Algorithms for Choosing Differential Gene Expression Experiments

Richard M. Karp* Roland Stoughton† Ka Yee Yeung‡

Abstract

Understanding biological systems at the level of genes and proteins is a major challenge. In this paper we represent the interactions among external environmental inputs and genes in a biological system with a graph-theoretic model called a *biological pathway*. Our goal is to verify a proposed biological pathway by observing the mRNA levels in the associated biological system under changing external environmental inputs and internal gene perturbations.

DNA microarrays allow large-scale comparisons of mRNA levels in pairs of cell cultures. A DNA microarray contains thousands of spots, each containing some portion of a gene. In a *differential test*, mRNAs from two different cell cultures are reverse transcribed to cDNAs, labeled with fluorescent dyes of two different colors, and applied to the array. The relative hybridization levels of the two cDNAs determine the colors of the spots. These colors form the differential gene expression data. In this paper we assume that biological pathways can be represented as boolean circuits without feedback, differential tests can be modeled as perturbations of the external inputs and genes, and different classes of genes controlled by the pathway can be associated with different outputs of the boolean circuit. A biological pathway can be verified by applying a set of differential tests and comparing the outcomes of the hybridization experiments with the predicted outputs of the pathway. Thus, selecting an economical set of differential tests to distinguish all the outputs is essential to the verification of biological pathways. In this paper we give an algorithm to construct such a set of tests. We have applied the algorithm to a model of the mating pathway in yeast.

*Supported in part by NSF Grant DBI-9601046. Address: Department of Computer Science and Engineering, University of Washington, Seattle, Washington 98195-2350. E-mail: karp@cs.washington.edu.

†Address: Rosetta Inpharmatics, 12040 115th Ave NE, Kirkland, Washington 98034. E-mail: rstoughton@rosetta.org.

‡Supported in part by Rosetta Inpharmatics and NSF Grant DBI-9601046. Address: Department of Computer Science and Engineering, University of Washington, Seattle, Washington 98195-2350. E-mail: kayee@cs.washington.edu.

1 Introduction and Motivation

A biological system consists of complicated interactions among external environmental inputs, genes and proteins. In this paper, we represent such a system by a graph-theoretic model called a *biological pathway*. Such a model captures the connectivity information and the interactions among the components of the system. A biological pathway can be inferred by observing the responses of a biological system due to changing external environmental inputs and internal gene perturbations.

In order to infer a biological pathway, the genes (the components) and their roles in the system have to be identified. Genes are converted into messenger RNAs (mRNAs) by the process of transcription. Messenger RNAs are in turn translated into proteins, which determine the functions of a cell. Hence, the distribution of mRNAs present in a cell provides a clue to its biological functioning. Comparative hybridization experiments compare mRNA levels in two cell cultures, and DNA microarrays allow comparative hybridization experiments to be carried out on a large scale.

After mRNAs are purified to be used in comparative hybridization experiments, they are reverse-transcribed to a more stable DNA form called complementary DNAs (cDNAs). The cDNA samples from each cell culture are labeled with fluorescent dyes with different emission wavelengths, such as Cy3 and Cy5 which have emission wavelengths corresponding to the colors green and red respectively, so that the relative abundance of mRNAs in the two cell cultures can be determined. The two fluorescently labeled cDNA samples are hybridized to a DNA microarray with thousands of spots, each of which contains a different DNA sequence. If a cDNA sequence is complementary to the DNA sequence on a particular spot, the cDNA sequence will hybridize to the spot, and the intensity of hybridization can be detected by fluorescence [3]. The relative fluorescent intensities of the two cell cultures on the thousands of spots on a DNA microarray give rise to extensive differential gene expression data that has to be processed. In the case of yeast, the complete genome has been sequenced, and the Brown Lab [4] has successfully arrayed all the known genes (approximately 6200) of yeast.

We are interested in the following scenario. A biologist has provided a description of a biological system. The relevant variables in this system are external inputs such as drugs, nutrients or metabolites, and the expression levels of genes within the system itself. Each of these variables is two-valued: an input may be present or absent and a gene may be either expressed or not expressed. The relationships

among these variables are described by a boolean circuit[1]. It is also assumed that the levels of many genes outside the pathway are regulated by the variables in the pathway, and that these genes fall into distinct coregulated sets associated with different output signals from the circuit. Each of these coregulated sets of genes is called an *output class*. Our goal is to devise a set of tests that can be applied to the circuit in order to determine, among a large set of genes, which ones fall into each output class. If these tests determine that the measured expression levels of many genes are consistent with their membership in particular output classes, then we have evidence that the boolean circuit model describes the biological system correctly.

Consider a fictitious biological pathway in which a protein P is manufactured by a gene G. A drug D can be applied to form a complex with protein P. Three types of genes are regulated by this complex: those that are expressed when drug D is present, those that are expressed when protein P is present and drug D is absent, and those that are expressed when protein P and drug D are both present. This biological pathway is shown in Figure 1, and its representation as a boolean circuit is shown in Figure 2. The three output lines of the circuit correspond to the three classes of genes that are regulated by the pathway.

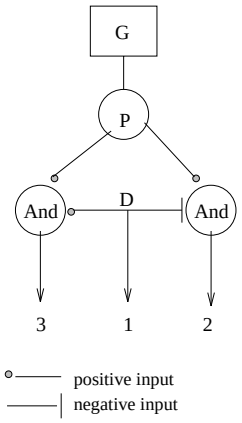

Figure 1: An example of a biological pathway.

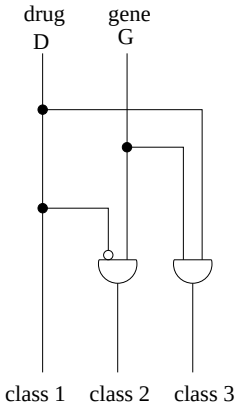

Figure 2: A biological pathway represented as a boolean circuit.

[1]A boolean circuit model of biological pathways has been considered in [1].

