

SOPT: ONTOLOGY FOR SIMULATION OPTIMIZATION FOR SCIENTIFIC EXPERIMENTS

Jun Han
John A. Miller

Department of Computer Science
University of Georgia
Athens, GA, 30602, USA

Gregory A Silver

College of Business
Anderson University
Anderson, SC, 29621, USA

ABSTRACT

Simulation optimization is attracting increasing research interest from the modeling and simulation community. Although there is much research on how to apply various simulation optimization techniques to solve numerous practical and research problems, researchers find that existing optimization routines are difficult to extend or integrate and often require one to develop their own optimization methods because the existing ones are problem-specific and not designed for reuse. In order to facilitate reuse of the available optimization routines and better capture the essence of different simulation optimization techniques, an ontology for simulation optimization (SoPT) is devised. SoPT includes concepts from both conventional optimization/mathematical programming and simulation optimization. Represented in ontological form, optimization routines can also be transformed into actual executable application code (*e.g.*, targeting JSIM or ScalaTion). As illustrative examples, SoPT is being applied to real scientific computational problems.

1 INTRODUCTION

In our prior work on ontology driven simulation (Silver et al. 2010), we provided a tool for assisting a simulation model developer to create a model from ontological information contained in domain ontologies. Focusing on modeling, the work left out two important steps, namely design of simulation experiments and simulation optimization.

When designing simulation experiments it is important to examine the list of inputs and the list of outputs. For simplicity, in this paper, we assume the inputs to a model are deterministic and the outputs from a model are stochastic. Since the outputs are stochastic, it is necessary to replicate the runs of a simulation model to obtain statistical estimates (*e.g.*, means, variances and confidence intervals). When designing simulation experiments, one typically wishes to explore a subset of the input parameter space, generating corresponding output results. Typically, enough replications should be run to obtain adequate confidence intervals.

We divide input parameters into three categories: regular input parameters, control parameters and model parameters.

Control parameters influence the behavior of a model and are in some sense controllable. An example for certain metabolic pathway simulations is concentrations of enzymes that catalyze reactions in pathways. Experiments may be conducted where enzyme levels are up-regulated, down-regulated or even reduced to zero by knocking out the responsible gene. Metabolic pathway models can simulate these possibilities so that the enzyme concentrations may be adjusted to optimize the production of certain bio-molecules.

Model parameters are intrinsically part of a model, but may be unknown and hence, need to be estimated from empirical data or calibrated by comparing output results, with empirical data. Reaction rates in metabolic pathways are good examples of model parameters. Many of these are not accurately known in the literature and are difficult to measure directly. They may, however, be estimated by performing

an optimization that adjusts the reaction rates with the objective of minimizing the differences (*e.g.*, least squares) between empirical pathway time series data and those generated by a metabolic pathway model.

As discussed, model parameters may be optimized to improve the accuracy of a simulation model, while control parameters may be optimized to improve the outcome of the model (*e.g.*, a better yield of bio-molecules). Simulation optimization can be accomplished in a couple of ways. One is to simply explore the input parameter space over a sufficiently detailed grid to produce a response surface. Then, for example, a quadratic surface can be fit to the output points produced by the simulation. An optimization method can now be used to find a global optimum. This approach is called Response Surface Methodology (RSM) and is currently not considered in our work. Rather, our work focuses on simulation optimization techniques that loosely couple a simulator with an optimizer, having them work together in an iterative fashion. The optimizer steers the exploration of the simulator’s input space.

In this paper, we consider how optimization can be used in simulation, as well as, the types of optimization problems that are encountered. We illustrate the approach by considering two related simulation optimization problems from the bioinformatics domain: Mass Spectrometry and Metabolic Pathways. Based on these case studies, we develop an ontology to complement the Discrete-event Modeling Ontology (Miller et al. 2004) that can serve as a knowledge repository for simulation optimization.

The rest of this paper is organized as follows: Section 2 introduces two scenarios in bioinformatics research that need modeling, simulation and optimization. Section 3 categorizes conventional optimization problems into groups based on characterization of optimization components. Section 4 reviews the current work in simulation optimization and ontology development in Modeling and Simulation (M&S). Section 5 outlines the design of the Simulation oPTimization (SoPT). Section 6 concludes the paper and discusses possible future work.

Notational conventions: (1) boldface letters in lower case for vectors, (2) capitalization for matrices and random variables.

2 MODELING SCENARIOS

2.1 Scenario 1: Mass Spectrometry Model

Mass spectrometers (MS) are powerful tools widely utilized in the identification and quantitative analysis of chemical samples due to their capacity for high-throughput, high precision and high sensitivity (Zaia 2010). An MS experiment involves the ionization and gas-phase analysis of molecules to generate data in the form of a mass spectrum, which is generally represented as an array of ion intensities versus mass-to-charge ratios (m/z), providing information regarding the molecular mass of the molecules in the sample.

Given the elemental composition of a molecule (*e.g.*, $C_xH_yO_zN_uNa$), the mass spectrum for this molecule is simulated in two major steps: (1) calculate the isotopic distribution of each isotopic configuration for this molecule via Multinomial distributions and (2) calculate the intensity via Normal distributions of all possible isotopic configurations, as shown in Figure 1.

