“Don’t Care” Modeling: A Logical Framework for Developing Predictive System Models[⋆]

Hillel Kugler[1], Amir Pnueli[1,2], Michael J. Stern[3], and E. Jane Albert Hubbard[1]

[1] New York University, New York, NY, USA
{kugler,amir}@cs.nyu.edu, jane.hubbard@nyu.edu
[2] The Weizmann Institute of Science, Rehovot, Israel
[3] Yale University, New Haven, CT, USA
Michael.Stern@yale.edu

Abstract. Analysis of biological data often requires an understanding of components of pathways and/or networks and their mutual dependency relationships. Such systems are often analyzed and understood from datasets made up of the states of the relevant components and a set of discrete outcomes or results. The analysis of these systems can be assisted by models that are consistent with the available data while being maximally predictive for untested conditions. Here, we present a method to construct such models for these types of systems. To maximize predictive capability, we introduce a set of “don’t care” (dc) Boolean variables that must be assigned values in order to obtain a concrete model. When a dc variable is set to 1, this indicates that the information from the corresponding component does not contribute to the observed result. Intuitively, more dc variables that are set to 1 maximizes both the potential predictive capability as well as the possibility of obtaining an inconsistent model. We thus formulate our problem as maximizing the number of dc variables that are set to 1, while retaining a model solution that is consistent and can explain all the given known data. This amounts to solving a quantified Boolean formula (QBF) with three levels of quantifier alternations, with a maximization goal for the dc variables. We have developed a prototype implementation to support our new modeling approach and are applying our method to part of a classical system in developmental biology describing fate specification of vulval precursor cells in the *C. elegans* nematode. Our work indicates that biological instances can serve as challenging and complex benchmarks for the formal-methods research community.

1 Introduction

Understanding a given complex system whose behavior can be observed, but whose behavioral program is not directly available, is an important yet difficult task. Examples of such systems include complex web services, software for which only the executable binary code is available, and legacy systems where the code is available but may be written in a language that is rarely used nowadays or lacking sufficient documentation and support from the original system

[⋆] This research was supported in part by NIH grant R24-GM066969.

developers. Our current work was motivated by a project that uses methods from software and system design to model and analyze biological systems. The behaviors of biological systems can be observed; however, the workings of their underlying behavioral programs are not directly available, and their elucidation is the subject of much biological research.

Analysis of biological data often requires an understanding of components of pathways and/or networks and their mutual dependency relationships. Such systems are often analyzed and understood from datasets made up of the states of the relevant components and a set of discrete outcomes or results. One type of biological example of such a system relates a "genotype" (the states of a set of genes, that can be either mutated or normal) and a resulting character trait (a "phenotype"). The understanding of the behavior of these systems is often constrained by the limited set of available condition-result data. The analysis of these types of systems can be assisted by models that are consistent with the available data while being maximally predictive for untested conditions. Here, we present a method to construct such models for these types of systems. Furthermore, our approach allows identifying those additional condition-result data that can most effectively constrain the set of possible models to those that match the behavior of the system. Our approach handles models with discrete variables, thus if the actual system variables are continuous, we assume that the domain has been discretized either manually or using other computational methods.

To maximize predictive capability, we introduce a set of "don't care" (dc) Boolean variables that must be assigned values in order to obtain a concrete model. When a dc variable is set to 1, this indicates that the information from the corresponding component does not contribute to the observed result. Thus the value of 1 denotes flexibility while the value of 0 denotes inflexibility. Intuitively, increasing the number of dc variables that are set to 1 increases both the potential predictive capability as well as the possibility of obtaining an inconsistent model. We thus formulate our problem as maximizing the number of dc variables that are set to 1, while retaining a model solution that is consistent and can explain all the given data. This amounts to solving a quantified Boolean formula (QBF) with three levels of quantifier alternations, with a maximization goal for the dc variables. We first show how our problem can be solved using QBF solvers, and later demonstrate how the special structure of our QBF instances can be used to reduce the problem and allow a more efficient solution.

We are applying our method to part of a classical system in developmental biology describing fate specification of vulval precursor cells in the *C. elegans* nematode. This is a well-characterized system that provides sufficient complexity to serve as a test case for our studies. Our work indicates that biological instances can serve as challenging and complex benchmarks for the formal-methods research community.

2 Example

This section introduces the problem statement, logical representation and possible solutions through a very simple example. For ease of presentation we make some simplifying assumptions in this section [1]; the more general case can be treated by adjusting this framework.

Let Boolean variable x denote a possible genetic locus:

$$x = \begin{cases} 1 & \text{if mutated} \\ 0 & \text{if wild-type} \end{cases}$$

In this simple example we consider three possible genetic loci with corresponding variables x_1, x_2, x_3. We assume three possible phenotypic outcomes, denoted by variable $y \in \{1, 2, 3\}$. The possible phenotypic outcomes are assumed to be disjoint, thus it is not possible to measure for example both $y = 1$ and $y = 2$.