All inputs in a biological pathway have default values. In the case of Figure 2, the default value of drug D is 0 which means that drug D is not applied, and the default value of gene G is 1 which means that gene G is not knocked out. The value of a perturbed input is the complement of its default value. When a cell culture is prepared for a hybridization experiment, one or more of the inputs may differ from their default values, and the actual values of the inputs constitute the *condition* of the cell culture. The condition in which all external inputs are at their default values and no genes are knocked out is called the *wild type*. A *differential test* is a pair of conditions. The mRNA levels in the hybridization experiments are discretized to either low or high. The differential expression data from the microarray are the colors of each spot on the DNA microarray. When the cell culture that is labeled with the green fluorescent dye has higher mRNA level than the cell culture labeled with the red fluorescent dye, the spot on the DNA microarray appears green on stimulation by a laser. Similarly, a spot on the DNA microarray will appear red if the cell culture labeled red has higher mRNA level. When both cell cultures have similar mRNA levels, the spot on the DNA array appears yellow on stimulation by a laser. Two genes belong to the same output class if their outcomes under differential tests are either always the same or always different. We assume that output classes represent disjoint sets of genes. We say that a set of output classes are *distinguished* if they respond differently with respect to a set of differential tests. Therefore, a proposed biological pathway can be verified by applying a set of differential tests, and comparing the outcomes of the hybridization experiments with the predicted outputs of the pathway. Hence, selecting an economical set of differential tests that can distinguish all output classes is essential to the verification of biological pathways.

In this paper, we assume biological pathways can be represented as boolean circuits without feedback. We also assume that genes regulated by the pathway fall into disjoint output classes. The goal is to determine an economical set of differential tests that can distinguish all the output classes.

2 The Problem Statement

A boolean circuit $B = (\mathcal{I}, \mathcal{O}, f)$ which represents a biological pathway is given, where $\mathcal{I}$ is the set of inputs, $\mathcal{O}$ is the set of output classes, and f is the function that maps conditions to output values (which are either 0 or 1) in the circuit. In addition to the external inputs, $\mathcal{I}$ includes an additional input, called *mutation*, for each gene in the pathway, indicating whether the gene is artificially mutated. At this level of discussion, we suppress the details of the boolean circuit, *i.e.*, how the gates of the circuit are interconnected, and how the output classes are associated with the wires in the circuit.

Our problem instance can be represented by $\mathcal{P} = (B, M, \rho, W)$ where B is a boolean circuit, M is the maximum number of input perturbations in a condition, ρ maps an input perturbation to the cost of obtaining that perturbation, and W is the set of default values of all the inputs in $\mathcal{I}$, *i.e.*, the wild type condition.

Our goal is to select an economical set of differential tests sufficient to distinguish all pairs of output classes. We attack the problem in two stages. First, we select a minimum-cost set of conditions sufficient for the creation of the necessary differential tests. Then, given these conditions, we minimize the number of differential tests. Thus, our primary emphasis is on the cost of the conditions, with the actual number of

tests playing a secondary role. This is reasonable, since growing a culture of perturbed cells is significantly more expensive than performing an array experiment.

We now define the properties required for a set of differential tests.

Definition 1 *A condition is a set of inputs, denoted by* $c = (I_{i_1}, I_{i_2}, \ldots, I_{i_m})$ *where* I_{i_j} *is an input perturbed in the boolean circuit,* $I_{i_j} \in \mathcal{I}$ *and* $0 \leq m \leq M$.

Definition 2 *Let* $O_p \in \mathcal{O}$ *be an output class in a boolean circuit* B. *The output value at* O_p *under condition* c *is given by* $f_p(c)$.

Example 1 *The following table shows the output values of the boolean circuit in Figure 2 with default values of D and G being 0 and 1 respectively:*

	()	(D)	(G)	(D G)
o1	*0*	*1*	*0*	*1*
o2	*1*	*0*	*0*	*0*
o3	*0*	*1*	*0*	*0*

The rows in the above table are the output classes, and the columns are the conditions. For example, () is the wild type condition, and (D G) is the condition with both inputs D and G perturbed from their default values (*i.e.* D = 1 and G = 0). The entries in the table are the output values of the circuit, for instance, f_1(D G) is 1.

Definition 3 *A pair of output classes* O_p, O_q *is said to agree on condition* c *if* $f_p(c) = f_q(c)$. *Similarly, a pair of output classes* O_p, O_q *is said to disagree on condition* c *if* $f_p(c) \neq f_q(c)$

In Example 1, output classes o1 and o3 agree on condition (), and they disagree on condition (D G).

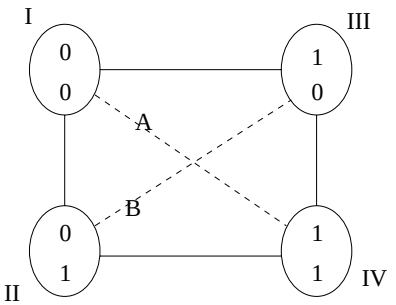

Figure 3: A Differential test diagram.

All the possible values of a pair of conditions can be represented graphically by a *differential test diagram* shown in Figure 3. The numbers inside the oval indicate the output values of a pair of output classes under a condition. For example, for a pair of output classes O_p, O_q, if both of the output values under a condition c are zero, *i.e.*, $f_p(c) = f_q(c) = 0$, then condition c is in category I. Categories I and IV correspond to agreement, while categories II and III correspond to disagreement. In Example 1, for output classes o1 and o2, condition (G) belongs to category I, condition () belongs to category II, and conditions (D) and (D G) belong to category III.

