OPEN ACCESS Freely available online

PLoS one

Optimal *In Silico* Target Gene Deletion through Nonlinear Programming for Genetic Engineering

Chung-Chien Hong[¤], **Mingzhou Song***

Department of Computer Science, New Mexico State University, Las Cruces, New Mexico, United States of America

Abstract

Background: Optimal selection of multiple regulatory genes, known as targets, for deletion to enhance or suppress the activities of downstream genes or metabolites is an important problem in genetic engineering. Such problems become more feasible to address *in silico* due to the availability of more realistic dynamical system models of gene regulatory and metabolic networks. The goal of the computational problem is to search for a subset of genes to knock out so that the activity of a downstream gene or a metabolite is optimized.

Methodology/Principal Findings: Based on discrete dynamical system modeling of gene regulatory networks, an integer programming problem is formulated for the optimal *in silico* target gene deletion problem. In the first result, the integer programming problem is proved to be *NP*-hard and equivalent to a nonlinear programming problem. In the second result, a heuristic algorithm, called GKONP, is designed to approximate the optimal solution, involving an approach to prune insignificant terms in the objective function, and the parallel differential evolution algorithm. In the third result, the effectiveness of the GKONP algorithm is demonstrated by applying it to a discrete dynamical system model of the yeast pheromone pathways. The empirical accuracy and time efficiency are assessed in comparison to an optimal, but exhaustive search strategy.

Significance: Although the *in silico* target gene deletion problem has enormous potential applications in genetic engineering, one must overcome the computational challenge due to its *NP*-hardness. The presented solution, which has been demonstrated to approximate the optimal solution in a practical amount of time, is among the few that address the computational challenge. In the experiment on the yeast pheromone pathways, the identified best subset of genes for deletion showed advantage over genes that were selected empirically. Once validated *in vivo*, the optimal target genes are expected to achieve higher genetic engineering effectiveness than a trial-and-error procedure.

Citation: Hong C-C, Song M (2010) Optimal *In Silico* Target Gene Deletion through Nonlinear Programming for Genetic Engineering. PLoS ONE 5(2): e9331. doi:10.1371/journal.pone.0009331

Editor: Diego Di Bernardo, Fondazione Telethon, Italy

Received July 10, 2009; **Accepted** September 17, 2009; **Published** February 24, 2010

Funding: The authors were supported in part by the National Cancer Institute grant number 5U54CA132383 and the funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Competing Interests: The authors have declared that no competing interests exist.

* E-mail: joemsong@cs.nmsu.edu

¤ Current address: Department of Statistics, National Cheng-Kung University, Tainan City, Taiwan

Introduction

Selecting *in silico*, in a dynamic model of gene regulatory and metabolic networks, the right target genes for deletion so as to modify phenotypes can substantially expedite and lower the cost of genetic engineering. The target genes for deletion typically play key regulatory roles in the expression of downstream genes or metabolites to alter a phenotype to desirable states. The applications of genetic engineering are enormous. By genetically engineering plants to contain high levels of cellulose and hemicellulose [1], one may absorb the prohibitive cost of cellulose pretreatment before biomass-to-biofuel conversion. The brain tumor therapy using genetically engineered brain cells has eradicated tumors completely and affects tumor regression [2]. Current *in vivo* genetic engineering is often by trial-and-error, and unavoidably slow and sub-optimal. The few extant *in silico* genetic engineering strategies are seriously hampered by the scarcity of realistic dynamic models of gene regulatory and metabolic networks. However, we anticipate a closing gap between *in vivo* and *in silico* genetic engineering as realistic computational models of networks are made increasingly available by powerful data-driven network reconstruction software from high-throughput systems biology experiments.

Recent work by Deutscher et al. [3] and Nakae et al. [4] provides multiple gene knockout solutions to optimize the concentrations of designated metabolites in static models of metabolic networks. Our work extends to dynamic models, searching the target genes *in silico* from any subset of genes in a gene regulatory network (GRN) for deletion to maximize the concentration of a downstream gene. Using the probabilistic Boolean network model, Faryabi et al. [5] pose an integer programming problem to maximize the benefit of a cancer patient from the treatment which intervenes the activity of a gene over time. The problem is solved using dynamic programming in optimizing a downstream gene by turning on or off only a single target gene. Based on flux balance analysis, Alper et al. [6] and Jin et al. [7] formulate a linear programming problem, to modify the metabolic pathways in wild type *E. coli*. They introduce the method of minimization of metabolic adjustment to revise the objective function

to be quadratic for mutants. Both an exhaustive search and a greedy algorithm have been employed to optimize the yield of lycopene synthesis in the metabolic network by overexpressing or deleting three genes. They show that deleting three genes improves the phenotype of interest more effectively than deleting a single gene.

Motivated by the three-gene-deletion advantage, we study the more general multiple gene knockout (GKO) problem. Although we call all variables gene in our terminology, a variable can represent the concentration of a protein, an mRNA, or a metabolite. We use the discrete dynamical system (DDS) model to represent GRNs [8–13]. DDS models can be reconstructed from observed trajectories through data-driven methods [14–16], some of which can run on parallel supercomputers such as [13]. A nonlinear integer programming problem is formulated to define the GKO problem. We prove the nonlinear integer programming problem to be *NP*-hard. To approach efficiently the global maximum of the nonlinear integer programming problem with a generally non-concave objective function, we transform it to a nonlinear programming problem with fewer decision variables. We offer an algorithm called GKONP to solve the nonlinear programming problem. GKONP prunes insignificant terms in the objective function and takes advantage of the differential evolution algorithm, a parallel global optimization method. We use both the yeast pheromone pathway model and simulated models to demonstrate the performance of the GKONP algorithm.

Methods

Mathematical Formulation of the GKO Problem

We introduce the DDS model and formulate a nonlinear integer programming problem to search the optimal regulatory target genes for deletion. Here, we give the problem definition and notations.