We assume two experiments were performed and show the direct logical representation:

Experiment 1. x_1 is mutated and all others are wild-type. The phenotype obtained was $y = 1$. The logical representation is:

$$x_1 = 1 \wedge x_2 = 0 \wedge x_3 = 0 \rightarrow y = 1$$

Experiment 2. x_2 is mutated and all others are wild-type. The phenotype obtained was $y = 2$. The logical representation is:

$$x_1 = 0 \wedge x_2 = 1 \wedge x_3 = 0 \rightarrow y = 2$$

We are interested in representing and understanding the connection between genotype and phenotype. For this purpose we would like to construct a model that explains experimental results, and can make predictions about new experiments.

Considering our simple example, a first attempt for a model is:

$$(x_1 = 1 \wedge x_2 = 0 \wedge x_3 = 0 \rightarrow y = 1) \wedge (x_1 = 0 \wedge x_2 = 1 \wedge x_3 = 0 \rightarrow y = 2)$$

The above model, being a conjunction of the two formulas representing the experimental results, is consistent with the respective experimental results, but does not provide any additional predictions about new experiments. For example, considering the experiment in which x_2 and x_3 are mutated while x_1 remains wild-type, the model predicts nothing about the phenotype y. Formally, given the assignment $x_1 = 0$, $x_2 = 1$, $x_3 = 1$ the model formula evaluates to *true* for any assignments of the phenotype y, which can be assigned to values in the range $\{1, 2, 3\}$. The meaning is that any phenotypic outcome is possible, which amounts to no prediction.

To enable prediction the model should allow generalization from the experimental results. Our second attempt for constructing a model is:

$$(x_1 = 1 \rightarrow y = 1) \wedge (x_2 = 1 \rightarrow y = 2)$$

[1] Simplifying assumptions include single variable for phenotype representation instead of cross product using several variables, single value for phenotype measurement instead of disjunction of possible outcomes.

The idea here is to generalize by recording explicitly only the information about the mutations, omitting the wild-type background. The problem is that this model is inconsistent. Specifically, for the experimental setup in which both x_1 and x_2 are mutated, obtained by assigning $x_1 = 1$, $x_2 = 1$ to the formula, no assignment to y can satisfy it since it implies both $y = 1$ and $y = 2$. Variable y denotes a phenotype output, which we assume has disjoint values, thus it is not possible to have two phenotypic outcomes simultaneously, and hence the inconsistency.

A third and final attempt is to generalize as much as possible, but avoid inconsistent models. For this goal, we add dc variables, which are Boolean variables of the form $d \in \{0, 1\}$. When $d = 1$ the phenotype of the system is unaffected by this specific mutation in a given genetic background (we "don't care" whether it is mutated or not in this situation). When d = 0, the mutation is important in this context.

In this formulation we construct for our example a general model as follows:

$$(x_1 = 1 \wedge x_2 <= d_1^2 \wedge x_3 <= d_1^3 \rightarrow y = 1) \wedge (x_1 <= d_2^1 \wedge x_2 = 1 \wedge x_3 <= d_2^3 \rightarrow y = 2)$$

Let us explain the formula in some more detail. Here d_i^j denotes the dc variable for locus j in experiment i. A concrete model is obtained by assigning values to all the dc variables d_i^j. The expression $x_j <= d_i^j$ allows the flexibility to determine whether or not it is important to keep variable x_j in it's "normal" restrictive setting ($x_j = 0$) for obtaining the outcome measured in experiment i. If we set the dc variable d_i^j to 1, then the expression $x_j <= d_i^j$ evaluates to $x_j <= 1$ which is equivalent to true, since x_j is a Boolean variable and the expression holds for values 0 (since $0 <= 1$) and 1 (since $1 <= 1$). In this case, the value of x_j in experiment i does not affect the phenotypic outcome. If, on the other hand, we set d_i^j to 0, the expression $x_j <= d_i^j$ evaluates to $x_j <= 0$ which, for Boolean variable x_j, is equivalent to $x_j = 0$, meaning that the value of variable x_j in experiment i is important to the phenotypic outcome.

Our first two attempts for constructing models are special cases in this formulation. If all dc variables are set to 0, this corresponds to our first model, which is not predictive, while assigning 1 to all dc variables corresponds to the second model which is inconsistent [2] . To maximize the predictiveness of consistent models, we would like to be able to maximize the number of dc variables that are set to 1, while maintaining the requirement that the model is still consistent. A model is consistent if for any assignment to the variables x_1, x_2, x_3 there exists an assignment to the phenotype variable y such that the formula evaluates to true. Intuitively, this corresponds to the fact that for any experimental setup that can be prepared in the lab some phenotype will be measured.

