models that incorporate processes for the growth, death, and predator-prey relations between species. Generic processes are appropriate because constants like growth rate, death rate, and predation rate depend on the species and ecosystem being studied, but the general concepts recur in many models.

The user can request that PROMETHEUS revise a model by specifying a data file to examine and indicating which processes to consider altering and which generic processes to consider as replacements. After entering revision mode, one clicks on processes in the graphical display to denote they can be revised. For each such model component, the system presents a menu of generic processes from which the user selects possible replacements. Each candidate has the same number of variables as the original process, and its variable types are the same as, or higher in the variable taxonomy than, the initial ones.

After the user has specified the processes that PROMETHEUS should consider during model revision and the generic processes it should try in their place, the environment calls on a subroutine that carries out exhaustive search through the space of model structures defined by these constraints.[3] For each model structure, the system carries out a gradient descent search for the parameters that provide the best fit to the data.

Once PROMETHEUS has determined parameters for each model structure, it returns the candidate with the lowest mean squared error as the revised model. The user can accept this revision or retain the original model. In either case, he can then invoke the revision module again in an effort to obtain further improvements, continuing until he is satisfied with the resulting model.

5. Initial Results on Model Revision

To demonstrate PROMETHEUS' abilities, we used the environment to revise a portion of CASA, an ecosystem model developed by Potter and Klooster [14] at NASA Ames Research Center. CASA aims to explain the production and absorption of atmospheric trace gases, changes in major vegetation types, and the net production of carbon by plants. For our revision task, we chose a portion of the model that focuses on carbon production, denoted by *NPPc*, at different locations on the globe. Computational revision of this submodel has

[3]The version we have described invokes the LAGRAMGE [19] discovery system to carry out this search, but we are also exploring other methods for implementing model revision.

Table 2: Initial model for carbon production.

```
model npp;
variables NPPc,E,IPAR,T1,T2,W,topt,tempc,eet,PET,
          ahi,pet_tw_m,A,FPARFAS,monthly_solar,
          days_per_month,fas_ndvi,umd_veg;
process CarbonProd;
  conditions E * IPAR > 0;
  equations NPPc = E * IPAR;
process NoCarbonProd;
  conditions E * IPAR <= 0;
  equations NPPc = 0;
process PhotoEfficiency;
  equations E = 0.56 * T1 * T2 * W;
process TempStress1;
  equations T1 = 0.8 + 0.02 * topt - 0.0005 * topt^2;
process TempStress2;
  equations
     T2 = 1.18/(1 + (e^(0.2 * (topt-tempc - 10))))
           * (1 + (e^(0.3 * (tempc - topt - 10))));
process WaterStress;
  conditions PET != 0;
  equations W = 0.5 + 0.5 * (eet / PET);
process NoEvapoTrans;
  conditions PET == 0;
  equations W = 0.5;
process EvapoTrans;
  conditions tempc > 0;
  equations
     PET = 1.6 * (10 * tempc / ahi)^A * pet_tw_m;
process NoEvapoTrans;
  conditions tempc <= 0;
  equations PET = 0;
process HeatAdj;
  equations A = (6.75 * 10^-7 * ahi^3) -
     (0.0000771 * ahi^2) + (0.01792 * ahi) + 0.49239;
process StressComb;
  equations IPAR = FPARFAS * monthly_solar *
                   (days_per_month * 0.0864) * 0.5;
process AbsRadiation;
  equations FPARFAS = (((1 + fas_ndvi / 1000) /
                (1 - fas_ndvi / 1000)) - 1.08) / srdiff;
intrinsic srdiff;
  variable umd_veg;
  mapping A → 3.06, B → 4.35, C → 4.35, D → 4.05,
          E → 5.09, F → 3.06, G → 4.05, H → 4.05,
          I → 4.05, J → 5.09, K → 4.05;
```

Table 3: Initial and revised values for the intrinsic property srdiff.