For example, light water ($^1H_2^{16}O$), semi-heavy water ($^1H^2H^{16}O$), heavy water ($^2H_2^{16}O$) and heavy-oxygen water ($^1H_2^{18}O$) are an incomplete list of six possible stable isotopic configurations for water whose elemental composition is H_2O . Now, suppose there are K atoms of a given element in a molecule. Each such atom may be one of l different isotopes, and therefore, the element has a vector $\mathbf{k} = (k_1, \dots, k_l)$ indicating the number of atoms for each of the l isotopes where $k_1 + \dots + k_l = K$. Given a molecule with K atoms of a certain element, the probability that the molecule has k_j such atoms having isotope j ($j = 1..l$), is determined by the *probability mass function* (pmf) of the Multinomial distribution (Hellerstein and Neese 1999).

$$p(\mathbf{k}) = p(k_1, \dots, k_l) = \frac{K!}{\prod_{j=1}^l k_j!} \times \prod_{j=1}^l p_j^{k_j} \quad (1)$$

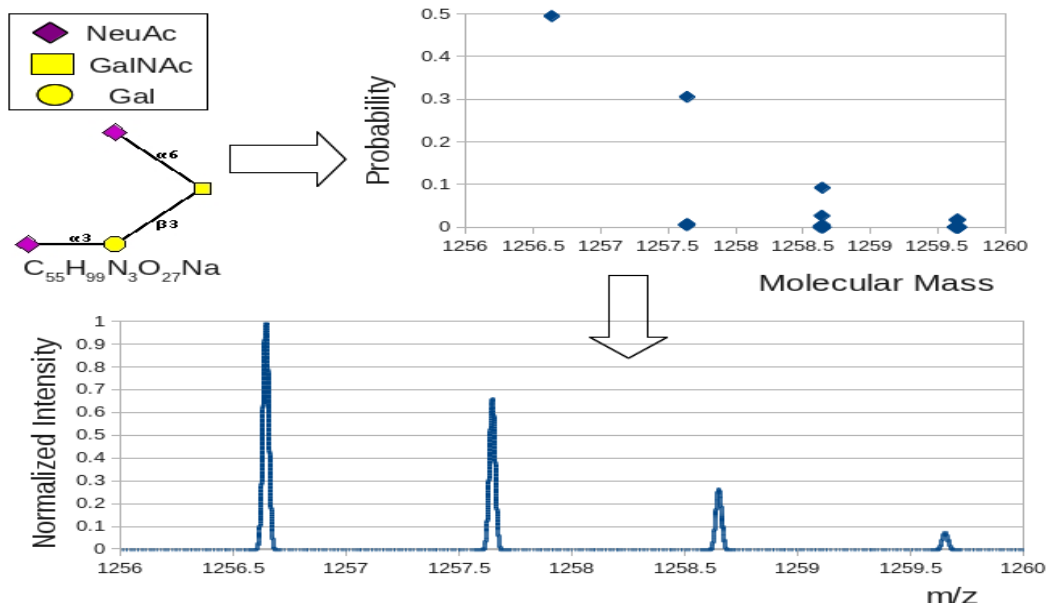


Figure 1: Mass Spectrometry Model: elemental composition \rightarrow isotopic distribution \rightarrow simulated mass spectrum. Cartoon representation comes from CFG glycan structure database (available at <http://www.functionalglycomics.org/glycomics/molecule/jsp/carbohydrate/carbMoleculeHome.jsp>).

where p_j is the relative abundance of each of the element's isotopes (*e.g.*, for element Hydrogen (H), relative abundances for ^1H and ^2H are 0.99985 and 0.00015, respectively). As a molecule is made up of E elements, to determine the probability for all of the elements in the molecule, the product of E such pmfs is required. The exact molecular mass for this isotopic configuration ($\mathbf{k}_1, \dots, \mathbf{k}_E$) can also be calculated.

$$p(\mathbf{k}_1, \dots, \mathbf{k}_E) = \prod_{i=1}^E p(\mathbf{k}_i) \quad (2)$$

$$m(\mathbf{k}_1, \dots, \mathbf{k}_E) = \sum_{i=1}^E \mathbf{k}_i \cdot \mathbf{m}_i \quad (3)$$

where i is the index of the i^{th} element and \mathbf{m}_i is the mass vector for the l isotopes of element i (*e.g.*, for element Hydrogen (H), $\mathbf{m} = (1.007825, 2.014101)$, representing the isotopic masses for ^1H and ^2H , respectively). These probabilities and masses for various isotopic configurations correspond to intensity and m/z readings obtained from the mass spectrometer, where z is the charge of the ion. Due to the fact that masses are clustered and inherent properties of mass spectrometers, intensity peaks (see Figure 1) will typically have Gaussian shapes (Normal Distribution).

Each isotopic configuration has its effect on the whole simulated mass spectrum based on Normal distributions; therefore, the actual intensity of each $\zeta = m/z$ in the mass spectrum is the sum of the intensities of all the possible isotopic configurations at m/z . Given experimental data represented as an array of $[m/z, \text{intensity}]$, a peak width (w) and charge state (z), the intensity value contributed by the j^{th} isotopic configuration is simulated by the pmf for the Normal distribution (Han et al. 2011).

$$f_j(\zeta) = \frac{1}{\sigma\sqrt{2\pi}} \times e^{-\frac{(\zeta - \zeta_j)^2}{2\sigma^2}} \quad (4)$$

where $\sigma = 0.4247 \times w$ (Inczédy et al. 1998). Weighting this by the pmf from Equation 3, we obtain:

$$f_G(\zeta) = \sum_j f_j(\zeta) p_j \quad (5)$$

In summary, the input parameters to a mass spectroscopy model are listed as follows:

- Input Parameters: (1) glycan structure, (2) experimental spectrum
- Control Parameters: (1) spectrum calibration, (2) low level noise filtering. Both parameters are controlled by the mass spectrometer, which changes the intensity response of experimental mass spectrum.
- Model Parameters: (1) peak width, (2) charge state, and (3) relative abundance vector for each structure.