The DDS Model. We use the DDS model [13] to represent dynamical interactions in GRNs. DDS modeling is data-driven and has been used for characterizing the cell cycle network [9]. The model assumes that the change rate of each gene at the current time point is a linear combination of concentrations of genes at the previous time point. Thus state transitions are independent of each other. Let N be the number of genes. Let t be the discrete time starting from 0. Let h denote the actual time between two consecutive discrete time points. Let $g_i[t]$ be the concentration of gene i at time t. Let $\mathbf{g}[t]=(g_1[t], g_2[t], \ldots, g_N[t])^T$ be a state vector of concentrations of all genes at time t. Then, the 1st-order linear DDS model is defined by

$$\frac{\mathbf{g}[t]-\mathbf{g}[t-1]}{h}=Q\mathbf{g}[t-1], \text{ for all positive integer } t, \quad (1)$$

where Q is an $N\times N$ regulation matrix, epitomizing a GRN. Q can be estimated with experimental data from wild type under normal and perturbed conditions. Letting $A=hQ+I$, we have

$$\mathbf{g}[t]=A\mathbf{g}[t-1]. \quad (2)$$

We call A the system matrix. Evidently the solution to the DDS model is

$$\mathbf{g}[t]=A^t\mathbf{g}[0]. \quad (3)$$

Let a_{ij} be the entry at row i and column j of matrix A. a_{ij} is zero if gene j is not a parent (regulator) of gene i. Matrix A is sparse when the number of parents of each gene is small.

Optimal Target Gene Deletion through Nonlinear Integer Programming. Based on the DDS model, a nonlinear integer programming is formulated to maximize a downstream gene by searching regulatory target genes for deletion. We define the binary knockout vector $x=(x_1,\ldots,x_N)^T$. $x_i\in\{0,1\}$ is 1 if gene x_i is intact; x_i is 0 if gene i is deleted, equivalent to setting all entries on either row i or column i in system matrix A to zero. A GRN with knockout can be represented by a new system matrix

$$A(x)=\text{diag}(x)\ A\ \text{diag}(x)$$
$$=\begin{bmatrix} x_1 & 0 & \cdots & 0 \\ 0 & x_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & x_N \end{bmatrix} A \begin{bmatrix} x_1 & 0 & \cdots & 0 \\ 0 & x_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & x_N \end{bmatrix}.$$

Thus, the knockout DDS solution is $\mathbf{g}[t]=(\text{diag}(x)A\ \text{diag}(x))^t\mathbf{g}[0]$. Using the DDS solution, we define the GKO problem to maximize the objective function $f(x)$, denoting the concentration of gene z at time T, by knocking out a subset of genes:

$$\text{Gene Knockout Problem: } \max_{x\in\{0,1\}^N} f(x)=g_z[T] \quad (4)$$

$$\text{subject to } \quad \mathbf{g}[T]=(\text{diag}(x)A\ \text{diag}(x))^T\mathbf{g}[0], \quad (5)$$

$$x_z=1. \quad (6)$$

Let x^* be an optimal solution to the GKO problem. As we want to maximize the concentration of downstream gene z, it should not be considered for deletion and hence the constraint $x_z=1$.

Notations – Path, Weight, and Contribution. We define path, weight of a path, and contribution of a path, to be used in the rest of the paper. A path from gene i at time 0 to gene z at time T over T time steps is a $T+1$ dimensional vector, $(k_0,k_1,\ldots,k_T)^T$, where $k_0=i$ and $k_T=z$. The path is illustrated in Fig. 1.

The weight of a path is $W(k_0,k_1,\ldots,k_T)=\Pi_{t=1}^{T} a_{k_t,k_{t-1}}$. The contribution of gene i to $g_z[T]$ through a path is defined by $g_i[0]\cdot W(k_0,k_1,\ldots,k_T)$. A path is negative/zero/positive if the contribution through the path to $g_z[T]$ is negative/zero/positive, indicating whether gene i influences $g_z[T]$ negatively or positively.

Time Complexity of the GKO Problem

We show that the GKO problem is *NP*-hard by reducing the *NP*-complete vertex cover problem to a special case of the GKO problem. Let $G=(V,E)$ be an undirected graph with a set V of n vertices and a set E of edges. A vertex cover is a subset of V that contains at least one end point of each edge in E. The vertex cover problem is to find a smallest vertex cover of G. We use C to represent the indices of vertices in a vertex cover of G.

Figure 1. A path over T time steps.
doi:10.1371/journal.pone.0009331.g001

The Vertex Cover Problem Is a Special Case of the GKO Problem. We construct a $(2n+1)\times(2n+1)$ matrix, A, from graph G by

$$A=\begin{bmatrix} A_1^{n\times n} & \mathbf{0}_1^{n\times n} & \mathbf{0}_2^{n\times 1} \\ B_1^{n\times n} & \mathbf{0}_3^{n\times n} & \mathbf{0}_4^{n\times 1} \\ -\mathbf{1}_v^{1\times n} & \mathbf{1}_v^{1\times n} & \mathbf{1}^{1\times 1} \end{bmatrix}=\{a_{ij}\}, \tag{7}$$

with

$$a_{ij}=\begin{cases} 3, & \text{if } (v_i,v_j)\in E \\ 2, & \text{if } i=n+j, j=1,\ldots,n \\ 1, & \text{if } i=2n+1, j=n+1,\ldots,2n+1 \\ -1, & \text{if } i=2n+1, j=1,\ldots,n \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

where A_1 is an $n\times n$ symmetric matrix, the row and column of whose non-zero entry corresponds to an edge in G, B_1 is a diagonal matrix $2I^{n\times n}$, $-\mathbf{1}_v$ is a $1\times n$ matrix whose entries are all -1, $\mathbf{1}_v$ is a $1\times n$ matrix whose entries are all 1, and $\mathbf{0}_1^{n\times n}$, $\mathbf{0}_2^{n\times 1}$, $\mathbf{0}_3^{n\times n}$, and $\mathbf{0}_4^{n\times 1}$ are all zero matrices.

Now, we formulate the GKO' problem of $2n+1$ genes, a special case of the GKO problem, as

$$\text{GKO}': \quad \max_{x\in\{0,1\}^{2n+1}} \; g_{2n+1}[2] \tag{9}$$

$$\text{subject to} \quad \mathbf{g}[2]=(\text{diag}(x)\ A\ \text{diag}(x))^2\mathbf{g}[0], \tag{10}$$

$$x_{2n+1}=1, \tag{11}$$

$$\mathbf{g}[0]=(\underbrace{1,1,\ldots,1}_{n\text{ items}},\underbrace{0,0,\ldots,0}_{n+1\text{ items}})^T. \tag{12}$$

The $2n+1$ genes in the GKO' problem can be separated into three groups by their indices: $\{1,\ldots,n\}$, $\{n+1,\ldots,2n\}$, and $\{2n+1\}$. Only paths originating from group one, shown in Fig. 2, influence $g_{2n+1}[2]$. All other paths to $g_{2n+1}[2]$, not shown, originating from either group two or three, contribute zero to $g_{2n+1}[2]$. We further define three types of paths, shown in Fig. 2, all originating from some genes in group one, as follows:

- Type 1 path: it goes from a gene in group one at time 0, via another gene in group one at time 1, to gene $2n+1$ at time 2 with a weight of $3\cdot-1=-3$. Both gene i and j contribute -3 through their corresponding type 1 paths if (v_i,v_j) is an edge in graph G and both gene i and j exist in the network. Therefore, the number of type 1 paths is the number of nonzero elements in A_1.
- Type 2 path: it goes from group one, via group two, to gene $2n+1$ with a weight of $2\cdot1=2$. A gene in group one contributes 2 to $g_{2n+1}[2]$ through its corresponding type 2 path if it exists in the network. Therefore, the number of type 2 paths is the number of existing genes in group one.
- Type 3 path: it goes from group one, via gene $2n+1$, to gene $2n+1$ with a weight of $-1\cdot1=-1$. A gene in group one contributes -1 to $g_{2n+1}[2]$ through its corresponding type 3 path if it exists in the network. Therefore, the number of type 3 paths is the number of existing genes in group one.