Definition 4 *A differential test* T *is denoted by* $\{c_i \; c_j\}$ *where* c_i *and* c_j *are conditions. A differential test* T *is said to be a distinguishing test for a pair of output classes* O_p, O_q *if one of the following is satisfied:*

- O_p, O_q *agree on condition* c_i *and* O_p, O_q *disagree on condition* c_j, **or**
- O_p, O_q *disagree on condition* c_i *and* O_p, O_q *agree on condition* c_j

A distinguishing test corresponds to a solid line in Figure 3. In Example 1, {() (G)} is a distinguishing test for output classes o1 and o2. Suppose condition () is labeled with the green fluorescent dye, and condition (G) is labeled with the red dye in the comparative hybridization experiments. Spots on the DNA array corresponding to output class o1 will appear yellow while spots corresponding to output class o2 will appear green.

Definition 5 *A pair of differential tests* $\{T_1, T_2\}$ *is said to be a cross test if they correspond to the pair of dotted lines in a differential test diagram.*

Conditions in categories I and IV (dotted line A in Figure 3) cannot distinguish a pair of output classes because both output classes show the same response to the conditions. Interestingly, conditions in categories II and III (dotted line B in Figure 3) also cannot distinguish a pair of output classses since the outcomes are always different and the two spots may belong to the same output class. However, a cross test, which is a pair of differential tests under both dotted lines, can distinguish a pair of output classes. In terms of the comparative hybridization experiments, the spots on the DNA microarray corresponding to one output class will show the same color on the pair of differential tests comprising the cross test, while the spots corresponding to another output class will show different colors on the two tests. Thus, a pair of output classes can be distinguished by a cross test since it is established that the outcomes for the two output classes are neither always the same nor always different.

Definition 6 *A set of differential tests* $\mathcal{T}$ *is said to distinguish a pair of output classes* O_p, O_q *if there is at least a distinguishing test or a cross test in* $\mathcal{T}$ *for output classes* O_p, O_q.

In general, a set of differential tests $\mathcal{T}$ *is said to distinguish a pair of output classes* O_p, O_q k *times if there are a total of* k *distinguishing tests and cross tests for output classes* O_p, O_q *in* $\mathcal{T}$.

Definition 7 *A condition cover is a set of conditions* $C = (c_1, c_2, \ldots, c_r)$ *such that for each pair of output classes, there is at least one condition for which they agree and at least one condition for which they disagree.*

In general, a k*-condition-cover is a set of conditions such that for each pair of output classes, there are at least* k *conditions for which they agree and at least* k *conditions for which they disagree.*

Definition 8 *A set of differential tests that can distinguish all pairs of output classes is called a test cover.*

In general, a set of differential tests that distinguishes each pair of output classes at least k *times is called a* k*-test-cover.*

2.1 The Cost Model

The cost function $\rho(I_i)$ in the problem instance $\mathcal{P}$ specifies the cost of perturbing input I_i from its default value. We assume that costs are additive. In addition, a cost tree model is assumed. A cost tree model allows us to compute

the cost of a set of conditions by determining the minimum cost of all the input perturbations in the set of conditions. Suppose the cost of a set of conditions $C = (c_1, c_2, \ldots, c_r)$ is to be determined. First of all, the conditions are sorted in increasing number of inputs perturbed. The wild type is assumed to have cost 0 and is the root of the cost tree. Conditions are added to a node in the existing cost tree in the order of increasing number of inputs perturbed such that the additional cost of adding each condition to the cost tree is minimum.

The above idea is illustrated with conditions C = {(), (D), (G), (D G)} in Example 1. Suppose the cost of perturbing input D and G are $\rho(D) = 2$ and $\rho(G) = 6$ respectively. First of all, conditions (D) and (G) with only one input perturbed are added to the cost tree with the wild type as the root. The cost of adding condition (D) to the wild type is 2 units, and the cost of adding condition (G) to the wild type is 6 units. Then, condition (D G) with two inputs perturbed is added to the cost tree. Condition (D G) will be added to condition (G) because it will yield the minimum additional cost of 2 units. The total cost of C is the total cost of all input perturbations, which is 2 + 6 + 2 = 10 units as shown in Figure 4.

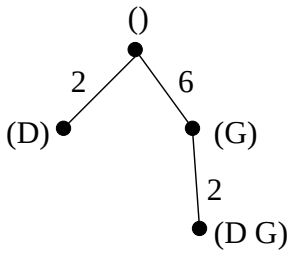

Figure 4: Cost tree for Example 1.

We assume that the cost of a test cover is the cost of the set of conditions comprising the test cover. This is because we assume that the cost of a test cover lies in obtaining the conditions, not in comparing the results of the two conditions. Furthermore, we assume the wild type condition is always included in the condition cover.

We believe that our problem is NP-hard, and the goal of our project is to develop algorithms which can efficiently generate a k-test-cover with minimum cost such that all output classes are distinguished, where $k \geq 1$.

3 The Naive Approach

One obvious approach is to enumerate all possible differential tests. Then, pairs of output classes that each distinguishing test and each cross test can distinguish are determined. Finally, a test cover with minimum cost is determined.

However, the enumeration approach is not very efficient. Suppose we have 15 inputs, and we can perturb at most 2 inputs at a time. The total number of possible conditions is $\binom{15}{0} + \binom{15}{1} + \binom{15}{2} = 121$. Hence, the number of possible distinguishing tests that have to be considered is $\binom{121}{2} = 7260$. Under the naive enumeration approach, we have to enumerate 7260 possible distinguishing tests, which is not very efficient.

4 Our Approach

Our approach is to divide the problem into two subproblems. The first subproblem is to determine a condition cover with minimum cost. The second subproblem is to determine a test cover with the minimum number of differential tests from the condition cover computed in the first subproblem. Since the condition cover consists of relatively few conditions compared to the set of all possible conditions, the expensive step of enumerating a huge number of differential tests is avoided.

4.1 Subproblem 1: Condition Cover

The first subproblem is to find a condition cover C^* with minimum cost. Our approach is illustrated in Example 2.