Vegetation type	A	B	C	D	E	F	G	H	I	J	K
original values	3.06	4.35	4.35	4.05	5.09	3.06	4.05	4.05	4.05	5.09	4.05
PROMETHEUS revision	2.52	4.48	0.01	2.84	3.31	3.27	2.31	0.01	1.71	2.84	1.04
Saito et al. revision	2.57	4.77	2.20	3.99	3.70	3.46	2.34	0.34	2.72	3.46	1.60

been the subject of a previous publication [16], which makes it a good problem for evaluating our system.

Table 2 presents the submodel for predicting NPPc cast in the process modeling formalism. In this case, there are no differential equations, but a number of processes include conditions, such as the two responsible for predicting the potential evapotranspiration *PET*. Note also the final entry, which specifies a mapping from vegetation type (one of 11 letters) onto different values of *srdiff*, which relates to the absorption of solar radiation.

Using this encoding for the NPPc model, we revisited the revisions reported by Saito et al. [16]. These had included changes to parameters in the equations for T2 and PET, the structure of the E equation, and the intrinsic values for the variable srdiff, which maps nominal values to numeric ones. To support these revisions, we needed three generic processes to specify the variable types and parameter constraints for PhotoEfficiency, TempStress2, and EvapoTrans, along with statements about the range of numeric values allowed for srdiff.

In addition, we provided another generic process that could replace the one used to compute the photosynthetic efficiency E. This took the form

```
generic process PhotoEfficiencyGen;
 variables S1{stress},S2{stress},S3{stress},F{efficiency};
 equations F = [0,0.56,100] * S1^[0,1,5]
                            * S2^[0,1,5] * S3^[0,1,5];
```

which has a functional form that is somewhat more general than the original process, allowing the variables S1, S2, and S3, which must have type stress, to take on powers between 0 and 5, with 1 as their default.

We invoked PROMETHEUS' revision module with this background knowledge, requesting that it consider altering the processes TempStress2, EvapoTrans, and PhotoEfficiency, shown in Table 2, along with the intrinsic values for srdiff. We provided the system with the same 303 training observations as used in the earlier study, which contained measurements for some variables in the model that had been collected from ground stations. In separate runs, we asked the system to consider revising these components both individually and together.

PROMETHEUS' individual revisions of TempStress2 and EvapoTrans produced equations with the parameters

```
T2 = 28.01 / [1 + e^(-0.0058 * (topt − tempc + 160.1))]
       * [1 + e^(-0.03 * (tempc − topt − 91.5))]
PET = 1.44 * (9.40 * tempc / ahi)^A * pet_tw_m
```

These revisions are not very enlightening, although they reduced the cross-validated error by three and two percent, respectively. Saito et al. [16] reported similar results, and these runs partly reproduced their findings.

However, when asked to revise the PhotoEfficiency by considering not only parameter changes but also different functional forms, the system selected the latter, giving a six percent error reduction with the equation

```
E = 0.53 * T1^0.0 * T2^0.055 * W^0.0
```

for the prediction of photosynthetic efficiency. These results are more interesting, as they suggest that the stress variables T1 and W have effectively no influence on E. The Earth scientists were intrigued with this outcome, since it suggests that simplifying their model can actually improve its fit.

Saito et al. also applied their method to revise the 11 values of the intrinsic property srdiff. For PROMETHEUS to revise intrinsic values, it need only search for the best-fitting parameters that fall within the specified range for each nominal value. In this case, revision yielded the results shown in Table 3, which reduced error by nine percent and which are generally similar to those found in the earlier study. The main differences occur on the vegetation types (C and H) for which few data were available, so that we cannot treat either result as especially reliable.

More important, our interactive environment lets the user revise multiple processes simultaneously, something that Saito et al.'s system did not support. When we asked PROMETHEUS to consider altering the processes TempStress2, EvapoTrans, and PhotoEfficiency, as well as the 11 srdiff values, it produced a revised model that contained changes only to EvapoTrans and srdiff, leaving the other two processes unmodified. However, these alterations produced an even greater reduction in error, in this case over 12 percent.

Table 4: Comparison of revision results for Saito et al.'s method and PROMETHEUS.