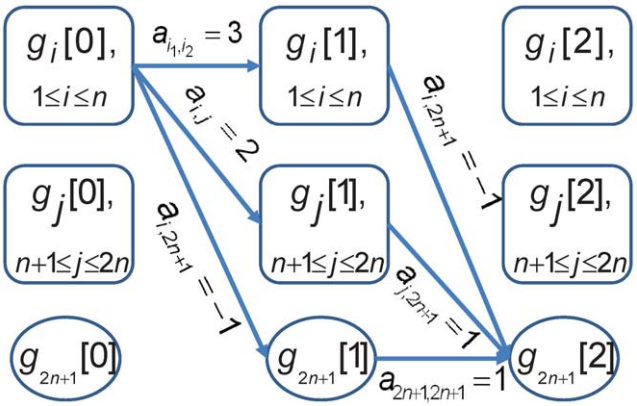

Figure 2. Schematic diagram for the three types of paths influencing $g_{2n+1}[2]$.
doi:10.1371/journal.pone.0009331.g002

As the initial state is non-negative, no genes in group two should be knocked out, because doing so would not possibly increase $g_{2n+1}[2]$, due to type 2 paths being non-negative. Thus, we consider deleting genes from group one as only feasible solutions to the GKO' problem.

Evidently, it takes polynomial time $O(n^2)$ to construct the GKO' problem from the vertex cover problem.

The GKO' Problem and the Vertex Cover Problem Are Equivalent. The two problems are equivalent if and only if any smallest vertex cover C^* of the vertex cover problem translates to an optimal solution x^* to the GKO' problem and *vice versa.*

Let X be the set of all feasible solutions to the GKO' problem. Let Θ be the power set of $\{1,2,\ldots,n\}$ representing all subsets of vertices in G. We define a bijective function, ϕ, from Θ to X by

$$x=\phi(\theta): \quad x_i=\begin{cases} 0, & \text{if } i\in\theta \\ 1, & \text{if } i\notin\theta \end{cases}, \quad i=1,\ldots,2n+1 \tag{13}$$

Function ϕ translates any subset $\theta\in\Theta$ of vertices in G to a feasible solution $x\in X$ to the GKO' problem with a corresponding objective function value $g^{\theta}_{2n+1}[2]$. When $\theta=C^*$, the objective function value is $g^{C^*}_{2n+1}[2]$.

Lemma 1. *If C is a vertex cover of graph G, then $g^{C}_{2n+1}[2]=n-|C|$.*

Proof. By Fig. 2, there are three types of paths influencing $g^{C}_{2n+1}[2]$. Since C is a vertex cover for graph G, A_1 in equation (7) of matrix $A(\phi(C))$ is a zero matrix. That means there is no network between any two genes in group one and, then, genes contribute nothing to $g_{2n+1}[2]$ through a type 1 path if we delete all gene i for all i in C from the GKO' problem. However, each gene i in group one, which is not deleted, contributes two through a type 2 path and negative one through a type 3 path. Therefore,

$$g^{C}_{2n+1}[2]=[2+(-1)](n-|C|)=n-|C|.$$

Lemma 2. *If C is a vertex cover of graph G, then $g^{C}_{2n+1}[2]\le g^{C^*}_{2n+1}[2]$.*

Proof. Since C^* is the index set for a minimum cover, we have

$$|C^*| \leq |C|.$$

According to Lemma 1, we have.

$$g^{C}_{2n+1}[2] = (n - |C|) \leq (n - |C^*|) = g^{C^*}_{2n+1}[2].$$

Lemma 3. *Let θ be a non-vertex-cover subset of vertices. Let C be a smallest vertex cover that subsumes θ. Then $g^{\theta}_{2n+1}[2] \leq g^{C}_{2n+1}[2]$ holds true.*

Proof. According to Fig. 2, one additional type 1 path contributes -3 to gene $2n+1$ at time two while one additional type 2 path contributes 2 to gene $2n+1$ and one additional type 3 path contributes -1 to gene $2n+1$.

Since C is a vertex cover and θ belongs to C, genes in group one have several additional paths to contribute nonzero values to g_{2n+1} at time two if we only delete gene i for all i in θ instead of in C. Let the difference of sets C and θ be C^{Δ}. One more gene adding into the network from C^{Δ} causes more than one additional nonzero element in A_1 in equation (7). We know that the number of type 1 paths is the number of nonzero elements in A_1. Therefore, the total contribution from the additional type 1 paths is less than

$$\sum_{i \in C^{\Delta}} -3g_i[0]. \tag{14}$$

As the number of type 2 or 3 paths is the number of existing genes in group one, the contribution from the additional type 2 paths is

$$\sum_{i \in C^{\Delta}} 2g_i[0], \tag{15}$$

and that from the additional type 3 paths is

$$\sum_{i \in C^{\Delta}} -g_i[0], \tag{16}$$

The total contribution of those additional paths is less than

$$\sum_{i \in C^{\Delta}} -2g_i[0]. \tag{17}$$

Since the value in equation (17) is negative, value $g^{\theta}_{2n+1}[2]$ is less than $g^{C}_{2n+1}[2]$ and this lemma is proved.

Combining Lemmas 2 and 3 establishes that $g^{\theta}_{2n+1}[2] \leq g^{C^*}_{2n+1}[2]$ for any subset $\theta \in \Theta$ of vertices in G if C^* is a smallest vertex cover. Let x^* be an optimal solution of GKO' and $g^*_{2n+1}[2]$ be its maximal value. We have the following two propositions.