All the inputs are real numbers. Constraints for the input parameters are in linear form with lower and upper bounds. The goal is to minimize the errors between the two and maximize the Pearson correlation coefficient $R = \sqrt{R^2}$, where the coefficient of determination R^2 is computed by comparing the normalized intensity of the simulated spectrum with the experimental one. Multinomial distributions and Normal distributions are used in isotopic distribution calculation and mass spectrum simulation, respectively. Therefore, the overall objective function is in nonlinear form. To sum up, the problem of mass spectrometry model calibration belongs to linear constrained, nonlinear programming. Both gradient-based and heuristic methods can be applied to solve this particular problem.

2.2 Scenario 2: Metabolic Pathway Model

A metabolic pathway consists of a series of chemical reactions, each of which modifies the structure of a biomolecule as it passes through the reaction. Figure 2a represents a portion of an O-linked glycan pathway. It is made up of four substrates (molecules with which enzymes react), five enzymes, three steps and five biosynthesis reactions. An enzyme catalyzes a reaction which transforms a substrate into a product which becomes a substrate for the next reaction.

Our objective is to build a simulation model, which accurately represents both the qualitative (structural/visual) and quantitative (mathematical) aspects of a metabolic pathway. Biochemical pathway models have traditionally been developed using a system of differential equations (Heinrich and Schuster 1998), but because differential equations model only the quantitative aspects, we have chosen to use Hybrid Functional Petri Nets (HFPN) (Matsuno et al. 2003), as shown in Figure 2, to model pathways. (See (Silver et al. 2009) for a description of using HFPNs to model biochemical pathways.)

Several of the model's input parameters, such as concentrations of substrates and enzymes, for the pathway can be estimated from experimental data, but accurate reaction rates are more difficult to obtain. In order to estimate accurate reaction rates, we will use simulation optimization. The rate of a reaction is modeled using Michaelis-Menten kinetics which makes use of the kinetics constants K_{cat} and K_m . These constants are the target of our simulation optimization. (See (Nimmagadda 2008) for a detailed description of Michaelis-Menten kinetics in pathway simulation.) The biomolecule and enzyme concentrations for the pathway can be estimated from experimental data, while the simulated results will be produced using the HFPN model depicted in Figure 2. The output of the simulation will consist of the biomolecule concentrations produced by each of the models three reactions at five time points over a 36 hour period. Our optimization will use experimental results along with simulated results in order to develop an accurate estimate of the K_{cat} and K_m constants, which control the pathway's reaction rates. The experimental and simulated results will be used as input for a least squares model that will quantify the difference between the two. The output of the least squares along with the experimental results will be used by an optimizer to adjust the K_{cat} and K_m constants, thus modifying the reaction rates, used in the next set of simulations.

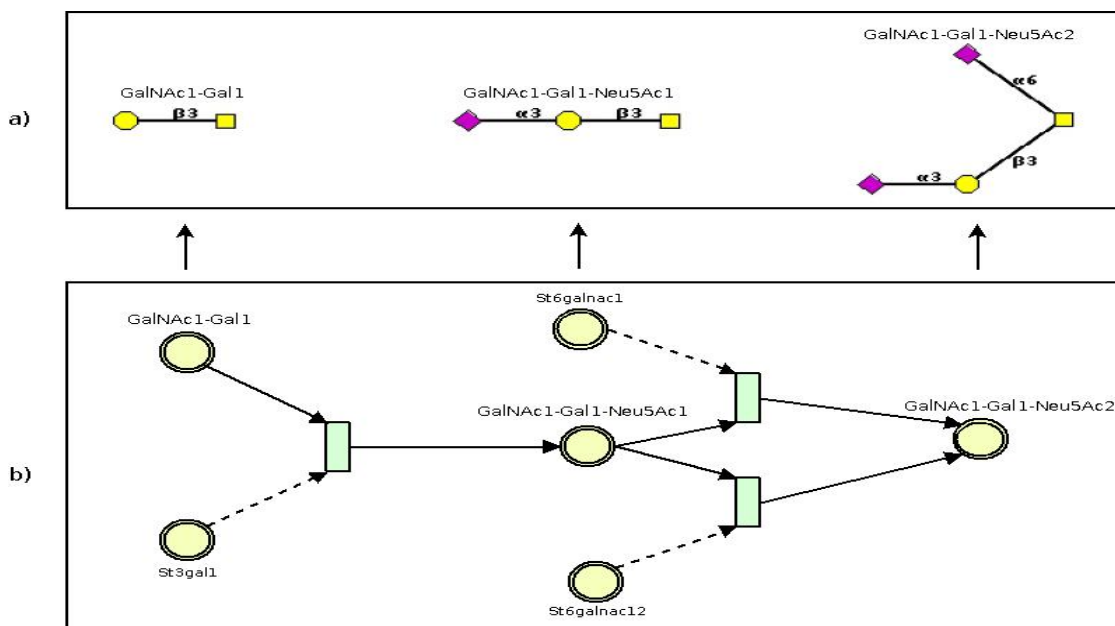


Figure 2: Metabolic Pathway Model. a) Substrates in each step of the O-Glycan metabolic pathway, b) HFPN model of O-Glycan metabolic pathway. Cartoon representation comes from the CFG glycan database.