[2] In general, assigning 1 to all the dc variables gives an inconsistent model, except for the degenerate case in which there are no two experiments that differ on their phenotypic outcome.

In our example we have four dc variables : $d_1^2, d_1^3, d_2^1, d_2^3$ (two experimental setups, each having two genetic loci that potentially can be mutated without affecting the phenotypic outcome). The maximal number of dc variables that can be assigned 1 while remaining with a consistent model is 3, since assigning all 4 dc variables to 1 results in an inconsistent model. A possible solution with 3 dc variables set to 1 is $d_1^2 = 1$, $d_1^3 = 1$, $d_2^3 = 1$ and $d_2^1 = 0$. Assigning these values for the dc variables we obtain :

$$(x_1 = 1 \wedge x_2 <= 1 \wedge x_3 <= 1 \rightarrow y = 1) \wedge (x_1 <= 0 \wedge x_2 = 1 \wedge x_3 <= 1 \rightarrow y = 2)$$

$$= (x_1 = 1 \rightarrow y = 1) \wedge (x_1 = 0 \wedge x_2 = 1 \rightarrow y = 2)$$

Unlike the first attempted model that was consistent but not predictive, this model is consistent and allows some predictions, for example for genotype $x_1 = 0$, $x_2 = 1$, $x_3 = 1$ the predicted phenotype is $y = 2$. The maximal number of dc variables that can be assigned 1 while keeping the model consistent does not in general determine a unique solution. In our example another maximal solution is $d_2^1 = 1$, $d_1^3 = 1$, $d_2^3 = 1$ and $d_1^2 = 0$.

3 Problem Formulation

This section formalizes the concepts of logical representation and model construction that were intuitively explained through the example in Section 2. It defines the mathematical problem we are interested in and then shows how we go about solving it.

We are interested in understanding the observable behavioral outcome of a system (defined by output variables) as a function of the experimental setup (defined by input variables) . The experimental setup for the system is controlled by binary input variables $x_1, x_2, \cdots x_n$. The "normal" value of an input variable is 0, and a change to the value (for example by a genetic mutation) is specified by assigning 1 to the variable.

The outcome is represented by an output variable y, that can assume a discrete and finite [3] set of values. In the general case, the phenotypic output specified by variable y can be a result of measuring several orthogonal phenotypic outputs, designated by variables $y_1, y_2, \cdots y_m$. In this case the value of the output behavior y can be viewed as a cross product of the values of each of the orthogonal phenotypes, $y = y_1 \times y_2 \cdots \times y_m$.

Given a dataset consisting of the values for the input variables and a discrete outcome result for the output variable we construct a formula of the form:

$$((x_1 = 1 \wedge x_2 <= d_1^2 \wedge x_3 <= d_1^3 \cdots \wedge x_n <= d_1^n) \rightarrow y = p_1) \wedge \quad (1)$$

[3] This is a simplifying assumption. If the outcomes are continuous values we assume that the domain has been discretized according to some biological criteria either manually or using other computational methods.

$$((x_1 <= d_2^1 \wedge x_2 = 1 \wedge x_3 <= d_2^3 \cdots \wedge x_n <= d_2^n) \rightarrow y = p_2) \wedge$$
$$\vdots$$
$$((x_1 <= d_s^1 \cdots \wedge x_q = 1 \cdots \wedge x_r = 1 \cdots \wedge x_n <= d_s^n) \rightarrow y = p_s) \wedge$$
$$\vdots$$
$$((x_1 <= d_l^1 \wedge x_2 <= d_l^2 \wedge x_3 = d_l^3 \cdots \wedge x_n = 1) \rightarrow y = p_l)$$

Let us now explain in detail how we construct a concrete formula of the same form as Formula 1 above from the experimental datasets that are available. Each line in the formula corresponds to an experiment, composed of an experimental setup and the phenotypic outcome measured. Each line is written as a logical implication, where the left-hand side of the implication corresponds to the experimental setup defined by the input variables, while the right-hand side corresponds to the phenotypic output defined by the output variables. For a given experiment, if the input variable x_j was changed from its "normal" setting, the left-hand side will contain the conjunct $x_j = 1$, while for an input variable that was set to its "normal" value $x_j = 0$, we will introduce a new dc variable and add the conjunct $x_j <= d_i^j$. Here d_i^j denotes the dc variable corresponding to input variable j in experiment i. For experiment i, if the phenotype measured was p_i, then the right-hand side of the corresponding implication will contain the expression $y = p_i$. The notation here can handle the more general case where y is determined by the cross product of the orthogonal phenotypic outputs of variables $y_1, y_2, \cdots y_m$, in which case $p_i = (p_{i,1}, p_{i,2} \cdots p_{i,m})$ where $p_{i,k}$ is the phenotypic output for variable y_k.