Proposition 4. *If C^* is a smallest vertex cover of G, then $g^{C^*}_{2n+1}[2] = g^*_{2n+1}[2]$.*

Proof. (By contrapositive) Assume $g^{C^*}_{2n+1}[2] < g^*_{2n+1}[2]$. x^* can be translated to C^o by ϕ^{-1}. If C^o is not a smallest vertex cover, $g^{C^*}_{2n+1}[2] < g^*_{2n+1}[2] = g^{C^o}_{2n+1}[2]$ contradicts either Lemma 2 or 3. If C^o is a smallest vertex cover, we have $|C^o| = |C^*|$. Then $g^{C^*}_{2n+1}[2] < g^*_{2n+1}[2]$ contradicts Lemma 1. Thus, $g^{C^*}_{2n+1}[2] \not< g^*_{2n+1}[2]$. By definition of $g^*_{2n+1}[2]$, it is also impossible to have $g^{C^*}_{2n+1}[2] > g^*_{2n+1}[2]$. Therefore, we must have $g^*_{2n+1}[2] = g^{C^*}_{2n+1}[2]$.

Proposition 5. *Let C^o be $\phi^{-1}(x^*)$. Then, C^o is a smallest vertex cover of G.*

Proof. (By contrapositive) Assume $C^o = \phi^{-1}(x^*)$ is not a smallest vertex cover of G. Then one can find a smallest vertex cover C^* of G. Thus, it must follow by either Lemma 2 or 3 that $g^{C^*}_{2n+1}[2] > g^*_{2n+1}[2]$, which contradicts the fact that $g^*_{2n+1}[2]$ is maximal. Therefore, C^o must be a smallest vertex cover with $|C^o| = |C^*|$.

Propositions 4 and 5 establish that the GKO' and the vertex cover problems are equivalent.

The GKO Problem is *NP*-Hard. Theorem 6. *The GKO problem is NP-hard. Proof.* By Propositions 4 and 5, any solution to the vertex cover problem translates to a solution to the GKO' problem and *vice versa*. Since the vertex cover problem is in its most general form, any instance of the vertex cover problem is thus reducible to the GKO' problem. As the vertex cover problem is *NP*-complete and it can be reduced in polynomial time to the GKO' problem, a special case of the GKO problem, the GKO problem is *NP*-hard.

The Approximation Algorithm of GKONP

As the number of feasible solutions to the GKO problem increases exponentially with network size N, it is impractical to solve it by exhaustive search when N is large. Using the concept of paths, the GKO problem is rewritten to an equivalent nonlinear programming problem. Combining a strategy on pruning the insignificant terms in the objective function and a differential evolution algorithm, we provide a heuristic algorithm to the *NP*-hard GKO problem.

Nonlinear Programming for the GKO Problem. We rewrite the nonlinear integer programming problem to an equivalent nonlinear programming problem. Let $P_T(i,z)$ be the collection of paths from gene i to gene z over T time steps. The sum of contributions of various paths from gene i to z over T time steps is

$$\sum_{(k_0,k_1,\ldots,k_T) \in P_T(i,z)} W(k_0,k_1,\ldots,k_T) \times g_i[0]. \tag{18}$$

It follows that the objective function $f(x)$ of the GKO problem is the sum of contributions from all genes to gene z:

$$f(x) = \sum_{i=1}^{N} \left(\sum_{(k_0,k_1,\ldots,k_T) \in P_T(i,z)} \left(\prod_{j=0}^{T} x_{k_j} \right) W(k_0,k_1,\ldots,k_T) g_i[0] \right). \tag{19}$$

A path $(k_0,k_1,\ldots,k_T)$ may visit a gene more than once. We extract the unique genes on the path to form a set $\{k'_0,k'_1,\ldots,k'_{\mu}\}$, $\mu \leq T$. As each element in $(x_{k_0},x_{k_1},\ldots,x_{k_T})$ is either zero or one, we have $\prod_{j=0}^{T} x_{k_j} = \prod_{j=0}^{\mu} x_{k'_j}$. Then, $f(x)$ can be rewritten as

$$f(x) = \sum_{i=1}^{N} \left(\sum_{(k_0,k_1,\ldots,k_T) \in P_T(i,z)} \left(\prod_{j=0}^{\mu} x_{k'_j} \right) W(k_0,k_1,\ldots,k_T) g_i[0] \right). \tag{20}$$

Only a negative path, $(k_0,k_1,\ldots,k_T)$, in $P_T(i,z)$ gives a negative term, $W(k_0,k_1,\ldots,k_T)g_i[0]$, in equation (20). Therefore, we shall delete genes in negative paths to maximize equation (20) and those genes only on non-negative paths need not to be considered for deletion. We denote the collection of genes on negative paths to gene z by $S^{-}_{T}(z)$. Then, the size of feasible solutions of the nonlinear integer programming problem is scaled down from 2^N to $2^{|S^{-}_{T}(z)|}$.

Let $S_T^-(i,z)$ be the collection of genes on negative paths from gene i to gene z. Let $\{k_0'', k_1'', \ldots, k_v''\}$ represent the intersection of $S_T^-(i,z)$ and $\{k_0', k_1', \ldots, k_\mu'\}$. It follows $\Pi_{j=0}^{\mu} x_{k_j'} = \Pi_{j=0}^{v} x_{k_j''}$. The objective function becomes

$$f(x)=\sum_{i=1}^{N}\left(\sum_{(k_0,k_1,\ldots,k_T)\in P_T(i,z)}\left(\prod_{j=0}^{v} x_{k_j''}\right)W(k_0,k_1,\ldots,k_T)g_i[0]\right). \quad (21)$$

Lemma 7. *If a nonlinear programming problem has objective function $f(x)$ (equation 21) and all decision variables x_i, $i\in\{k_0'',k_1'',\ldots,k_v''\}$, bounded by $[0,1]$, then there exists an optimal solution which is a vertex of the feasible hypercube.*

Proof. Assume x^*, $(x_{k_0^*},x_{k_1^*},\ldots,x_{k_T^*})$, is an optimal solution but not a vertex. Therefore, there must exist an element $0<x_r<1$ in the solution. Then, the value of objective function (equation 21) with this solution is

$$x_r\sum_{i=1}^{N}\left(\sum_{\substack{(k_0,k_1,\ldots,k_T)\in P_T(i,z)\\ r\in(k_0'',k_1'',\ldots,k_v'')}}\frac{x_{k_0''}x_{k_1''}\ldots x_{k_v''}}{x_r}W(k_0,k_1,\ldots,k_T)g_i[0]\right)+\sum_{i=1}^{N}\left(\sum_{\substack{(k_0,k_1,\ldots,k_T)\in P_T(i,z)\\ r\notin(k_0'',k_1'',\ldots,k_v'')}}x_{k_0''}x_{k_1''}\ldots x_{k_v''}W(k_0,k_1,\ldots,k_T)g_i[0]\right). \quad (22)$$

If the value of function (equation 22) is positive at point $(x_{k_0^*},x_{k_1^*},\ldots,x_{k_T^*})$, we can increase x_r to one to improve the value. Otherwise, we decrease x_r to zero. Since we can improve the value of objective function (equation 21) by moving x^* to a vertex of the hypercube search space, this lemma is proved.