3 CLASSIFICATION OF OPTIMIZATION PROBLEMS

To facilitate component reuse, we are primarily interested in systems where simulators and optimizers are loosely-coupled. In addition, there may be a third component, a cost analyzer. In general, the simulator will take a set of input/parameter vectors $\{\mathbf{x}\}$, and produce a set of output/response vectors, $\{\mathbf{Y}\} = R(\{\mathbf{x}\})$ where \mathbf{x} is thought of as deterministic and \mathbf{Y} is often stochastic. The cost analyzer will take the response vectors \mathbf{Y} along with a quality vector \mathbf{q} , and provide a set of triples $\{(\mathbf{x}, \mathbf{Y}, Z)\}$ using a cost function to compute $Z = c(\mathbf{Y}, \mathbf{q})$. These triples are then fed into the optimizer, which will compare these values with those it has stored from previous iterations. After applying a stopping rule, the optimizer will either report a solution or ask for more response data from the simulator, as shown in Figure 3. Denoting the composition of the cost and response functions as $F = c \circ R$, optimizations may be formulated in 6.

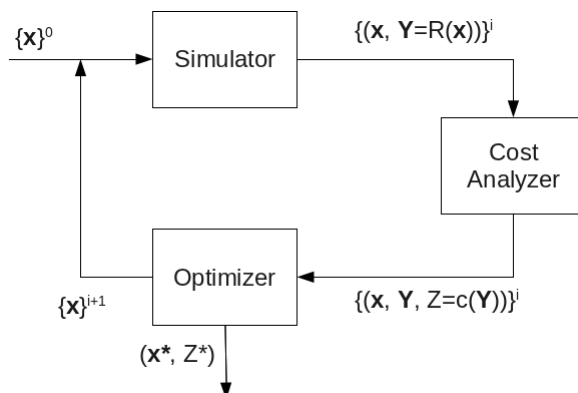


Figure 3: Loosely-coupled software architecture for Simulation Optimization.

$$\begin{aligned}
\min \quad & E[F(\mathbf{x})] \\
\text{s.t.} \quad & V[F(\mathbf{x})] \leq t \\
& g(\mathbf{x}) \geq 0 \\
& \mathbf{x} \in D
\end{aligned} \tag{6}$$

where $F(\mathbf{x})$ is a stochastic objective function, g is a constraint function, t is the threshold for variance and the domain D (often a vector space) can be subsets of \mathbb{R}^n , \mathbb{Z}^n or \mathbb{B}^n , for real, integer or binary vector spaces, respectively. Due to the stochastic nature of the objective function, one must focus on characteristics of $F(\mathbf{x})$, such as moments or quantiles. Typically, the mean (E) of $F(\mathbf{x})$ is used. Other characteristics can be used in constraints (*e.g.*, limiting the variance (V) to be within a threshold).

Now, if R is deterministic, the simulator may be thought of as a function evaluator (using either analytic or numerical evaluation). Then F becomes deterministic, so that $E[F(\mathbf{x})] = F(\mathbf{x})$ and $V[F(\mathbf{x})] = 0$, leading to conventional optimization. For efficiency, the same evaluations can be produced by a meta-model, which typically provides more rapid evaluation and produces deterministic responses ($\{\mathbf{Y}\}$ is deterministic). Replacing F with f , this may be formulated as follows:

$$\begin{aligned}
\min \quad & f(\mathbf{x}) \\
\text{s.t.} \quad & g(\mathbf{x}) \geq 0 \\
& \mathbf{x} \in D
\end{aligned} \tag{7}$$

In conventional optimization, the problem may be defined as finding the minimum value of an objective function $f(\mathbf{x})$ over domain D subject to the constraint function $g(\mathbf{x})$ being non-negative. We use minimization for simplicity (maximization could be used as well). Classification of the type of optimization problem is primarily determined by the characteristics of f , g and D . The objective function f can be linear, quadratic or non-linear. The constraint function may be non-existent (unconstrained), linear, quadratic or non-linear. Note, in this paper, non-linear more accurately means what is not accounted for by the special cases of linear (and quadratic where applicable). D is typically a subset of \mathbb{R}^n , \mathbb{Z}^n or \mathbb{B}^n , but may also be mixed.

A secondary classification is based on solution quality, in terms of finding a global minimum versus a local minimum as well as whether an exact, approximate or heuristic solution is acceptable. Exact involves convergence on a solution, while approximate typically guarantees a solution within a relative error bound and heuristics offer no guaranteed error bounds.

Techniques and applications of engineering optimization are addressed in (Rao 2009) from a broad perspective. Convex optimization and numerical optimization techniques are discussed in (Boyd and Vandenberghe 2004, Nocedal and Wright 2006).

A portion of our classification of optimization methods is listed in Table 1. To save space, the column of Constraint, including unconstrained, linear, quadratic and nonlinear constraints is omitted; the distinctions are saved for the discussion below. Typically generalized solvers (*e.g.*, for nonlinear programming) can be applied to solve the more specific form (*e.g.*, linear programming), although less efficiently. People try to look for a one-size-fits-all solution for various practical optimization problems. However, since the generalized solvers (*e.g.*, for nonlinear programming) may get trapped in local optima and take much longer time than some more specific solvers, there is a need to have several solvers (*e.g.*, the Simplex method for linear programming). Therefore, in order to determine the most suitable and efficient solvers, the first and crucial step in optimization is always to determine the classification of a particular optimization problem.