Equation	Saito et al.		PROMETHEUS	
	RMSE	err. red.	RMSE	err. red.
$T2$	457.8	0.02	453.7	0.03
PET	464.3	0.01	460.9	0.02
E	443.3	0.05	439.8	0.06
srdiff	432.4	0.08	424.6	0.09
All revisions	N/A	N/A	409.8	0.12

Table 4 summarizes the key statistics for the results obtained with PROMETHEUS and Saito et al.'s method. For each revision, we report the root mean squared error (RMSE) on the dependent variable NPPc for the revised model and the error reduction over the original amount of 467.9. As the table shows, the two approaches improved the model's fit to data by about the same amounts, even though they used different methods for parameter optimization. It also highlights the fact that PROMETHEUS can revise several aspects of a model, which in this case produced substantial improvements that were not possible with the earlier approach.

Another clear advantage is that PROMETHEUS does not require the user to specify a new error function for each revision, which would make the task intractable for all but experts in parameter fitting. Instead, the environment requires only that it have access to a library of generic processes that it should consider during its search for improved models. In this example, we introduced the necessary background knowledge just before calling the revision module, but in normal use it would already be stored in a library of generic processes that has been developed by the scientific community.

In summary, we have shown that PROMETHEUS can carry out a number of distinct revisions to its quantitative process models, from changing the values of parameters and intrinsic values to replacing one functional form with another. The framework lets the user constrain the search for improved models by specifying which processes to alter and how it might change them, and provides the ability to revise a number of model components in a single run. Thus, PROMETHEUS offers a significant advance over previous revision methods, which were less flexible and much more difficult to use.

6. Discussion

Our approach to scientific modeling incorporates ideas from two previously disconnected literatures – simulation environments and computational scientific discovery. With respect to the former, PROMETHEUS has many similarities to modeling frameworks like STELLA [15] and MATLAB [12]. These also let the user specify quantitative models in terms of mathematical equations, edit these models, and invoke a simulator to generate predictions. Moreover, they provide a graphical interface that lets the user display and inspect the logical structure of these mathematical models.

Our approach also shares many features with Keller's [7] SIGMA, another graphical environment that takes an interactive approach to model building, visualization, and analysis, though it also provides extensive checks to ensure model consistency and handle unit conversions. Stickel et al. [18] report still another approach to synthesizing simulation programs; their AMPHION system lets the user specify a model using a graphical interface, then draws on software libraries to compile the model into executable code.

However, PROMETHEUS moves beyond these earlier modeling environments by requiring the user to organize equations into *processes*. This idea plays a central role in many scientific disciplines, but previous quantitative simulation languages have not supported it. The main exception comes from Forbus and Falkenhainer [6], who developed a self-explanatory simulator that creates numerical simulation code from a combination of qualitative and quantitative structures. Their SIMGEN system exploits qualitative mathematics to provide causal explanations and produces numerical output that can be embedded in training simulators and other software. PROMETHEUS goes farther to support computational revision of models in response to data, constrained by domain knowledge in the form of generic processes and input from the user. MATLAB includes facilities for fitting a model's parameters to data, but it cannot alter the basic structure of a model.

PROMETHEUS also incorporates many ideas from earlier work on computational scientific discovery. In particular, it adopts the metaphor of heuristic search through a space of candidate hypotheses or models guided by their ability to fit the data. Our approach differs from other quantitative discovery work (e.g., [10], [19], [22]) by focusing on process models, rather than on independent sets of equations, and by emphasizing model revision rather than generation, though it borrows ideas on this front from earlier efforts, especially Saito et al. [16].

The environment also draws upon the notion of using explicit domain knowledge to constrain the search for models. For example, Easley and Bradley [4] utilize generalized equations as background knowledge in their approach to identifying differential equation models of dynamic systems. Similarly, Todorovski and Džeroski's [19] LAGRAMGE casts background knowledge in terms of context-free grammars that specify the space of equations to consider. PROMETHEUS incorporates a similar mechanism, but states its domain knowledge in terms of generic processes rather than these other formalisms.

Todorovski and Džeroski [20] also report an approach to revising quantitative models, in particular the NPPc model considered earlier. However, their framework requires the user to specify the space of candidate models as a grammar, which will be neither familiar or especially communicable to most scientists. In contrast, PROMETHEUS encodes constraints on models in terms of generic processes that make contact with both the conceptual content and the formalisms familiar to many domain experts.