Example 2 *Suppose a 1-condition-cover is to be computed for a boolean circuit with the following table of output values.*

	W	A	B	C	D	E
o1	*0*	*1*	*0*	*1*	*0*	*1*
o2	*1*	*0*	*1*	*1*	*0*	*1*
o3	*0*	*1*	*1*	*1*	*1*	*0*
o4	*1*	*1*	*0*	*1*	*0*	*1*

In Example 2, W, A, B, C, D, and E are conditions while o1, o2, o3, and o4 are output classes. Furthermore, W is the wild type condition. Let

$$x_\alpha = \begin{cases} 1 & \text{if condition } \alpha \text{ is in the condition cover } C^* \\ 0 & \text{otherwise} \end{cases}$$

where α = W, A, B, C, D or E. Consider output classes o1 and o2, by Definition 7 there must be a condition in the condition cover for which they disagree. The idea is illustrated by the following constraint inequality:

$$x_W + x_A + x_B \geq 1$$

Moreover, at least one condition in the condition cover has to agree on output classes o1 and o2:

$$x_C + x_D + x_E \geq 1$$

Using this approach, two constraint inequalities can be obtained for each pair of output classes. The *branch and bound* approach [6] can be used to solve for a condition cover with minimum cost. The search tree is built in a depth-first manner by adding variables to the current path. Initially, the search tree consists of the root only. Then, the constraint inequality with the least number of variables which is not yet satisfied is considered. The variable in the inequality which gives the lowest cost when added to the current path will be added to the current path. This process is continued until all the constraint inequalities are satisfied. Conditions on the path from the root to a leaf form a condition cover. The current solution is the condition cover so far obtained from the search tree with the minimum cost. Then, backtracking occurs at the most recent constraint inequality where there is a choice of variables. If the cost of adding the current condition to the path exceeds that of the best solution so far, the subtree is pruned.

Example 2 is used to illustrate the branch and bound approach. The set of inequality constraints in order of increasing number of variables is as follows:

$$x_W \geq 1$$

$$x_B + x_C \geq 1$$

$$x_A + x_B \geq 1$$

$$x_A + x_C \geq 1$$

$$x_W + x_A + x_C \geq 1$$

$$x_C + x_D + x_E \geq 1$$

$$x_B + x_D + x_E \geq 1$$

$$x_W + x_A + x_C \geq 1$$

$$x_W + x_A + x_D + x_E \geq 1$$

$$x_W + x_C + x_D + x_E \geq 1$$

$$x_W + x_B + x_D + x_E \geq 1$$

$$x_A + x_B + x_C + x_D + x_E \geq 1$$

Our cost model assumes the wild type condition is always included in the condition cover. Therefore, condition W is the root of the search tree as shown in Figure 5. For simplicity, suppose the costs of adding conditions A, B, C, D, and E are constant and are 3, 1, 2, 8, and 2 units respectively (In our cost model, the additional cost of adding a condition to a set of conditions depends on the conditions already in the set.)

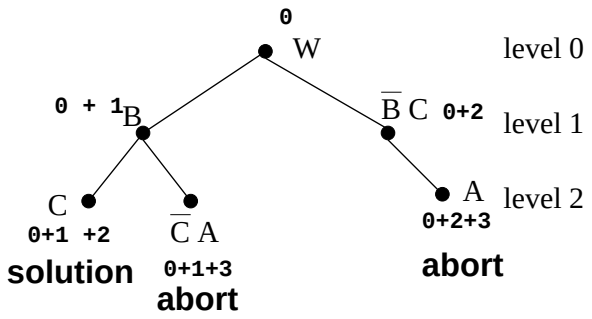

Figure 5: Search tree illustrating branch and bound.

The cost of a path is the cost of the set of conditions represented by the path. For example, the cost of the path WB is 1 unit and the cost of the path WBC is 3 units. The bar above a condition in Figure 5 means that the particular condition is not chosen and will not be included in the subtree. For example, at level 1, when condition C is added to the search tree, paths containing condition B have already been explored, so the subtree rooted at $\overline{B}$C does not contain condition B. Therefore, there is no branch for condition B under node $\overline{B}$C at level 2. On path W$\overline{B}$$\overline{C}$A, the cost at node $\overline{C}$A is 4 units, which is greater than the cost of the current best solution WBC, so the search tree is pruned at node $\overline{C}$A. Similarly, the search tree is also pruned at node A on the path W$\overline{B}$CA. The conditions in the search tree are processed in the order of increasing costs, for example, at level 1, condition B is processed before condition C because conditions (W,B) have a lower cost than conditions (W,C). The heuristic of considering lower cost conditions first makes it more likely that the best solution will be discovered in the early stage of the search.

4.2 Subproblem 2: Test Cover

The second subproblem is to find a test cover $\mathcal{T}^*$ with the minimum number of differential tests from the condition cover C^* computed in the first subproblem. The idea is to enumerate all the possible differential tests from the condition cover C^*. Then, constraint inequalities are set up in a similar way as in the case of a condition cover. Let

$$y_\beta = \begin{cases} 1 & \text{if differential test } \beta \text{ is in the test cover } \mathcal{T}^* \\ 0 & \text{otherwise} \end{cases}$$

Example 3 *Consider a condition cover {W, R, S, T} with the following table of output values:*

	W	R	S	T
o1	*0*	*0*	*1*	*1*
o2	*1*	*1*	*1*	*1*
o3	*0*	*1*	*1*	*0*
o4	*0*	*1*	*1*	*1*
o5	*0*	*1*	*0*	*0*

In Example 3, the set of all possible differential tests from the given condition cover is {WR, WS, WT, RS, RT, ST}. Figure 6 is the differential test diagram for output classes o1 and o3. WR, WT, RS, and ST are distinguishing tests. WS and RT form a cross test which is represented in the constraint inequality by * which has the effect of "and".

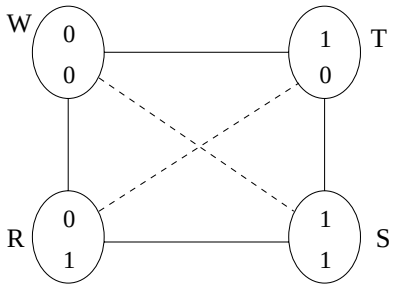

Figure 6: Differential test diagram for output classes o1 and o3.