Consider for example the first line from the formula. It was constructed based on an experiment in which variable x_1 was perturbed, while all the other input variables assumed their "normal" value, and the phenotypic outcome $y = p_1$ was measured. Thus the corresponding logical representation we obtain for this experiment is:

$$((x_1 = 1 \wedge x_2 <= d_1^2 \wedge x_3 <= d_1^3 \cdots \wedge x_n <= d_1^n) \rightarrow y = p_1)$$

Formula 1 describes the results of l different experiments. Each experiment is not necessarily restricted to single variable changes, for example, experiment s in the above formula shows a case were both input variables x_q and x_r were changed, while the other input variables remain "normal", as shown in the corresponding logical representation for the experiment:

$$((x_1 <= d_s^1 \cdots \wedge x_q = 1 \cdots \wedge x_r = 1 \cdots \wedge x_n <= d_s^n) \rightarrow y = p_s)$$

Our approach thus allows encoding experiments with any number of input variables changes, including double, triple and higher degrees of variable changes. It is also not necessary that all results for single variable changes appear in the formula; the formula will just encode all the information that is available in the experimental dataset. Another point worth noting is that our approach can accommodate systems where the outcome may be nondeterministic. If an experiment is repeated several times with the same experimental setup and different phenotypic outputs are observed, then the right-hand side for the corresponding experiment in the formula will be a disjunction of the observed phenotypes. Thus the notation $y = p_i$ can stand for a set of possible outcomes, where $p_i = p_i^1 \vee p_i^2 \cdots \vee p_i^k$ if k different outcomes were observed when the experiment was repeated. Our current work does not consider a probabilistic distribution related to the number of times each outcome is measured when repeating the experiment, only the set of observed outcomes.

We call a formula of the type appearing in Formula 1 a **generic model** formula, since a concrete model is obtained from it by assigning values to all the dc variables.

Definition 1. *A* **model** *is a formula obtained from a generic model formula by assigning values* 0 *or* 1 *to all the dc variables.*

When assigning values to all the dc variables to obtain a model, the formula can be simplified as follows. For each dc variable that is assigned the value 0 the expression $x_j <= d_i^j$ is replaced by $x_j = 0$, while for each dc variable that is assigned the value 1 the expression $x_j <= d_i^j$ is replaced by *true*, which is then used to further simplify the formula.

Definition 2. *The* **prediction** *that model ϕ gives about a certain experimental setup, is obtained by assigning the values of all input variables according to the experimental setup resulting in a formula ϕ' and then finding all the assignments to the output variables that satisfy formula ϕ'.*

Definition 3. *A model is* **consistent** *if for any assignment to the input variables $x_1, x_2, \cdots x_n$ there exists an assignment to the output variables $y_1, y_2, \cdots y_m$ such that the formula evaluates to true.*

Definition 4. *A model with k dc variables set to* 1 *is* **maximally predictive** *if it is consistent and any model with more than k dc variables set to* 1 *is inconsistent.*

We are interested in developing efficient algorithms for finding a maximally predictive model, a topic that is studied in the next section.

4 Solutions

According to Definition 3 checking the consistency of a model ϕ amounts to checking the satisfiability of the following quantified Boolean formula:

$$\forall x_1, x_2, \cdots x_n \exists y_1, y_2 \cdots y_m \phi$$

The intuition behind these definitions is that we require that the x_i variables are universally quantified since they are input variables representing an experimental setup that in principle can be set to any possible combination, while the existential quantification of the y_j output variables represents the fact that for any experiment that is done there will be some phenotypic output measured in the biological system.

Proposition 1. *For any given generic model formula, if there are no consistent models with exactly k dc variables set to* 1*, then a maximally predictive model has less than k dc variables set to* 1*.*

Proof. Omitted from this version of the paper due to space limitations.

Given a generic model formula, the existence of a consistent model can be formulated as a quantified Boolean formula as follows:

$$\exists d_1, d_2 \cdots d_p \forall x_1, x_2, \cdots x_n \exists y_1, y_2 \cdots y_m \phi$$

This is a quantified Boolean formula with three levels of quantifier alternations. The outermost existential quantification over the dc variables corresponds to fixing a concrete model, the universal quantification over the input variables and existential quantification over the output variables corresponds to the requirement that the fixed model is consistent.

If we encode in the formula $\chi(k)$ the requirement that exactly k dc variables are assigned to 1, then the question of the existence of a consistent model with exactly k dc variables set to 1 is reduced to the satisfiability of the formula:

$$\exists d_1, d_2 \cdots d_p \forall x_1, x_2, \cdots x_n \exists y_1, y_2 \cdots y_m \phi \wedge \chi(k)$$

Given an algorithm for checking the satisfiability of such a formula, we can use the result in Proposition 1, and perform a binary search on k, the number of dc variables set to 1, to find the maximal k for which a consistent model exists, and obtain a maximally predictive model.