By Lemma 7, the original GKO problem becomes Nonlinear Programming for the GKO Problem

$$\max f(x)=\sum_{i=1}^{N}\left(\sum_{(k_0,k_1,\ldots,k_T)\in P_T(i,z)}\left(\prod_{j=0}^{v} x_{k_j''}\right)W(k_0,k_1,\ldots,k_T)g_i[0]\right) \quad (23)$$

$$\text{subject to}\quad 0\le x_r\le 1,\ \text{for } r\in S_T^-(z). \quad (24)$$

The Filter-Dynamical-Path Algorithm. We introduce the Filter-Dynamical-Path (FDP) algorithm to approximate the objective function $f(x)$ in the form of equation (23). The FDP algorithm, generating the terms of objective function $f(x)$ step by step backward from time T to time 0, discards insignificant terms at each step. Since the long run behavior of most GRNs shall be stable, A^t in the DDS model also has to be such when t increases. The contributions of most paths will thus vanish and the corresponding terms are removed by the FDP algorithm when time t is long enough.

Let $\mathcal{P}^+(j,t,z,T)$ ($\mathcal{P}^-(j,t,z,T)$) denote the collection of those positive (negative) paths through gene j at time t to gene z at time T and their weights. $\mathcal{P}(j,t,z,T)$ represents the union of $\mathcal{P}^+(j,t,z,T)$ and $\mathcal{P}^-(j,t,z,T)$. $W_j^+[t]$ ($W_j^-[t]$) denotes the total weight of positive (negative) paths in $\mathcal{P}^+(j,t,z,T)$ ($\mathcal{P}^-(j,t,z,T)$). The FDP algorithm, moving backward over time, removes those

for $t = T-1$ down to 0 **do**
  for each node $j \in S$ at time t **do**
    for each child node i at time $t+1$ of node j at time t **do**
      Let $a_{i,j}$ be the weight from node j at time t to node i at time $t+1$
      Update $\mathcal{P}^+(j,t,z,T)$ from $a_{i,j}$ and $\mathcal{P}(i,t+1,z,T)$
      Update $\mathcal{P}^-(j,t,z,T)$ from $a_{i,j}$ and $\mathcal{P}(i,t+1,z,T)$
    end
    Combine those paths in $\mathcal{P}^+(j,t,z,T)$ that passes through the same nodes after time t
    Combine those paths in $\mathcal{P}^-(j,t,z,T)$ that passes through the same nodes after time t
    $W_j^+[t]$ = sum of all weights of paths in $\mathcal{P}^+(j,t,z,T)$
    $W_j^-[t]$ = sum of all weights of paths in $\mathcal{P}^-(j,t,z,T)$
    for $i = 1$ to $|\mathcal{P}^+(j,t,z,T)|$ **do**
      Remove path i from $\mathcal{P}^+(j,t,z,T)$ if both the weight of path i and the total weight of those removed paths are less than $\sigma \cdot W_j^+[t]$.
    end
    for $i = 1$ to $|\mathcal{P}^-(j,t,z,T)|$ **do**
      Remove path i from $\mathcal{P}^-(j,t,z,T)$ if both the weight of path i and the total weight of those removed paths are less than $\sigma \cdot W_j^-[t]$.
    end
  end
end
return $\mathcal{P}(1,0,z,T),\ldots,\mathcal{P}(N,0,z,T)$

Figure 3. Algorithm 1. Filter-Dynamical-Path(A, S, z, T, σ).
doi:10.1371/journal.pone.0009331.g003

1. Call the FDP algorithm:
   (a) Generate the terms of objective function (equation(23)) step by step backward from time T to time 0 and, at each step, remove those insignificant terms relying on σ
   (b) Obtain a simplified approximate objective function $\tilde{f}(x)$ for the GKO problem
2. Call the DE algorithm:
   (a) Generate an initial solution set $X = \{x^1, x^2, \ldots, x^\tau\}$
   (b) Evolve set X in the feasible solution space to increase the value of function $\tilde{f}(x)$
   (c) Stop if the difference of the maximum and minimum of $\{\tilde{f}(x^i)|x^i \in X\}$ is less than ρ
3. Return the best solution x^* from X

Figure 4. Algorithm 2. GKONP(prune coefficient σ, tolerance ρ).
doi:10.1371/journal.pone.0009331.g004

positive (negative) terms such that the total weight of the related paths is at most σ of the total weight of the remaining positive (negative) paths. We call σ the prune coefficient. Let $\tilde{f}$ be the approximate value to the true objective function value f. In our simulation study, the relative error is roughly bounded by

$$\left|\frac{\tilde{f}-f}{f}\right| \le \frac{\sigma-\sigma^{T-1}}{1+\sigma}, \text{ if } T \text{ is odd; or } \left|\frac{\tilde{f}-f}{f}\right| \le \frac{\sigma+\sigma^{T-1}}{1+\sigma}, \text{ if } T \text{ is even.}$$

The inequalities suggest that the smaller σ is, the closer the approximation is to the true value. For instance, when σ is 0.001 and T is 10, we have $0.99f \le \tilde{f} \le 1.01f$. Details of the FDP algorithm is shown as Fig. 3.

The GKONP Algorithm. Based on the nonlinear formulation, we develop a heuristic algorithm to solve the GKO problem. We call it the GKONP algorithm, shown as Fig. 4.

It combines the FDP algorithm and a differential evolution (DE) algorithm for nonlinear programming. The GKONP algorithm simplifies the objective function first by the FDP algorithm and then use the DE algorithm to obtain a final solution to the GKO problem.

The DE algorithm [17,18] approaches a global maximum of non-concave objective functions as in the GKO problem. The DE algorithm is an evolutionary optimization method. The first step is to generate an initial population of feasible solutions, typically 2 to 50 times of the decision variables. Each individual in the population either remains unchanged or mutates to a new feasible solution in one iteration of evolution. The occurrence of a mutation depends on a trial vector and a probability p. The trial vector combines three other individuals, randomly chosen from the population. If the trial vector is a feasible solution and improves the value of objective function, then the individual mutates to the trial vector with probability p. Since the evolution of an individual is independent of others, evolutions of individuals can progress simultaneously and hence can be done in parallel.