The constraint inequality to distinguish output classes o1 and o3 is as follows:

$$y_{WR} + y_{WT} + y_{RS} + y_{ST} + y_{WS} * y_{RT} \geq 1$$

Similarly, a constraint inequality for each pair of output classes can be determined.

Solving the set of constraint inequalities by branch and bound with the cost of a test being 1 unit gives a 1-test-cover. In the case of a cross test, both conditions in the cross test have to be added to the path in the search tree.

4.3 k-Test-Cover

A k-test-cover does not necessarily arise from a k-condition-cover. For example, the minimum cost 1-condition-cover {W, B, C} in Example 2 contains a 2-test-cover {WB, WC, BC}. Therefore, our algorithm loops through all the l-condition-cover, where $l = 1, \ldots, k$, and determine if a k-test-cover exists. If a k-test-cover is found, the loop terminates. A l-condition-cover, where $1 \leq l \leq k$, can be determined by replacing all the right hand sides in the constraint inequalities with l instead of 1 in Section 4.1, and solving the set of constraint inequalities by branch and bound. Similarly, a k-test-cover can be obtained by replacing all the right hand sides in the constraint inequalities in Section 4.2 with k, which are then solved by branch and bound.

4.4 Test Cover as a Graph Theory Problem

The problem of finding a test cover from a given condition cover $C = \{c_1, c_2, \ldots, c_n\}$ can be modeled as a graph-theoretic problem. A graph G is defined with vertices being the conditions in the condition cover C, and edges being the differential tests. More specifically, let $G = (C, E)$ where $(c_i, c_j) \in E$ if conditions c_i and c_j form a differential test in a test cover, $1 \leq i, j \leq n$.

Definition 9 *Conditions* $P = c_{l_1}, \ldots, c_{l_h} \mid c_{l_{h+1}}, \ldots, c_{l_n}$ *form a partition for a pair of output classes* O_p, O_q *if one of the following is satisfied:*

- $f_p(c_{l_i}) = f_q(c_{l_i})$ *and* $f_p(c_{l_j}) \neq f_q(c_{l_j})$ *where* $i = 1, 2, \ldots, h$ *and* $j = h+1, h+2, \ldots, n$, **or**
- $f_p(c_{l_i}) \neq f_q(c_{l_i})$ *and* $f_p(c_{l_j}) = f_q(c_{l_j})$ *where* $i = 1, 2, \ldots, h$ *and* $j = h+1, h+2, \ldots, n$.

A single partition $c \mid (C \setminus c)$ *is a partition with exactly one condition* c *on one side of the partition.*

Consider the condition cover in Example 2, C = {W, B, C}. For output classes o1 and o2, C | WB forms a single partition because they agree on condition C and disagree on conditions W and B.

A partition can be defined for each pair of output classes. A distinguishing test is a differential test between conditions on different sides of a partition. For example, CW and CB are distinguishing tests in partition C | WB. Cross tests cannot be inferred from partitions. A 1-test-cover graph for Example 2 is shown in Figure 7.

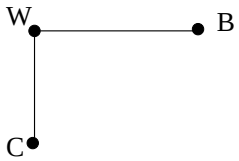

Figure 7: A 1-TEST-COVER GRAPH FOR EXAMPLE 2.

A *k-edge-connected graph* [2] can be used to infer a k-test-cover when all single partitions exist, where $k > 1$. The idea will be formalised in Theorem 1.

Definition 10 *A graph is k-edge-connected if it cannot be disconnected by the removal of fewer than k edges, where k > 1.*

A minimal k-edge-connected graph is a k-edge-connected graph with the least number of edges.

Theorem 1 *Given a condition cover, if all single partitions exist, then a minimal k-edge-connected graph gives a k-test-cover with the least number of tests, where k > 1.*

Proof:

Let $C = \{c_1, c_2, \ldots, c_n\}$ be a condition cover. From the definition of a k-edge-connected graph, every k-edge-connected graph gives a k-test-cover. Since all single partitions exist, *i.e.*, all $c_i \mid (C \setminus c_i)$ where $i = 1, 2, \ldots, n$ exist, a k-test-cover requires that each vertex c_i in G has degree at least k. Let $deg(c_i)$ denote the degree of vertex c_i. Summing up the degrees of all the vertices in the graph G, we get $\sum_{i=1}^{n} deg(c_i) \geq k * n$. Since $\sum_{i=1}^{n} deg(c_i) = 2 * |E|$ for any graph, the number of edges $|E|$ required for a k-test-cover is at least $\lceil 1/2 * k * n \rceil$. It is shown in [2] that there exists a k-edge-connected graph with $\lceil 1/2 * k * n \rceil$ edges. In fact, any k-edge-connected graph has at least $1/2 * k * n$ edges. Therefore, a minimal k-edge-connected graph gives a k-test-cover with the least number of tests. □

Theorem 1 fails for $k = 1$ because there does not exist a 1-cover graph with $\lceil 1/2 * 1 * n \rceil$ edges.

In the case of single partitions, cross tests cannot help to reduce the number of differential tests required for a k-test-cover when $k > 1$ because cross tests require at least two conditions on each side of the partition.

Corollary 1 *For* $k > 1$, *if all single partitions exist, the minimum k-test-cover can be computed without the branch and bound step.*

Proof: The proof follows from Theorem 1 since the k-edge-connected graph does not have to be computed from the branch and bound step. □

The significance of Theorem 1 is that determining whether all single partitions exist is a relatively inexpensive step compared to the branch and bound step. Theorem 1 and Corollary 1 apply only when k is strictly greater than 1. In the case of a 1-cover, the existence of all single partitions does *not* imply that a spanning tree gives a 1-test-cover with the minimum number of tests. Consider the following counter-example on condition cover {A, B, C, D} with partitions A | BCD, B | ACD, C | ABD, D | ABC, AB | CD, AC | BD and BC | AD. Moreover, partition AB | CD arises from the differential test diagram shown in Figure 8.