Linear constrained, linear programming (LP) (8), integer linear programming (ILP) (9) and linear constrained, quadratic programming (QP) (10) are three types of such problems. Because they are simple to express in canonical forms, they are taken as examples to illustrate the basic components involved in optimization, as listed in Table 2.

Table 1: Classification of conventional optimization problems.

Objective Function	Restriction	Problem	Methods
Linear	Real	Linear Programming	Simplex Method, Interior Point Methods (Khachiyan and Karmarkar)
	Integer	Integer Linear Programming	Branch and Bound
	Real/Integer	Mixed Integer Programming	Branch and Bound
	Binary	Binary Integer Programming	Balas Additive Algorithm
Quadratic	Real	Quadratic Programming	Calculus and Quadratic Simplex method
	Integer	Quadratic Integer Programming	Special Branch and Bound
	Real/Integer	Mixed Integer Quadratic Programming	Generalized Benders Decomposition (GBD)
	Binary	Binary Quadratic Programming	Heuristic Methods
Nonlinear	Real	Nonlinear Programming	Gradient-based Methods
	Integer	Nonlinear Integer Programming	Branch and Bound, Outer-Approximation
	Real/Integer	Mixed Integer Nonlinear Programming	Branch and Bound, Outer-Approximation
	Binary	Nonlinear Integer Programming	Heuristic Methods

Table 2: Examples of optimization components.

LP	ILP	QP
$\begin{aligned} \min \quad & \mathbf{c}^T \mathbf{x} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} \geq \mathbf{b} \\ & \mathbf{x} \in \mathbb{R}^{+n} \end{aligned} \quad (8)$	$\begin{aligned} \min \quad & \mathbf{c}^T \mathbf{x} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} \geq \mathbf{b} \\ & \mathbf{x} \in \mathbb{Z}^{+n} \end{aligned} \quad (9)$	$\begin{aligned} \min \quad & \frac{1}{2} \mathbf{x}^T \mathbf{Q}\mathbf{x} + \mathbf{c}^T \mathbf{x} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} \geq \mathbf{b} \\ & \mathbf{x} \in \mathbb{R}^n \end{aligned} \quad (10)$

where \mathbf{x} , \mathbf{b} and \mathbf{c} are the vector of input/decision variables, the constant vector, and the cost coefficient vector, respectively, A and Q are the coefficient matrices for constraint and objective functions, respectively. If Q is zero, (10) is reduced to (8).

4 RELATED WORK

The need of combining simulation and optimization is summarized and reviewed in (Fu et al. 2000, Fu 2001, Fu 2002, Better et al. 2008). On the one hand, simulation is an approximation to the real world, and in most cases it is impossible to find a *good enough* solution by enumerating the possible scenarios in huge search spaces, therefore simulation needs optimization techniques to provide some guidance towards a global optimal solution. On the other hand, without the help of simulation, many real world problems are too complicate to be modeled by explicit mathematical formulations and traditional optimization techniques (*e.g.*, gradient-based approaches) may not achieve satisfactory results. This has led to a major dilemma for the researchers who want to approximate the real world as closely as possible and find a *good enough* solution at the same time. According to the classification in (Fu 2002), the first situation is *optimization for simulation*, while the other is called *simulation for optimization*.

In the M&S community, ontologies have been proposed and constructed to facilitate the sharing of domain knowledge. The Discrete-event Modeling Ontology (DeMO) (Miller et al. 2004, Silver et al. 2010) represents the domain of discrete-event modeling and COmponent-oriented Simulation and Modeling Ontology (COSMO) (Teo and Szabo 2008) describes the simulation components and compositions from a component-oriented perspective. Process Interaction Modeling Ontology for Discrete-event Simulations (PIMODES) (Lacy 2006) facilitates the exchange of model information between various simulation software

package. The Discrete-event Simulation Ontology (DeSO) (Miller et al. 2007) is a small prototype ontology for discrete-event simulation.

In the research area of simulation optimization, the previous effort has focused almost exclusively on the optimization techniques and application of specific mathematical programs to practical simulation problems. Surveys and reviews of Simulation Optimization from various aspects are given in (Andradottir 1998, Swisher et al. 2000, Fu 2001, Ólafsson and Kim 2002, April et al. 2003, Fu et al. 2005, Fu et al. 2008). A review focusing on both gradient-based techniques (for continuous parameter estimation) and random search methods (for discrete parameter estimation) is presented in (Andradottir 1998). A review of optimization techniques for discrete-event simulation is given in (Swisher et al. 2000) covering both continuous and discrete input parameters. Discrete simulation optimization is reviewed in (Ólafsson and Kim 2002). However, little work has been done to gather domain knowledge on optimization techniques to construct an ontology that can be shared with others in the M&S community. ONTOP (Witherell et al. 2007) is an ontology for engineering design optimization, available at http://edesign.ecs.umass.edu/ontologies/Framework2.0/Optimization_Model2.0.owl. It defines the class of Optimization Model, which includes objective function, input and output variables, constraints, etc. For our purposes, the hierarchical structure of the class Optimization Model is not organized suitably, and optimization problems (e.g., linear programming) and optimization methods (e.g., Simplex method) are defined in the same place. This brings two disadvantage: (1) it does not address the condition where one method can be applied to solve multiple problems, e.g., Simplex and its revisions can be used to solve both Linear Programming and Quadratic programming problems; (2) adding new mathematical programs can be difficult. For example, quadratically constrained, quadratic program (QCQP) cannot find a suitable position because it can be put either under Nonlinearly_Constrained or Quadratic_Programming.