But the main difference from earlier discovery research concerns the interactive nature of our environment. Previous work on computational scientific discovery has focused almost exclusively on automated methods, while PROMETHEUS aims explicitly to support scientists rather than to replace them. This philosophy is consistent with a general trend in artificial intelligence research toward advisory systems, but it means we have had to address issues of human-computer interaction that some algorithm-oriented researchers will find uninteresting. Nevertheless, such issues must receive serious attention if we hope to develop computational assistants that practicing scientists will use on a regular basis.

We should note that PROMETHEUS is not quite the first discovery environment designed to accept user input. For example, Valdés-Pérez' [21] MECHEM lets users influence its search for chemical reaction pathways by setting switches that specify constraints, expressed in terms familiar to chemists, that the inferred pathways must satisfy. Another example is Mitchell et al.'s [13] DAVICCAND, which encourages users to direct its search for quantitative relations in metallurgical data, provides control points where they can influence its choices, and presents its results in terms of graphical displays and functional forms that are familiar to metallurgists. Both systems have been used to produce novel results that have appeared in the refereed scientific literature.

However, the research closest to our own comes from Mahidadia and Compton [11], who report an integrated environment for the revision of causal models. Their JUSTAID system starts with a user-provided model and recommends changes to this model that would improve its fit to experimental results, checking with the user before implementing them. Our efforts share many goals, including a concern with encoding knowledge in forms familiar to domain scientists, a focus on model revision, and an emphasis on interactive discovery rather than automated methods. The primary difference is that their work concentrates on qualitative modeling, whereas ours centers on quantitative modeling.

One topic for further research involves extending the modeling language to incorporate the notion of subsystems that map to physical entities. This should let users manage models of increased complexity and provide additional constraints on the revision process, provided we augment background knowledge to include generic systems for the domain. We should also incorporate another form of domain knowledge – a taxonomy of quantitative processes – that would let users specify initial models in more abstract terms while still constraining their revision.

Another important extension would enable the revision module to add new processes to the current model, remove existing ones, and even incorporate new variables and associated processes. To this end, we can adapt methods that we have described elsewhere [9] for inducing models by composing generic processes. Moreover, we should develop a more interactive version of the environment that produces a number of candidate revisions which it then presents to the user for evaluation.

Finally, we should test PROMETHEUS on models and data from additional scientific domains in order to provide evidence of its generality, and we should study its use by scientists in controlled settings, which should help us evaluate its suitability as a practical modeling tool. Such studies, and the improvements that result from them, should take us closer to an interactive environment for modeling and discovery that effectively aids scientific research.

7. Concluding Remarks

In this paper, we described an interactive environment, PROMETHEUS, for the construction and revision of scientific models. The system relies centrally on a a new formalism for encoding both models and domain knowledge that is communicable to scientists and useful for constraining search through the model space, thus filling a gap in earlier discovery systems.

We applied our framework to the revision of an existing ecosystem model that involved the same changes attempted in earlier work. The resulting models had approximately the same accuracy as the previous revisions, but they required must less effort, since PROMETHEUS needed no custom modification of the equations, creation of error functions, or reformulation of the model. Moreover, the environment let us consider combinations of changes that were not possible in the earlier scheme, which suggests that it offers a more powerful approach to model revision.

In summary, our research on PROMETHEUS contributes to the the capture of scientific knowledge along a number of fronts. These include a new formalism for representing quantitative models and interactive tools that let the user visualize, simulate, analyze, and revise models encoded in this notation. Together, they provide a unified computational framework that should aid scientists in their modeling efforts.

Acknowledgements

This research was supported in part by NTT Communication Science Laboratories, Nippon Telegraph and Telephone Corporation. We thank Christopher Potter for making available both the CASA model and the relevant data set, Kazumi Saito for sharing his results on revising this model, and Ljupčo Todorovski for his help integrating LaGramge into Prometheus and his feedback on the modeling language.