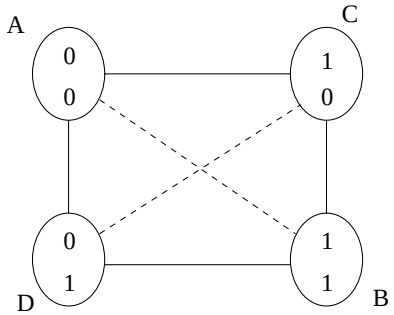

Figure 8: DIFFERENTIAL TEST DIAGRAM FOR PARTITION AB | CD.

From Figure 8, cross test {AB CD} distinguishes the pair of output classes giving the differential test diagram. Therefore, {AB, CD} is a minimum 1-test-cover in this example.

Hence, in the case of a 1-test-cover, even if all single partitions exist, the branch and bound step cannot be avoided.

5 A Restricted Problem

In case the biological pathway in our model is part of a larger biological system, mutations may have effects on other components in the biological system that are not represented in the model. These components may in turn affect other components in our model. Therefore, in this section, we consider the problem in which differential tests are restricted to a pair of conditions with the same mutations.

Our approach to solving the restricted problem is similar to the original problem in that the problem is divided into two subproblems: determining the condition cover, and then

the test cover from the chosen condition cover. However, there are some modifications to both steps so as to ensure that differential tests have the same mutations.

As in the original problem, the first subproblem is to determine a condition cover. In the following description, we assume that a 1-test-cover is to be determined. In order to determine a set of conditions with minimum cost that guarantees differential tests with the same mutations, our approach is to divide the condition cover problem into two stages. In stage A, a set of conditions, C_A, that has at least one condition that disagree for each pair of output classes is determined. In stage B, additional conditions are added to the condition cover obtained in stage A, C_A, such that for each pair of output classes, there is at least one differential test with the same mutation. The condition cover obtained in stage B is denoted as C_B. A test cover can be obtained by the same branch and bound method on the condition cover C_B as in the original problem. The only difference is that we restrict attention to differential tests in which both conditions have the same mutations.

Example 4 *Figure 9 is a table showing the values of output classes o1, o2, and o3 for conditions A to K. A 1-condition-cover is to be determined.*

In Example 4, a and c are external inputs, while b and d are mutations. The costs of perturbing inputs a, b, c, and d from their default values are 3, 8, 9, and 20 respectively.

In stage A, the set of conditions C_A that disagree for each pair of output classes is determined. In order to distinguish output classes o1 and o2, at least one condition in the condition cover has to have different output values for the pair of output classes. The idea is illustrated by the following constraint inequality:

$$x_A + x_B + x_D + x_E + x_F + x_G + x_H + x_K \geq 1$$

Similarly, in order to distinguish o1 and o3, we get,

$$x_A + x_B + x_E + x_F + x_H + x_I + x_K \geq 1$$

In order to distinguish o2 and o3, we get,

$$x_D + x_G + x_I \geq 1$$

The above set of inequalities can be solved by branch and bound as in Section 4.1. The following possible C_A's are obtained: {A, D} with a cost of 9, {A, G} with a cost of 12, and {A, I} with a cost of 17.

In stage B, for each of the possible C_A's in the order of increasing cost, a set of conditions which guarantees a 1-test-cover is determined. For C_A = {A, D}, since neither condition A (which is the wild type) nor condition D contain any mutation, we need at least one condition with no mutation such that output classes o1 and o2 agree. However, inspecting the output value table closely shows that there is no such condition. For the pairs o1 and o3, and o2 and o3, conditions A and D form a valid differential test, so no inequality is required. Since there is no valid test with C_A = {A, D} to distinguish output classes o1 and o2, there is no valid condition cover in stage B for C_A = {A, D}. Similarly, there is no valid condition cover in stage B for C_A = {A, G}.

Consider C_A = {A, I}, since condition A has no mutation, while condition I contain mutation b, there is no differential test already contained in C_A. In order to distinguish output classes o1 and o2, at least one condition is needed with the same mutation that forms a valid differential test with conditions A or I. As shown in Figure 10, condition F has the same mutation as condition I and it forms a distinguishing test with condition I. Hence, we have the following constraint inequality:

$$x_F \geq 1$$

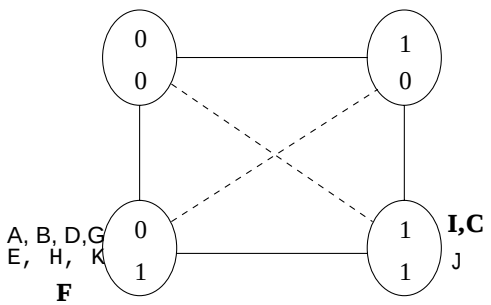

Figure 10: Differential Test Diagram for output classes o1 and o2 in Example 4.

Proceeding in this way, we obtain the following two constraint inequalities to distinguish o1 and o3, and o2 and o3 respectively:

$$x_C + x_D + x_G \geq 1$$

$$x_C + x_D + x_F + x_G \geq 1$$

Solving the above set of inequalities by branch and bound gives C_B = {A, I, F, C}. A 1-test-cover can then be computed by branch and bound on condition cover C_B, restricting to differential tests with the same mutations.

5.1 k-Test-covers

In the restricted problem, a k-test-cover may arise from a m-condition-cover from stage A, and a l-condition-cover from stage B, where $1 \leq m, l \leq k$. Therefore, our algorithm loops through both stage A and stage B of the condition cover step to compute a k-test-cover:

Algorithm:

- For $m = 1$ to k do
  - Compute C_A^m: a m-condition-cover from stage A.
  - For $l = m$ to k do
    - Compute C_B^l: a l-condition-cover from stage B with C_A^m.
    - Determine if a k-test-cover exists in C_B^l.
    - Terminate the algorithm if a k-test-cover is found.