5 ONTOLOGY FOR SIMULATION OPTIMIZATION

In order to describe numerous simulation optimization methods, some common building block components and taxonomy need to be defined first and then be used to characterize the particular optimization problems and specific methods. The top-level abstract classes of SoPT are Optimization Component, Optimization Problem and Optimization Method. The relationships among them are shown in Figure 4.

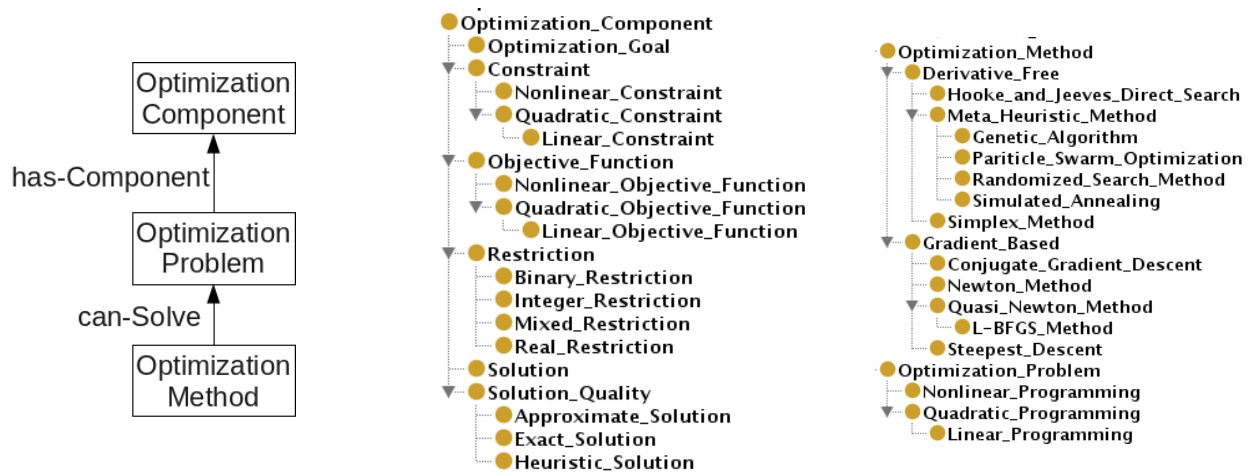


Figure 4: (Left) Top level abstract classes for SoPT ontology. (Middle) Components involved in Simulation Optimization. (Right) Optimization Method and Optimization Problem.

5.1 Optimization Component

An optimization problem consists of several parts, the most important of which are the objective function, the set of constraints and the domain over which optimization is to be performed. It is also important to specify the goal of the optimization (*e.g.*, $\min f(\mathbf{x})$ or $\max E[F(\mathbf{x})]$, etc.) and the desired solution quality.

The subclasses of Optimization Component, as shown in Figure 4, include data types and restrictions of input variables, objective function, constraints, solution, solution quality and optimization goal:

- Data Types: input variables may be represented in the form of vector or single element, which can be used as the initial values and part of a Solution.
- Restriction: input variables may have restrictions on the types of numbers, *e.g.*, real, integer, binary and mixed (real/integer/binary).
- Objective Function: it can be a single objective or multiple objective function by using a data property called `has-MultiObjectives` and needs to be represented as specifically as possible, so that more specific methods for this particular problem can be found.
- Constraints: they have the forms of inequality or equality, and may be unconstrained, linear, quadratic and nonlinear.
- Solution: it contains the output vector and value of the objective function.
- Solution Quality: it is used by both the Optimization Problem and Optimization Method classes to specify which solution quality is desired for certain optimization problem or which solution quality can be achieved by certain optimization methods.
- Optimization Goal: it specifies the type of optimization desired on the objective function.

5.2 Optimization Problem

The subclasses within the Optimization Problem taxonomy, Linear Programming (LP), Quadratic Programming (QP) and Nonlinear Programming (NLP) have object properties that link to the subclasses of both Optimization Component and Optimization Method.

Based on the available optimization components, the aforementioned optimization problems (LP (8) and QP (10)) can be represented as instances of the ontological class of Optimization Problem. Various optimization components, such as constraints, restriction, solution quality can be shared between both instances. Although LP and QP have different forms of objective functions, both can also share the same input variable vector and cost coefficient vector as LP is special case of QP. The difference is that LP has a zero-matrix for its quadratic coefficient matrix Q while QP has a non-zero Q .

5.3 Optimization Method

As mentioned, our primary interest is focused on iterative interaction between the optimizer and simulator. Therefore, at this time, we do not include Response Surface Methodology (RSM) in the ontology. Although meta-modeling is presently not explicitly included, we plan to add it, when we provide some cross linkage between SoPT and DeMO. A meta-model could be viewed as an alternative path in Figure 3.

The Optimization Method class in SoPT is shown in Figure 4. Due to the wide variety of methods used for simulation optimization, it is difficult to determine where to start. Because we are focusing on iterative interaction between the simulator and optimizer, we concentrate on what the optimizer needs from the simulator, a set of response vectors estimating a portion of response surface, and how the optimizer explores the parameter space using search techniques. A simple illustration is a gradient based technique (*e.g.*, steepest descent or conjugate gradient), where search involves direction determination and how far to move in that direction (line search).