4.1 Implementing the Basic Algorithm

As shown above our problem amounts to solving a quantified Boolean formula with three levels of quantifier alternations, with a maximization goal for the dc variables. We next provide some information on a direct implementation to solve our problem, using two tools, one based on a binary decision diagrams (BDDs) [1] as implemented in the TLV tool [13], the other directly on a QBF solver using the Quaffle tool [20]. As will be shown later, the special structure of our QBF instances can be used to reduce the problem and allow a more efficient solution. We still explain the direct implementation for presentation purposes; it also may be the case that for various extensions of the problem the direct solution is required.

BDD Solver. TLV [13] is a symbolic model checker that uses binary decision diagrams as the basic underlying data structure. One of the strong aspects of TLV is that it provides a high-level scripting language called TLV-basic, which is especially convenient for experimenting with the design and implementation of new verification algorithms.

We have implemented in TLV the direct algorithm based on performing a binary search on the value of k, the number of dc variables set to 1. Each iteration solves the QBF formula described above with the constraint on the value of k. Universal and existential quantification are supported as existing functions in TLV (forall, exists) and are based on direct manipulation of the BDDs. The encoding of the generic model formula in TLV is straightforward as all logical operators are directly supported. At the end of each iteration we get a BDD that represents the (possibly empty) set of all consistent models with the current value of dc variables set to 1. This turns out to be useful since at the end of the algorithm we obtain the set of all maximally predictive models. The main disadvantage of using TLV is in terms of performance. Solving QBF formulas using BDD technology is not efficient, and for this reason we have experimented with applying a QBF solver to our problem.

QBF Solver. To allow applying a QBF solver to our problem we have to encode it in one of the standard formats accepted by these tools. QBF solvers typically accept only Boolean variables, thus the output variables in our problem that are not necessarily Boolean must be encoded using several Boolean variables. The input variables and the dc variables are originally Boolean so they can be accommodated directly. Another requirement of standard formats is that the propositional part of the QBF formula is written as a CNF formula, which requires some modifications to the generic model formula. To add the constraints on the parameter k, the number of dc variables assigned to 1, we created a circuit for performing the addition of dc variables and translated the circuit to CNF using a canonical translation. We have experimented with the QBF solver Quaffle [20] on some instances we have generated manually. We are currently working on developing a program that will handle all the translations automatically, and given a generic model formula in the high level representation as that of Formula 1 and a value for the parameter k will generate an instance in the standard QBF format. This will allow a much more effective use of the QBF solvers, in fact our plan is to make these instances publicly available, since they can serve as interesting benchmarks in QBF evaluations and libraries [6].

4.2 Improved Algorithm

We now show for deterministic systems[4] how to find a maximally predictive model in a more efficient way, by reducing the original problem to that of solving a set of inequalities involving only the dc variables d_i^j where each inequality is of the form $\sum d_i^j < C$ for an integer constant C. We start with a generic model formula of the form of Formula 1 and construct the set of inequalities over the dc variables. A generic model formula is of the following form:

$$((x_1 = 1 \wedge x_2 <= d_1^2 \wedge x_3 <= d_1^3 \cdots \wedge x_n <= d_1^n) \rightarrow y = p_1) \wedge$$

[4] We have also extended the algorithm to deal with the general case of nondeterministic systems, allowing several phenotypic outputs for the same experimental setup. Due to space limitations and to allow a simpler presentation this extension is omitted from this version of the paper.

$$((x_1 <= d_2^1 \wedge x_2 = 1 \wedge x_3 <= d_2^3 \cdots \wedge x_n <= d_2^n) \rightarrow y = p_2) \wedge$$
$$\vdots$$
$$((x_1 <= d_l^1 \wedge x_2 <= d_l^2 \wedge x_3 = d_l^3 \cdots \wedge x_n = 1) \rightarrow y = p_l)$$

The formula is a conjunction of implications, each one appears in a separate line in the formula above, each line corresponds to an experimental setup and phenotypic outcome measured. For each pair of lines i, j if the phenotypic outputs are different, we add the following constraint:

$$\sum_{x_q=1 \text{ in line j}} d_i^q + \sum_{x_q=1 \text{ in line i}} d_j^q < C_{i,j} \tag{2}$$

Here $C_{i,j}$ is the number of dc variables appearing in the sums of the left-hand side of the inequality. This is equal to the number of input variables set to 1 in experiment i plus the number of input variables set to 1 in experiment j minus twice the number of input variables that are set to 1 in both experiments i and j. We subtract this number since if an input variable is set to 1 in both experiments i and j it was counted in the first two terms but there are no corresponding dc variables in the generic model formula since they are added only when an input variable is set to 0 in a given experiment.