5.2 Critique of the Approach

The approach of dividing the condition cover problem into stages A and B has the drawback that the cost of the condition cover found in stage A, C_A, may not reflect the cost of the condition cover in stage B, C_B. It is possible for the best condition cover C_B to have an expensive C_A. In the above example, it can be seen that only the most expensive C_A has a valid C_B. In our actual implementation, there is a

	() A	(a) B	(b) C	(c) D	(d) E	(a b) F	(a c) G	(a d) H	(b c) I	(b d) J	(c d) K
o1	*0*	*0*	*1*	*0*	*0*	*0*	*0*	*0*	*1*	*1*	*0*
o2	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*
o3	*1*	*1*	*1*	*0*	*1*	*1*	*0*	*1*	*0*	*1*	*1*

Figure 9: Output value table for Example 4.

time limit to each of the branch and bound step. The best solution C_B may not be computed if it has an expensive subset C_A.

An alternative approach is to determine a mutation cover, *i.e.*, a set of mutations that guarantees a k-test-cover, in the first subproblem. In the second subproblem, all possible differential tests in the mutation cover is enumerated, and a k-test-cover can be determined as in Section 4.2. The drawback of this approach is that the first subproblem minimizes only the cost of a mutation cover, and not the cost of all the conditions comprising a test cover. Another drawback is that the second subproblem may be computationally expensive if there are a large number of external inputs.

6 Implementational Details and Results

Algorithms for both the general problem and the restricted problem have been implemented in C. The current implementation accepts an input file, a time limit for the branch and bound step, and the number of test covers required as arguments. The input file contains a boolean circuit representation of the biological pathway to be tested. The boolean circuit representation specifies the input and output lines for each gate in the circuit. The current implementation allows the following gates with two inputs: AND, NAND, OR, NOR, EXCLUSIVE OR, and EXCLUSIVE NOR gates. It also allows the unary NOT gate and the identity gate. The input file also specifies the external inputs, mutations, output classes, the default values, the maximum number of input perturbations in a condition and the costs of perturbing external inputs and mutations. The implementation can be run in both interactive and batch modes.

We have run our implementation on the yeast mating pathway[2] shown in Figure 11. Yeast is well-suited for experiments because the entire genome is known and it has been arrayed [4]. Moreover, mutations in yeast are relatively easy to achieve.

There are four pathways in the yeast mating pathway that are of interest. The *kinase cascade* consists of output classes 3, 4, and 5. The *kinase cascade with second pathway* consists of output classes 3, 4, 5, and 6. The *coarse mating pathway* consists of output classes 1, 5 and 8. The *full pathway* consists of all the output classes.

In order to represent mutations in a boolean circuit, each of the mutable inputs z is replaced by an AND gate with z and the wild type of z, Wz, as inputs and the output is called Mz as shown in Figure 12. When Wz is 1, Mz gets the value of z, thus input z is not mutated. However, when Wz is 0, Mz becomes 0, which has the interpretation that gene z is knocked out in the actual experiments.

The results of running our implementation on the yeast mating pathway are shown in the next section. There are usually more than one condition cover and one test cover for

[2]We thank Matt Marton and Chris Roberts of Rosetta Inpharmatics for sharing their knowledge of the mating pathway in yeast.

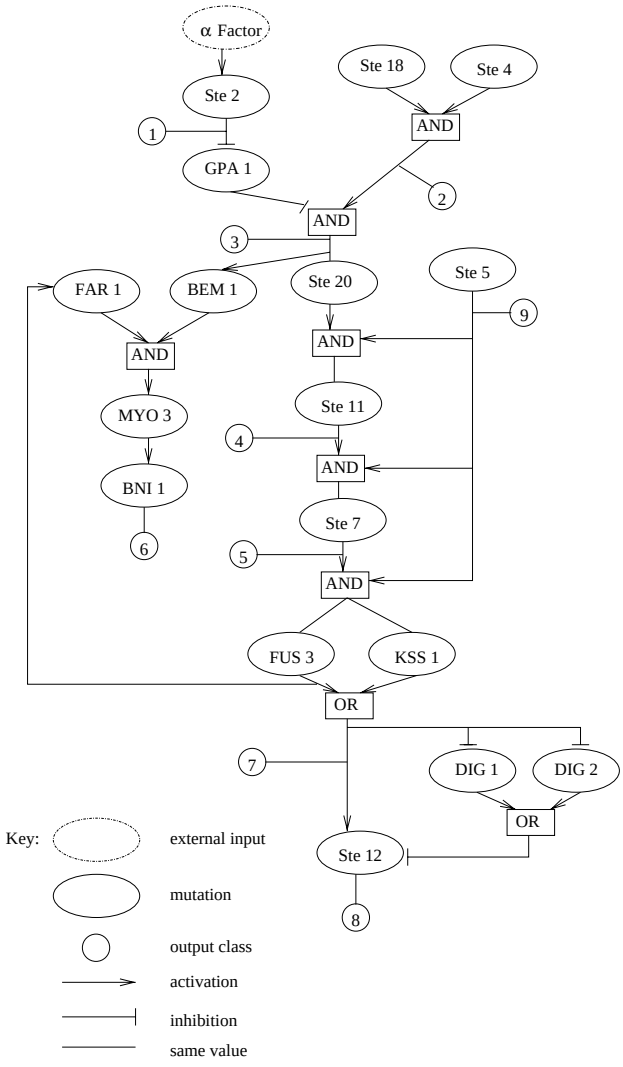

Figure 11: The Yeast Mating Pathway.

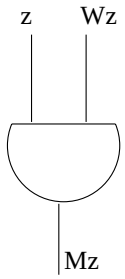

Figure 12: Boolean representation of mutations.

each of the four pathways, but only a condition cover and a test cover are shown.

6.1 Summarized Results of the General Problem

The results in Figure 13 are obtained with the maximum number of perturbations in any condition, M, equal to 3, and with a running time of two minutes on a Pentium 200. The same results are obtained with a higher time limit.

The 2-test-covers shown in Figure 13 arise from the 1-condition-covers shown.