5.3.1 Surface Sampling

As a single data point cannot provide enough useful information regarding huge search space, surface sampling is crucial to determine $\{\mathbf{x}\}^i$. For example, gradient based, newton and quasi-newton methods compute first-order derivatives and/or second-order derivatives for d parameters, either directly or numerically. When the objective function becomes nondifferentiable or noncontinuous, derivative free methods may be helpful. For one specific point within an n -dimensional problem space, downhill simplex method maintains $n + 1$ points, pattern search maintains a set of points called a pattern, and random search samples from a hypersphere around the current point. Heuristic methods often maintain a population of candidates, such as Genetic Algorithms and Particle Swarm Optimization. Although Simulated Annealing only keeps track of a single solution, it will choose from a number of its neighbors.

5.3.2 Search Techniques

Many optimization algorithms work in the following fashion: Iteratively establish a search direction (*e.g.*, -gradient for Steepest Descent) and then move in this direction by an amount determined by a line search algorithm that either optimizes or provides sufficient decrease of the objective function. An alternative is to use trust regions instead of line search.

Response Surface Methodology (RSM) fits a surface to a portion of the domain D and finds the optimal solution, *e.g.*, using QP $\frac{1}{2}\mathbf{x}^T Q\mathbf{x} + \mathbf{c}^T\mathbf{x}$, $A\mathbf{x} \geq \mathbf{b}$. Meta-Modeling replaces a complex, stochastic simulation with a simpler, likely deterministic meta-model.

5.4 Evaluation

Given the existing DeMO model and a well-populated SoPT ontology, optimization routines can be proposed automatically based on the model information and the capabilities of available optimization routines. This can be achieved by either applying an inference engine or running the simulation and investigating the response surface. For example, by examining the Q matrix as well as the size of the problem, rules can be used to select between active set and interior point methods for quadratic programming.

6 CONCLUSIONS AND FUTURE WORK

The major contributions of this paper are as follows: (1) investigated loosely-coupled software architecture for simulation optimization based on two scenarios in the bioinformatics area. (2) developed an ontology for simulation optimization, SoPT, to represent three fundamental concepts, namely Optimization Component, Optimization Problem and Optimization Method, and their relationships in simulation optimization. The ontology, SoPT is available at <http://www.cs.uga.edu/~jam/jsim/DeMO/>.

Our future work includes (1) enrichment of SoPT to cover RSM and meta-modeling, (2) incorporation of stochastic optimization, and (3) development of a small domain specific language (DSL) to transform the ontological form of optimization routines into actual executable application code (*e.g.*, targeting JSIM or ScalaTion (Miller et al. 2010)).

REFERENCES

- Andradottir, S. 1998. "A Review of Simulation Optimization Techniques". In *Proceedings of the 1998 Winter Simulation Conference*, edited by D. J. Medeiros, E. F. Watson, J. S. Carson, and M. S. Manivannan, 151–158. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.
- April, J., F. Glover, J. P. Kelly, and M. Laguna. 2003. "Practical Introduction to Simulation Optimization". In *Proceedings of the 2003 Winter Simulation Conference*, edited by S. Chick, P. J. Sánchez, D. Ferrin, and D. J. Morrice, 71–78. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

- Better, M., F. Glover, G. Kochenberger, and H. Wang. 2008. "Simulation Optimization: Applications in Risk Management". *International Journal of Information Technology & Decision Making (IJITDM)* 7 (4): 571–581.
- Boyd, S., and L. Vandenberghe. 2004. *Convex Optimization*. Cambridge University Press.
- Fu, M. C. 2001. "Simulation Optimization". In *Proceedings of the 2001 Winter Simulation Conference*, edited by B. A. Peters, J. S. Smith, D. J. Medeiros, and M. W. Rohrer, 53–61. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.
- Fu, M. C. 2002. "Feature Article: Optimization for simulation: Theory vs. Practice". *INFORMS JOURNAL ON COMPUTING* 14 (3): 192–215.
- Fu, M. C., S. Andradóttir, J. S. Carson, F. Glover, C. R. Harrell, Y.-C. Ho, J. P. Kelly, and S. M. Robinson. 2000. "Integrating Optimization and Simulation: Research and Practice". In *Proceedings of the 2000 Winter Simulation Conference*, edited by J. A. Joines, R. R. Barton, K. Kang, and P. A. Fishwick, 610–616. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.
- Fu, M. C., C.-H. Chen, and L. Shi. 2008. "Some Topics for Simulation Optimization". In *Proceedings of the 2008 Winter Simulation Conference*, edited by S. J. Mason, R. R. Hill, L. Moench, O. Rose, T. Jefferson, and J. W. Fowler, 27–38. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.
- Fu, M. C., F. W. Glover, and J. April. 2005. "Simulation Optimization: a Review, New Developments, and Applications". In *Proceedings of the 2005 Winter Simulation Conference*, edited by M. E. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Joines, pp 83–95. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.
- Han, J., J. A. Miller, M. Fang, L. Wells, K. J. Kochut, R. Ranzinger, and W. S. York. 2011. "GlycoQuant: An Automated Simulation Framework Targeting Isotopic Labeling Strategies in MS-Based Quantitative Glycomics". Technical report.
- Heinrich, R., and S. Schuster. 1998. "The Modelling of Metabolic Systems. Structure, Control and Optimality". *Biosystems* 47 (1-2): 61 – 77.
- Hellerstein, M. K., and R. A. Neese. 1999. "Mass Isotopomer Distribution Analysis at Eight Years: Theoretical, Analytic, and Experimental Considerations.". *The American journal of physiology* 276 (6 Pt 1): E1146–70.
- Inczédy, J., T. Lengyel, and M. A. Ure. 1998. *Compendium of Analytical Nomenclature Definitive Rules 1997 (the Orange Book)*. 3 ed. International Union of Pure and Applied Chemistry (IUPAC).
- Lacy, L. W. 2006. *Interchanging Discrete Event Simulation Process Interaction Models Using the Web Ontology Language—OWL*. Ph. D. thesis, University of Central Florida, Orlando, FL, USA. AAI3242447.
- Matsuno, H., Y. Tanaka, H. Aoshima, A. Doi, M. Matsui, and S. Miyano. 2003. "Biopathways Representation and Simulation on Hybrid Functional Petri Net". *In Silico Biology* 3 (3): 389 – 404.
- Miller, J. A., G. T. Baramidze, A. P. Sheth, and P. A. Fishwick. 2004. "Investigating Ontologies for Simulation Modeling". In *Proceedings of the 37th Annual Simulation Symposium*, edited by H. Karatza, ANSS '04, 55–63. Washington, DC, USA: IEEE Computer Society.
- Miller, J. A., J. Han, and M. Hybinette. 2010. "Using Domain Specific Languages for modeling and simulation: ScalaTion as a case study". In *Proceedings of the 2010 Winter Simulation Conference*, edited by B. Johansson, S. Jain, J. Montoya-Torres, J. Hagan, and E. Yücesan, 741–752. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.
- Miller, J. A., C. He, and J. L. Couto. 2007, may. *Handbook of Dynamic System Modeling*, Volume 14 of *Computer and Information Science Series*, Chapter Impact of the Semantic Web on Modeling and Simulation, 3–1 – 3–22. CRC Press.
- Nimmagadda, K. 2008. "Ontology Driven Simulation of Biochemical Pathways Using Hybrid Petri Nets". Master's thesis, University of Georgia, Athens GA, USA.
- Nocedal, J., and S. J. Wright. 2006. *Numerical Optimization*. 2nd ed. New York, USA: Springer.