Following this construction we obtain a set of inequalities on the dc variables. The input variables and output variables do not appear in these inequalities. The number of equations is at most quadratic in the number of experiments, or equivalently in the number of lines in the generic model formula. We will next prove that to find a maximally predictive model it is sufficient to solve the obtained set of inequalities under the maximization goal for the number of dc variables set to 1. Before stating and proving the relevant theorem we illustrate its application to the simple example described in Section 2.

The generic model formula we have for this example is:

$$((x_1 = 1 \wedge x_2 <= d_1^2 \wedge x_3 <= d_1^3) \rightarrow y = 1)$$
$$\wedge((x_1 <= d_2^1 \wedge x_2 = 1 \wedge x_3 <= d_2^3) \rightarrow y = 2)$$

It was derived from two experiments that have different phenotypic outputs, $y = 1$ and $y = 2$. We therefore add the following inequality:

$$d_1^2 + d_2^1 < 2$$

We ask what is the maximal k that satisfies the inequality and

$$d_1^2 + d_1^3 + d_2^1 + d_2^3 = k$$

The maximal solution is $k = 3$, and there are indeed solutions for the original formula with 3 dc variables set to 1 and no solutions with all 4 dc variables set to 1 as shown in 2.

Theorem 1. *For a given generic model formula, a model ϕ defined by an assignment D to the dc variables is consistent iff D satisfies the set of all inequalities defined in Formula 2.*

Proof. ($\Rightarrow$) Assume that the model ϕ defined by assignment D is consistent. By Definition 3 this holds if the following formula is satisfiable:

$$\forall x_1, x_2, \cdots x_n \exists y_1, y_2 \cdots y_m \phi$$

We need to show that D satisfies the set of inequalities. Assume towards contradiction that there is an inequality constructed from the pair of experiments i, j that does not hold:

$$\sum_{x_q=1 \text{ in line j}} d_i^q + \sum_{x_q=1 \text{ in line i}} d_j^q < C_{i,j}$$

This inequality does not hold if:

$$\sum_{x_q=1 \text{ in line j}} d_i^q + \sum_{x_q=1 \text{ in line i}} d_j^q = C_{i,j}$$

We get this equation only in the case that D assigns 1 to all the dc variables appearing in the left-hand side of the equation, since only then the sum of these dc variables is equal to the number of these dc variables. Consider an experimental setup that assigns the value 1 to the union of all input variables that are assigned 1 in either experiment i or experiment j (or both). All the other input variables are set to 0. This experimental setup satisfies the left-hand side of the implications for both lines i and j in the formula ϕ and thus both phenotypes defined by the right-hand side must occur, but the two original experiments i, j have different phenotypic outcomes, since only in this case we constructed the inequality. As a result for this new experimental setup no phenotype can satisfy the model ϕ, in contradiction to the assumption that ϕ is a consistent model. Thus assignment D satisfies all the inequalities defined in Formula 2.

($\Leftarrow$)

Assume that assignment D satisfies the set of all inequalities defined in Formula 2. We need to show that the model ϕ defined by assignment D is consistent. Assume towards contradiction that the model ϕ is not consistent, thus according to Definition 3 the following formula is not satisfiable:

$$\forall x_1, x_2, \cdots x_n \exists y_1, y_2 \cdots y_m \phi$$

If the formula is not satisfiable there exists an assignment for the input variables $x_1, x_2, \cdots x_n$ such that for any assignment of the output variables $y_1, y_2 \cdots y_m$ the formula evaluates to false. The formula ϕ is composed of a conjunction of implications. Consider the assignment to the input variables $x_1, x_2, \cdots x_n$ in which for any assignment to the output variables the formula evaluates to false. For this to occur there are at least two lines for which the left-hand side of the implication is satisfied and the phenotypic outcomes are different. Otherwise assigning the output variables to the unique fate defined by the left-hand side expressions that are true, will satisfy the formula. Considering these two lines, for their left-hand side expressions to hold, the dc variables for the union of input variables that are set to 1 in each of the experiments, must be set to 1 in the assignment D. Thus denoting these two lines i and j the following equation is satisfied.

$$\sum_{x_q=1 \text{ in line j}} d_i^q + \sum_{x_q=1 \text{ in line i}} d_j^q = C_{i,j}$$

And this is a direct violation of one of the inequalities defined in Formula 2:

$$\sum_{x_q=1 \text{ in line j}} d_i^q + \sum_{x_q=1 \text{ in line i}} d_j^q < C_{i,j}$$

In contradiction to our assumption that D satisfies all inequalities defined in Formula 2, therefor the model ϕ is consistent.