References

[1] Bannai, H. Y. T., Maruyama, O., Miyano, S.: VML: A view modeling language for computational knowledge discovery. In: Proceedings of the Fourth International Conference on Discovery Science. Springer (2001) 30–44.

[2] Cartier, J., Rudolph, J., Stewart, J.: The nature and structure of scientific models. The National Center for Improving Student Learning and Achievement in Mathematics and Science, Working Paper (2001).

[3] Cohen, S., Hindmarsh, A.: CVODE, A stiff/nonstiff ODE solver in C. SciCADE95: Scientific Computing and Differential Equations, Stanford University (1995).

[4] Easley, M., Bradley, E.: Generalized physical networks for automated model building. In: Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence. Morgan Kaufmann (1999) 1047–1053.

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

[6] Forbus, K. D., Falkenhainer, B.: Self-explanatory simulations: An integration of qualitative and quantitative knowledge. In: Proceedings of the Eighth National Conference on Artificial Intelligence. AAAI Press (1990) 380–387.

[7] Keller, R. M.: An intelligent visual programming environment for scientific modeling. Science Information Systems Newsletter, **35**, Information Systems Branch, Office of Space Science, NASA (1995).

[8] Langley, P.: The computational support of scientific discovery. International Journal of Human-Computer Studies **53** (2000) 393–410.

[9] Langley, P., Sánchez, J., Todorovski, L., Džeroski, S.: Inducing process models from continuous data. In: Proceedings of the Eighteenth Conference on Machine Learning. Morgan Kaufmann (2002) 347–354

[10] Langley, P., Simon, H. A., Bradshaw, G. L., Żytkow, J. M.: *Scientific discovery: Computational explorations of the creative processes*. Cambridge, MA: MIT Press (1987).

[11] Mahidadia, A., Compton, P.: Assisting model-discovery in neuroendocrinology. In: Proceedings of the Fourth International Conference on Discovery Science. Springer (2001) 214–227.

[12] The MathWorks, Inc.: Simulink *user's guide: Dynamic system simulation for MATLAB*. Natick, MA (1997).

[13] Mitchell, F., Sleeman, D., Duffy, J. A., Ingram, M. D., Young, R. W.: Optical basicity of metallurgical slags: A new computer-based system for data visualisation and analysis. Ironmaking and Steelmaking **24** 306–320 (1997).

[14] Potter, C. S., Klooster, S. A.: Inter annual variability in soil trace gas (CO_2, N_2O,NO) fluxes and analysis of controllers on regional to global scales. Global Biochemical Cycles **12** (1998) 621–635.

[15] Richmond, B., Peterson, S., Vescuso, P.: An academic user's guide to STELLA. Lyme, NH: High Performance Systems (1987).

[16] Saito, K., Langley, P., Grenager, T., Potter, C., Torregrosa, A., Klooster, S.A.: Computational revision of quantitative scientific models. In: Proceedings of the Fourth International Conference on Discovery Science. Springer (2001) 336–349.

[17] Schneiderman, B.: Creating creativity: User interfaces for supporting innovation. In: ACM Transactions of Computer-Human Interaction **7** (2000) 114–138.

[18] Stickel, M., Waldinger, R., Lowry, M., Pressburger T., Underwood, I.: Deductive composition of astronomical software from subroutine libraries. In: Proceedings of the Twelfth Conference on Automated Deduction Springer-Verlag (1994) 341–355.

[19] Todorovski, L., Džeroski, S.: Declarative bias in equation discovery. In: Proceedings of the Fourteenth International Conference on Machine Learning. Morgan Kaufmann (1997) 376–384.

[20] Todorovski, L., Džeroski, S.: Theory revision in equation discovery. In: Proceedings of the Fourth International Conference on Discovery Science. Springer (2001) 389–400.

[21] Valdés-Pérez, R. E.: Machine discovery in chemistry: New results. Artificial Intelligence **74** (1995) 191–201.

[22] Washio, T., Motoda, H. (1998). Discovering admissible simultaneous equations of large scale systems. In: Proceedings of the Fifteenth National Conference on Artificial Intelligence. AAAI Press (1998) 189–196.