Results

The GKONP algorithm is applied to improve the concentrations of downstream proteins or protein complexes Fus3PP, Fur1PP-Cdc28 (complex N) and Fur1PP-G$\beta\gamma$ (complex M), involved in the yeast pheromone pathways. Moreover, we evaluate our algorithm on randomly generated DDS models to illustrate its empirical accuracy and running time.

Optimal Deletion in the Yeast Pheromone Pathways

We demonstrate our GKONP algorithm using a realistic *Saccharomyces cerevisiae* pheromone pathway model developed by Kofahl et al. [14], shown in Fig. 5. The model is obtained after they

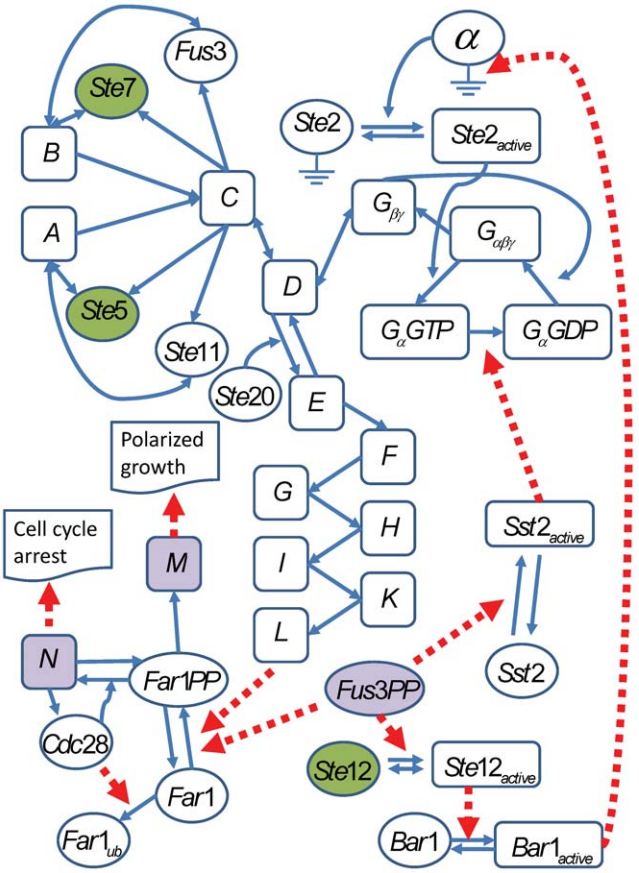

Figure 5. The schematic diagram for the pheromone pathway [14]. The ellipse shapes represent proteins while the rectangle shapes represent protein complexes. The solid lines represent the intracellular reactions while the thick dash lines represent catalysis. We note that the decomposition from complexes E, F, G, H and L to proteins Ste20, $G_{\beta\gamma}$, Ste5, Ste11, Ste7 and the dephosphorylation of Fu3PP are not shown in the diagram since they are less dominant than those shown in the pheromone pathway.
doi:10.1371/journal.pone.0009331.g005

studied cell cycle arrest, mating activity, and pheromone sensitivity. The model is publicly available from the BioModels database [19] in the form of a dynamical system model composed of ordinary differential equations (ODEs). The pheromone signaling pathway involves a series of biochemical reactions starting with the receptor of MAT_a receiving pheromone α factor from haploid MAT_{alpha}. From the cytoplasm, the pheromone signal enters the nucleus to express downstream protein Fus3PP, protein complex N and protein complex M, which together control pheromone sensitivity, cell polarity and cell cycle arrest for preparation of cell fusion between two mating haploid yeast cells, MAT_{alpha} and MAT_a. Haploid MAT_a cannot stop cell cycle to mate with MAT_{alpha} if the concentrations of the three protein products are low. Therefore, it is desirable to engineer the yeast to improve these downstream protein products to increase mating activity.

Thus, we applied the GKONP algorithm to identify upstream knockout genes to improve the concentrations of the three downstream protein products. By simulation using the ODE model, we first generated continuous-time trajectories. Second, we sampled them every 0.6 seconds from 0 to 6 seconds to obtain discrete-time trajectories. Then we reconstructed a DDS model (Appendix S1) from the discrete-time trajectories using a data-driven method [13] that balances goodness-of-fit and model complexity. The DDS model captures the transient dynamics in the pathway in which the three protein products are actively expressed. Using the DDS model as input, we ran the GKONP algorithm three times to search for three optimal target gene sets in the pathway for improving the concentrations of downstream products of Fus3PP, complex N and complex M, respectively. A feasible solution is any subset of {Ste2, Ste5, Ste11, Ste7, Ste20, Ste12, Fus3PP, Bar1, Far1PP, Cdc28}. The optimal target gene sets for improving each of Fus3PP, complex N and complex M are {Ste5, Ste7, Ste12},{Ste12} and {Ste12}, respectively. The optimal target genes obtained through GKONP algorithm were validated in the original ODE model. By assigning zero values to the deleted genes, we simulated the modified dynamics of the engineered ODE model. Figure 6 presents the transient dynamics, computed from the original ODE model as a validation, of the concentrations of Fus3PP, complex N and complex M in the wild type and five mutants from 0 to 6 seconds. The modified dynamics are compared with those of wild type and three observed mutants which have high concentrations of at least one of those three downstream protein products. These three observed mutants include a mutant whose $G_{\beta\alpha}$ is overexpressed (double amount of $G_{\beta\alpha}$) [20], a mutant whose Ste2 loses function (the hydrolysis of $G_{\alpha}GTP$ to $G_{\alpha}GDP$ is almost stopped) [21] and a mutant that has no phosphatase activity on Fus3PP (the concentration of Fus3PP is strongly increased for a long time) [22]. From this figure, we demonstrate that, by deleting optimal subsets of target genes, the concentrations of all three desirable downstream protein products are higher than the wild type and the trial-and-error *in vivo* mutants.