6.2 Summarized Results of the Restricted Problem

The results in Figure 14 are obtained with the maximum number of inputs perturbed in any condition, M, equal to 3, and with an upper bound of three minutes for the branch and bound step on a Pentium 200.

Condition covers are not shown in Figure 14 because the 1-test-covers and 2-test-covers in the table arise from different condition covers.

It is interesting to note that there does not exist a test cover for the full pathway when a differential test consists of two conditions with the same mutation. This is because the values of output classes 2 and 9 cannot be changed by the external input α-factor.

A major drawback of our approach is that when there are a large number of solutions in stage A of the condition cover, there may not be enough memory to compute and store all the possible solutions of C_A. Hence, our implementation may not be able to compute test covers for certain problem instances. For example, our implementation fails to compute a 2-test-cover for output classes 1 to 8 when M is 4.

7 Conclusions and Future Work

Efficient algorithms have been developed and implemented to determine informative experiments that can classify data from competitive experiments into the output classes. As seen from our experimental results, the number of conditions in the condition cover is insignificant compared to the total number of possible conditions. Hence, our approach of dividing the problem into two subproblems is justified in the case of the yeast mating pathway.

In order to extend our work to other biological systems, it may be necessary to use a more general model. Shapiro *et. al.* [5] proposed a genetic circuit for the lysis-lysogeny decision in lambda phage. However, the circuit they proposed is a sequential logic circuit, and our approach does not handle circuits with feedback. One potential direction of future work is to extend our work to handle sequential circuits. Davidson *et. al.* [7] proposed a computational network model for the Endo16 cis-regulatory system in sea urchins. Their model is beyond the scope of our methods, since it is time-dependent and requires signal values that are not boolean (although they are drawn from a small discrete set). Therefore, another possible direction of future work is to extend our work to handle circuits that are multi-valued and time dependent.

References

[1] S. Akutsu, T. Kuhara, O. Maruyama, S. Miyano. *Identification of gene regulatory networks by strategic gene disruptions and gene over-expressions.* Annual ACM-SIAM Symposium on Discrete Algorithms, 1998.

[2] J. A. Bondy, U. S. R. Murty. *Graph Theory with Applications.* London : Macmillan, 1976.

[3] J. D. Buhler. *Anatomy of a Comparative Gene Expression Study.* http://www.cs.washington.edu/homes/jbuhler/research/array

[4] J. L. DeRisi, V. R. Iyer, P. O. Brown. *Exploring the Metabolic and Genetic Control of Gene Expression on a Genomic Scale.* Science, vol. 278, pp. 680-686, 1997.

[5] H. H. McAdams, L. Shapiro. *Circuit Simulation of Genetic Networks.* Science, vol 269, August 1995.

[6] G.L. Nemhauser and L.A. Wolsey. *Integer and Combinatorial Optimization.* Wiley (1988) pp. 349-367.

[7] C. H. Yuh, H. Bolouri, E. H. Davidson. *Genomic Cis-Regulatory Logic: Experimental and Computational Analysis of a Sea Urchin Gene.* Science, vol 279, March 1998.

	1-condition-cover	*1-test-cover*	*2-test-cover*
Kinase Cascade	() (WSte7) (Ste5 WSte7)	any spanning tree on the condition cover	any cycle on the condition cover
with Second Pathway	() (WBNI1) (WSte7 WBNI1) (Ste5 WSte7 WBNI1)	{(WBNI1) (Ste5 WSte7 WBNI1)} {() (WSte7 WBNI1)}	any cycle on the condition cover
Coarse Mating Pathway	() (WSte12) (WSte7 WSte12)	any spanning tree on the condition cover	any cycle on the condition cover
Full Pathway	() (WSte12) (WSte7), (WFus3) (WFus3 WKSS1), (WSte2 WFus3 WKSS1), (Ste4 WFus3 WKSS1), (Ste5 WFus3 WKSS1)	{(Ste5 WSte7) (Ste4 WFus3 WKSS1)}, {(WFus3 WKSS1) (WSte2 WFus3 WKSS1)}, {(WSte7) (WFus3)}, {(WSte12) (Ste5 WFus3 WKSS1)}	{(Ste5 WFus3 WKSS1) (Ste4 WFus3 WKSS1)}, {(WSte2 WFus3 WKSS1) (Ste4 WFus3 WKSS1)}, {(WFus3) (WSte2 WFus3 WKSS1)}, {(WSte7) (Ste5 WFus3 WKSS1)}, {(WSte12) (WSte7)}, {(WFus3 WKSS1) (WSte12)}, {(WFus3 WKSS1) (WFus3)}

Figure 13: Summarized results of the general problem.

	1-test-cover	*2-test-cover*
Kinase Cascade	{(Ste5) (aF Ste5)} {(WSte7) (aF WSte7)}	{(Ste5) (aF Ste5)} {(Ste5 WSte12) (aF Ste5 WSte12)} {(WSte7) (aF WSte7)} {(WSte7 WSte12) (aF WSte7 WSte12)}
with Second Pathway	{(Ste5 WSte7) (aF Ste5 WSte7)} {(WSte7) (aF WSte7)} {(WBNI1) (aF WBNI1)}	{(Ste5) (aF Ste5)} {(Ste5 WSte12) (aF Ste5 WSte12)} {(WSte7) (aF WSte7)} {(WSte7 WSte12) (aF WSte7 WSte12)} {(WBNI1) (aF WBNI1)} {(WSte12 WBNI1) (aF WSte12 WBNI1)}
Coarse Mating Pathway	{(WSte7) (aF WSte7)} {(WSte12) (aF WSte12)}	{(WSte7) (aF WSte7)} {(WSte7 WSte12) (aF WSte7 WSte12)} {(WSte12) (aF WSte12)} {(WSte12 WBNI1) (aF WSte12 WBNI1)}
Full Pathway	cannot be distinguished	cannot be distinguished

Figure 14: Summarized results of the restricted problem.