5 Biological Application

A great deal of biological research currently focuses on the analysis of molecular and cellular pathways and networks. An understanding of components of pathways and/or networks and their interdependencies is an important aspect of these studies. For example, a set of genes that affect a similar process (either positively or negatively) may be characterized by the effect of specific mutations of these genes on the outcome of the process. Data describing the outcome of combinations of such mutations may add additional information. Of particular interest in constructing pathways and networks is information that distinguishes between conditions in which the genotype of one genetic component in the pathway (or activity of a gene or protein component) is or is not relevant to the final outcome. Genetic epistasis analysis and analysis of modifier effects have been used to great advantage to parse many pathways [9]. With the advent of large-scale molecular-genetic data collection, the data space of genetic interactions is becoming increasingly unwieldy, even for relatively simple processes. It is, therefore, advantageous to identify methods by which dependency relationships between pathway components can be analyzed and modeled. Models of a subset of the data serve two general purposes: they may be used to predict the outcome of genetic combinations that have not been tested, and they may provide a means to readily identify the key combinatorial experiments that can be performed to distinguish between two or more equally viable models.

We are applying our method to part of a classical system in developmental biology describing fate specification of cells in the *C. elegans* nematode. *C. elegans* is widely studied in many labs worldwide where it serves as a model organism. Various fundamental biological phenomena that also exist in higher-level organisms can be studied in effective ways in *C. elegans.* The field has taken particular advantage of the genetic approach to investigating biological processes whereby a process is perturbed by genetic mutation and the genes involved in the normal process are thereby identified. In addition, because the animals are relatively simple and the entire cell lineage is known, cell ablation experiments (in which particular cells are removed from an intact animal using a laser) have also been instrumental in discovering cell-cell interactions. The combinatorial effects of various mutations on the process that they perturb individually and the effects of combinations of mutations with cell ablations has generated a large body of complex data. Further, the molecular nature and biochemical roles of many of the gene products involved in developmental processes link the functional perturbation data to particular biochemical pathways and networks [19].

Our application focused on the process of fate specification of vulval precursor cells. The vulva is a structure through which eggs are laid. This structure derives from three cells within a set of six cells with an equivalent set of multiple developmental potentials. Due to their potential to participate in the formation of the vulva, they are known as vulval precursor cells (VPCs). Each cell has the potential to acquire either a non-vulval fate (a 3° fate) or one of two vulval cell fates (a 1° or 2° fate). The fate, 1°, 2° or 3° is expressed by the number of divisions the cells undergoes and the axis of the divisions. The fate of the VPCs is influenced by cell-cell signalling — signaling between neighboring VPCs, from the gonadal anchor cell (AC), and from the hypodermis. Vulval development was one of the first areas to which considerable effort was applied to achieve a molecular understanding of how cells acquire their particular fates. The system, though limited in cell number, provides sufficient complexity to serve as a test case for our studies [16].

The VPC system has been one of the motivations for developing the current work, after it has been modeled in a relatively detailed manner in [10,5,7]. While there are many advantages in modeling efforts such as those mentioned, in terms of the insights that are gained, one of the remaining challenges is to integrate effects of different genetic components.

As part of our initial effort to test our "don't care" modeling approach, we have encoded the results of a small subset of the experimental results on VPC fate specification as reported in one of the key publications [18] on this topic. Our output variables are of the form $y_i \in \{1, 2, 3\}$, we have six such output variable corresponding to the fates of each of the VPCs. An experiment consists of recording the results of the pattern of fate specification among the 6 VPCs after perturbations such as genetic mutations or cell ablations. In our initial evaluation we used 8 input variables corresponding to gonad ablation ($x_0 = 1$ if gonad ablated, $x_0 = 0$ if gonad intact), and the mutations *lin-12(0), lin-12(d), lin-15, lin-7, lin-3, lin-2* and *lin-10*, measured by input variables $x_1, x_2, \cdots x_7$ respectively. We have entered experimental data from [18] about set-ups when only one of the input variables was perturbed, and then using our basic algorithm implementation solved for the maximal number of dc variables that can be set to 1 and found a maximally predictive model. We then compared the predictions of the model for experiments involving perturbations to 2 or 3 of the input variables with the actual data reported in [18]. The initial results, which seem encouraging in terms of predictive capabilities and runtime performance, should be interpreted very carefully, due to the limited size of the dataset. We are in the process of evaluating the results taking into account more experiments from [18], and also experiments reported in [17,15].