Accuracy and Time Efficiency

Nine randomly generated DDS models were used to test the performance of the GKONP algorithm. The model sizes are

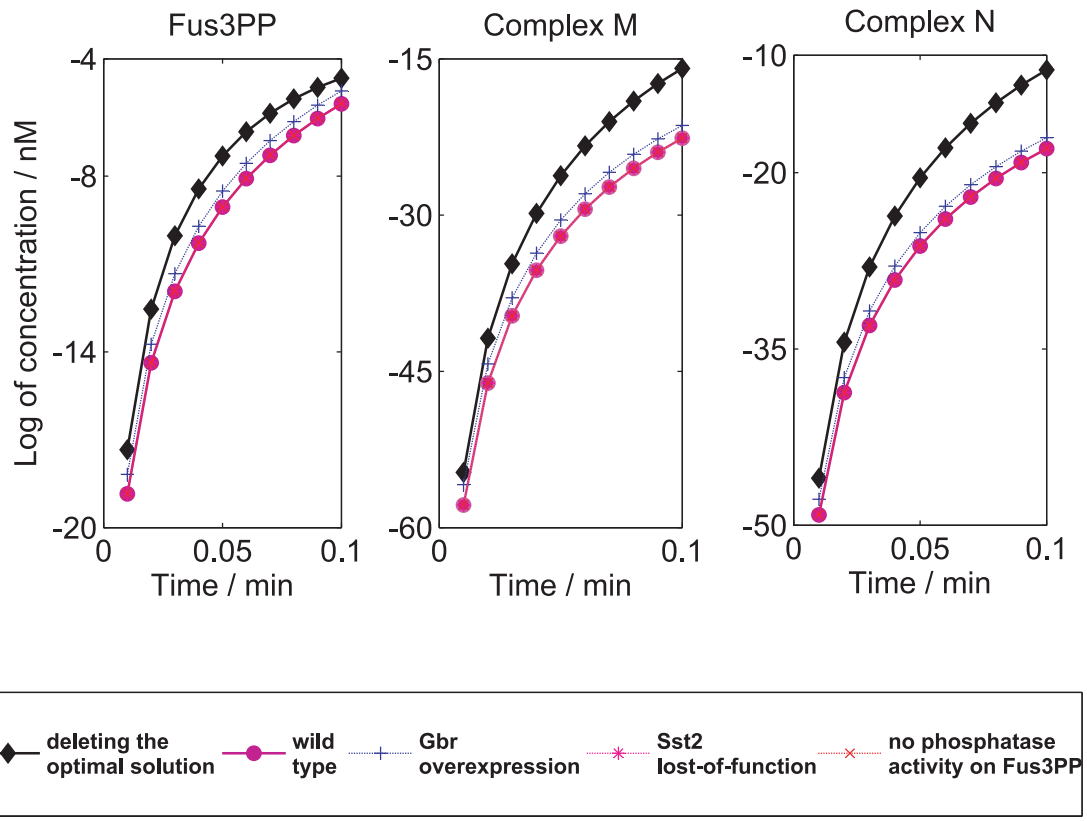

Figure 6. The optimal solution from the GKONP algorithm vs. the wild type and three observed mutants. The GKONP optimization (top diamond lines) returned higher concentrations of desired downstream proteins (Left: Fus3PP, Middle: Complex M, and Right: Complex N) than both the wild type and the *in vivo* trial-and-error mutants [14]. Left: the optimal target genes are Ste5, Ste7 and Ste12. Middle and right: the optimal target gene are both Ste12. The dynamics of wild type, the mutant (Sst2 lost-of-function), and the mutant without phosphatase activity on Fus3PP overlap in all three plots.
doi:10.1371/journal.pone.0009331.g006

10, 20 and 30, with 3 instances for each size. As each gene is only regulated directly by a very small portion of its network, without loss of generality, we constrained each DDS model such that each gene has less than three parent genes. Thus, no more than three entries were non-zero in each row of matrix A, whose values were between -1 to 1. Each gene had one unit of concentration at time 0. We applied the GKONP algorithm on the DDS models to maximize the concentration of the first gene at time 10. Prune coefficient σ was set to 10^{-5}. We also ran the brute force exhaustive search algorithm on the DDS models as a reference for performance evaluation of the GKONP.

Table 1 shows the GKONP algorithm approaches optimal solutions accurately because all six approximations for the 10 gene and 20 gene DDS models are identical to the optimal values, given by the exhaustive algorithm.

The running time as a function of model sizes is shown in Fig. 7. The exhaustive search algorithm for the 30-gene DDS models took more than five days and we rounded the time to five days. The speedup of the GKONP algorithm ranges from 0.09 to 1.42 in the 10-gene models, 47 to 11,388 in the 20-gene models, and 4,763 to 165,390 in the 30-gene models. Therefore, Figure 7 suggests that the GKONP algorithm is much more efficient than the exhaustive search algorithm.

In Fig. 8 the number of paths decreases significantly after the FDP algorithm is applied, so that the DE algorithm runs in a reduced search space.

Since the GKONP searches negative paths of a DDS model, it was slower with 10 genes than the exhaustive search. However, as the number of genes increases to 20 and 30, our approach has extraordinary speedup over the exhaustive search. With the same model size, the running time advantage of GKONP becomes evident when the topology of a DDS model contains either few negative paths or very few genes in negative paths. For instance, a 20-gene DDS model had 8 genes in negative paths and the GKONP yielded a speedup of 11,388 versus another 20-gene model with a speedup of only 47.

This simulation study demonstrates empirically that the GKONP algorithm has achieved good accuracy in a practical amount of running time.

Discussion

We have established that the optimal *in silico* target gene deletion problem is challenging, by showing that a nonlinear integer programming formulation of the GKO problem based on the DDS model is *NP*-hard. A nonlinear programming solution is provided that combines heuristics based on the sparsity of typical GRNs and a parallel differential evolution algorithm for nonlinear programming. Multiple simultaneous gene deletion is handled in our approach, while all existing strategies delete one gene at a time. Our algorithm GKONP has shown its substantially reduced running time and comparable accuracy with the optimal solutions using exhaustive search algorithms. Demonstration of our solution on a realistic model of yeast pheromone pathways has suggested potential impact of our work. Hopefully, ideas presented in this paper will bring out potentially harder but biologically more viable computational problems for richer formulation of the target gene deletion problem, based on more complex dynamical system models of gene regulatory and metabolic networks with additional constraints on side effects.

Table 1. Approximate GKONP solutions versus the optimal solutions.