- Ólafsson, S., and J. Kim. 2002. "Simulation optimization". In *Proceedings of the 2002 Winter Simulation Conference*, edited by E. Yücesan, C. H. Chen, J. L. Snowdon, and J. M. Charnes, 79–84. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.
- Rao, S. S. 2009. *Engineering Optimization: Theory and Practice*. 4th ed. New York: Wiley.
- Silver, G. A., K. R. Bellipady, J. A. Miller, W. S. York, and K. J. Kochut. 2009. "Supporting Interoperability Using the Discrete-event Modeling Ontology (DeMO)". In *Proceedings of the 2009 Winter Simulation Conference*, edited by M. D. Rossetti, R. R. Hill, B. Johansson, A. Dunkin, and R. G. Ingalls, 1399 – 1410. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.
- Silver, G. A., J. A. Miller, M. Hybinette, G. Baramidze, and W. S. York. 2010. "DeMO: An Ontology for Discrete-event Modeling and Simulation". *Simulation* 87 (9): 747–773.
- Swisher, J. R., P. D. Hyden, S. H. Jacobson, and L. W. Schruben. 2000. "A Survey of Simulation Optimization Techniques and Procedures". In *Proceedings of the 2000 Winter Simulation Conference*, edited by J. A. Joines, R. R. Barton, K. Kang, and P. A. Fishwick, 119–128. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.
- Teo, Y. M., and C. Szabo. 2008. "CODES: An Integrated Approach to Composable Modeling and Simulation". In *Proceedings of the 41st Annual Simulation Symposium*, edited by H. Karatza, 103–110. Washington, DC, USA: IEEE Computer Society.
- Witherell, P., S. Krishnamurthy, and I. R. Grosse. 2007. "Ontologies for Supporting Engineering Design Optimization". *Journal of Computing and Information Science in Engineering* 7 (2): 141–150.
- Zaia, J. 2010. "Mass Spectrometry and Glycomics". *OMICS: A Journal of Integrative Biology* 14 (4): 401–418.

AUTHOR BIOGRAPHIES

JUN HAN is a Ph.D. candidate in Computer Science at the University of Georgia. His research interests include modeling and simulation, ontologies, Web services and bioinformatics. Jun Han received a B.S. in Computer Science from BeiHang University and an M.S. in Computer Science from Institute of Software, Chinese Academy of Sciences. His email address is jun@cs.uga.edu.

JOHN A. MILLER is a Professor of Computer Science at the University of Georgia. His research interests include database systems, simulation, Web services and bioinformatics. Dr. Miller received the B.S. degree in Applied Mathematics from Northwestern University in 1980 and the M.S. and Ph.D. in Information and Computer Science from the Georgia Institute of Technology in 1982 and 1986, respectively. During his undergraduate education, he worked as a programmer at the Princeton Plasma Physics Laboratory. He is an Associate Editor for the ACM TOMACS, IEEE TSMC and SCS SIMULATION as well as an Editorial Board Member for the JOS and International Journal of SPM. His email address is jam@cs.uga.edu.

GREGORY A. SILVER is an Assistant Professor of Computer Information Systems at Anderson University and a Ph.D. candidate in Computer Science at the University of Georgia. Mr. Silver received his M.S. degree in Computer Information System from Georgia State University in 1996. His research interests include modeling and simulation, ontologies and Web services. His email address is gsilver@andersonuniversity.edu.