6 Related Work

How to form a general description of a class of objects given a set of examples is a basic problem in machine learning and has been studied in the artificial intelligence community [12,11]. This problem is termed *Generalization* or *Inductive Learning* and is viewed as a search through the hypothesis space. The general framework considers both positive and negative training examples, while our work currently is restricted to positive examples. Our method uses the 'technological' advances made in the formal methods community using tools like BDDs [1], QBF [14] and SAT solvers based on the DPLL method [3,2] to search the hypothesis space efficiently. The connection between machine learning and circuit design is explored in [8,4] demonstrating that logic-synthesis methods can be applied effectively to certain learning problems and can compete with standard machine learning programs.

Acknowledgements. We would like to thank Dennis Shasha for helpful discussions. This research was supported in part by NIH grant R24-GM066969.

References

1. R.E. Bryant. Graph-based algorithms for Boolean function manipulation. *IEEE Transactions on Computers*, C-35(12):1035–1044, 1986.
2. M. Davis, G. Logemann, and D. Loveland. A machine program for theorem-proving. *Comm. ACM*, 5(7):394–397, 1962.
3. M. Davis and H. Putnam. A Computing Procedure for Quantification Theory. *J. ACM*, 7(3):201–215, 1960.
4. C.M. Files and M.A. Perkowski. Multi-Valued Functional Decomposition as a Machine Learning Method. In *Proc. 28th IEEE International Symposium on Multiple-Valued Logic (ISMVL'98)*, pages 173–179, Fukuoka, Japan, May 1998. IEEE Computer Society.
5. J. Fisher, N. Piterman, E.J.A. Hubbard, M.J. Stern, and D. Harel. Computational Insights into *C. elegans* Vulval Development. *Proceedings of the National Academy of Sciences*, 102(6):1951–1956, 2005.
6. E. Giunchiglia, M. Narizzano, and A. Tacchella. Quantified boolean formulas satisfiability library (qbflib), 2001. `http://www.qbflib.org`.
7. C.A. Giurumescu, P.W. Sternberg, and A.R. Asthagiri. Intercellular coupling amplifies fate segregation during *Caenorhabditis elegans* vulval development. *Proceedings of the National Academy of Sciences*, 103(5):1331–1336, 2006.
8. J. A. Goldman and M. L. Axtell. On Using Logic Synthesis for Supervised Classification Learning. In *Proc. 7th Int. Conference on Tools with Artificial Intelligence (ICTAI95')*, pages 198–205. IEEE Computer Society, 1995.

9. L.S. Huand and P.W. Sternberg. Genetic dissection of developmental pathways. The *C. elegans* Research Community, ed. WormBook, 2006. `http://www.wormbook.org`.
10. N. Kam, D. Harel, H. Kugler, R. Marelly, A. Pnueli, E.J.A. Hubbard, and M.J. Stern. Formal Modeling of C. elegans Development: A Scenario-Based Approach. In Corrado Priami, editor, *Proc. Int. Workshop on Computational Methods in Systems Biology (CMSB 2003)*, volume 2602 of *Lect. Notes in Comp. Sci.*, pages 4–20. Springer-Verlag, 2003.
11. R. S. Michalski. A Theory and Methodology of Inductive Learning. *Artificial Intelligence*, 20(2):111–161, 1983.
12. T. M. Mitchell. Generalization as Search. *Artificial Intelligence*, 18(2):203–226, 1982.
13. A. Pnueli and E. Shahar. A platform for combining deductive with algorithmic verification. In R. Alur and T. Henzinger, editors, *R. Alur and T. Henzinger, editors,* Proc. 8th Intl. Conference on Computer Aided Verification (CAV'96)*, volume 1102 of* Lect. Notes in Comp. Sci.*, Springer-Verlag*, pages 184–195, 1996.
14. D. P. Ranjan, D. Tang, and S. Malik. A Comparative Study of 2QBF Algorithms. In *Proc. 7th Int. Conference on Theory and Applications of Satisfiability Testing*, 2004.
15. P.W. Sternberg. Lateral inhibition during vulval induction in *Caenorhabditis elegans*. *Nature*, 335:551–554, 1989.
16. P.W. Sternberg. Vulval development. The *C. elegans* Research Community, ed. WormBook, 2005. `http://www.wormbook.org`.
17. P.W. Sternberg and H.R. Horvitz. Pattern formation during vulval development in *C. elegans*. *Cell*, 44:761–772, 1986.
18. P.W. Sternberg and H.R. Horvitz. The combined action of two intercellular signaling pathways specifies three cell fates during vulval induction in *C. elegans*. *Cell*, 58:679–693, 1989.
19. The *C. elegans* Research Community, ed. WormBook, 2006. `www.wormbook.org`.
20. L. Zhang and S. Malik. Conflict Driven Learning in a Quantified Boolean Satisfiability Solver. In *Proc. of the 2002 IEEE/ACM International Conference on Computer-aided Design (ICCAD'02)*, pages 442–449. ACM, November 2002.