Data set	# Genes	Optimal value	GKONP value
1	10	0.1734801	0.1734801
2	10	0.3546365	0.3546365
3	10	0.01020017	0.01020017
4	20	0	0
5	20	0.02599726	0.02599726
6	20	0.1683985	0.1683985

doi:10.1371/journal.pone.0009331.t001

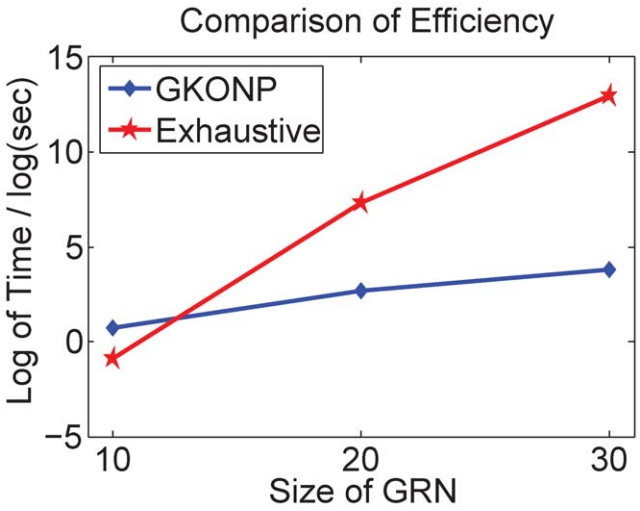

Figure 7. Average running time of the GKONP algorithm and the exhaustive search algorithm. The GKONP algorithm is represented by the blue line with diamonds while the exhaustive search algorithm is by red line with circles.
doi:10.1371/journal.pone.0009331.g007

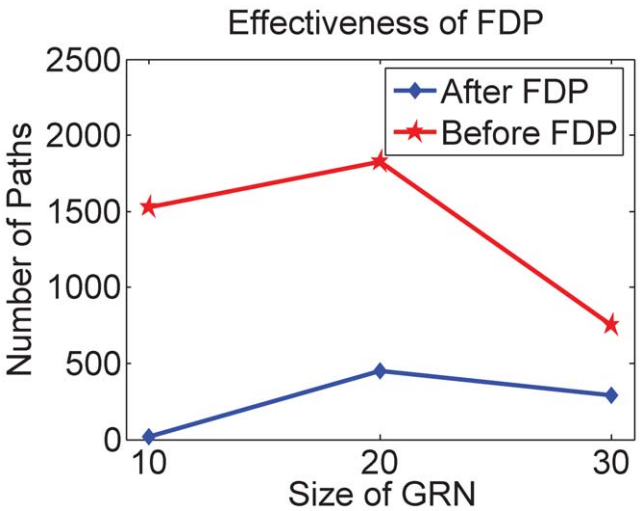

Figure 8. Average path reduction by the FDP algorithm. The red line with circles represents the number of paths before FDP reduction while the blue line with diamonds represents the number after FDP reduction.
doi:10.1371/journal.pone.0009331.g008

Supporting Information

Appendix S1 The DDS Model for the Pheromone Pathway
Found at: doi:10.1371/journal.pone.0009331.s001 (0.03 MB PDF)

Author Contributions

Conceived and designed the experiments: CCH MS. Performed the experiments: CCH. Analyzed the data: CCH. Contributed reagents/materials/analysis tools: CCH MS. Wrote the paper: CCH MS.

References

1. Sticklen MB (2008) Plant genetic engineering for biofuel production: towards affordable cellulosic ethanol. Nature Reviews Genetics 9: 433–443.
2. Maguire CA, Meijer DH, LeRoy SG, Tierney LA, Broekman ML, et al. (2008) Preventing growth of brain tumors by creating a zone of resistance. Molecular Therapy 16: 1695–1702.
3. Deutscher D, Meilijson I, Kupiec M, Ruppin E (2006) Multiple knockout analysis of genetic robustness in the yeast metabolic network. Nature Genetics 38: 993–998.
4. Nakae J, Oki M, Cao Y (2008) The FoxO transcription factors and metabolic regulation. FEBS Letters 582: 54–67.
5. Faryabi B, Vahdi G, Chamberland JF, Datta A, Dougherty ER (2008) Optimal constrained stationary intervention in gene regulatory networks. EURASIP Journal on Bioinformatics and Systems Biology 1: 1.
6. Alper H, Jin YS, Moxley JF, Stephanopoulos G (2005) Identifying gene targets for the metabolic engineering of lycopene biosynthesis in *Escherichia coli*. Metabolic Engineering 7: 155–164.
7. Jin YS, Stephanopoulos G (2007) Multi-dimensional gene target search for improving lycopene biosynthesis in *Escherichia coli*. Metabolic Engineering 9: 337–347.
8. Holter NS, Maritan A, Cieplak M, Fedoroff NV, Banavar J (2000) Dynamic modeling of gene expression data. PNAS 98: 1693–1698.
9. Dewey TG, Galas DJ (2001) Dynamic models of gene expression and classification. Functional and Integrative Genomics 1: 269–278.
10. Shmulevich I, Dougherty ER, Zhang W (2002) Control of stationary behavior in probabilistic Boolean networks by means of structural intervention. Biological Systems 10: 431–445.
11. Goutsias J, Kim S (2004) A nonlinear discrete dynamical model for transcriptional regulation: Construction and properties. Biophysical Journal 86: 1922–1945.
12. Wang Y, Joshi T, Zhang XS, Xu D, Chen L (2006) Inferring gene regulatory networks from multiple microarray datasets. Bioinformatics 22: 2413–2420.
13. Song M, Ouyang Z, Liu ZL (2009) Discrete dynamical system modeling for gene regulatory networks of HMF tolerance for ethanologenic yeast. IET Systems Biology 3: 203–218.
14. Kofahl B, Klipp E (2004) Modeling the dynamics of the yeast pheromone pathway. Yeast 21: 831–850.
15. Meir E, Munro EM, Odell GM, Dassow GV (2002) Ingeneue: A versatile tool for reconstituting genetic networks, with examples from the segment polarity network. Journal of Experimental Zoology 294: 216–251.
16. Takahashi K, Arjunan SNV, Tomita M (2005) Space in systems biology of signaling pathways towards intracellular molecular crowding *in silico*. FEBS Letters 579: 1783–1788.
17. Storn R, Price K (1995) Differential evolution - A simple and efficient adaptive scheme for global optimization over continuous spaces. Technical report, ICSI. November.
18. Lampinen J, Zelinka I (1999) Mixed variable nonlinear optimization by differential evolution. In: Proceedings of Nostradamus '99. volume 2. pp 45–55.
19. Bornstein BJ, Keating SM, Jouraku A, Hucka M (2008) LibSBML: An API Library for SBML. Bioinformatics 24: 880–881.
20. Cole GM, Stone DE, Reed SI (1990) Stoichiometry of G protein subunits affects the *Saccharomyces cerevisiae* mating pheromone signal transduction pathway. Mol Cell Biol 10: 510–517.
21. Dohlman HG, Thorner JW (2001) Regulation of G protein-initiated signal transduction in yeast: paradigms and principles. Annu Rev Biochem 70: 703–754.
22. Zhan XL, Deschenes RI, Guan KI (1997) Differential regulation of FUS3 MAP kinase by tyrosine-specific phosphatases PTP2/backslashPTP3 and dual-specificity phosphatase MSG5 in *Saccharomyces cerevisiae*. Genes Dev 11: 1690–1702.