PreProPath Algorithm: An Uncertainty-Aware Algorithm for Identifying Predictable Profitable Pathways in Biochemical Networks

Ehsan Ullah, *Member, IEEE,* Mark Walker, Kyongbum Lee, and Soha Hassoun, *Senior Member, IEEE*

Abstract—Pathway analysis is a powerful approach to enable rational design or redesign of biochemical networks for optimizing metabolic engineering and synthetic biology objectives such as production of desired chemicals or biomolecules from specific nutrients. While experimental methods can be quite successful, computational approaches can enhance discovery and guide experimentation by efficiently exploring very large design spaces. We present a computational algorithm, *PreProPath* (Predictably Profitable Path), to identify target pathways best suited for engineering modifications. The algorithm utilizes uncertainties about the metabolic networks operating state inherent in the underdetermined linear equations representing the stoichiometric model. Flux Variability Analysis is used to determine the operational flux range. *PreProPath* identifies a path that is predictable in behavior, exhibiting small flux ranges, and profitable, containing the least restrictive flux-limiting reaction in the network. The algorithm is computationally efficient because it does not require enumeration of pathways. The results of case studies show that *PreProPath* can efficiently analyze variances in metabolic states and model uncertainties to suggest pathway engineering strategies that have been previously supported by experimental data.

Index Terms—Flux Balance Analysis, Flux Variability Analysis, Metabolic Networks, Uncertainty

1 Introduction

Engineered cells have been used to produce various commercially significant bio-molecules, including biofuels [1], amino acids [2], and therapeutic proteins [3]. Current cell engineering approaches broadly fall into one of three categories. The first approach is to embed non-native reactions into a host organism to enable a synthesis route. For example, production of butanol [4], [5] and isopropanol [6], two potential biofuels, was enabled in *Escherichia coli* (*E. coli*) by importing different genes from *Clostridium acetobutylicum*. The second approach is to eliminate pathways that compete for cellular resources [1] or otherwise inhibit product synthesis. In a recent example, Yomano and co-workers deactivated the methyl glyoxal pathway to reduce catabolite repression and thereby accelerate co-metabolism of hexose and pentose sugars into ethanol [7]. The third approach is to tune the activities of existing pathways, for example by altering enzyme concentrations through gene expression changes. It should be noted that the above categorization is far from strict. Indeed, combinations of the various approaches are increasingly used to simultaneously enable new synthesis routes and optimize the yield. Keasling and co-workers have recently reported on engineered strains of *E. coli* capable of producing a variety of fatty esters (biodiesel), fatty alcohols, and waxes directly from simple sugars [8]. Fatty acid overproduction was achieved by over-expressing native thioesterases and acyl-CoA ligases while eliminating β-oxidation. To produce branched chain alcohols which are non-native to *E. coli*, a biosynthetic operon for branched chain amino acids (*thrABC*) was over-expressed, genes encoding competing pathways were deleted, and additional genes encoding the missing synthesis steps were imported from *Salmonella typhimurium* and *Corynebacterium glutamicum* [9].

A common thread in these approaches is that the engineered interventions targeted pathways, as opposed to individual reactions, as the functional units of cellular biosynthesis. While experimental approaches have often achieved significant success, the efficiency whereby the intervention targets may be identified and the optimality of results remain open questions due to the complexity of biological systems. In this regard, computational methods can serve as useful guides to efficiently explore the pathway design space.

Computational pathway analysis has shown great promise in rationally designing cells for efficient

- *E. Ullah is with the Department of Computer Science, Tufts University, Medford, MA, 02421. E-mail: ehsan.ullah@tufts.edu.*
- *M. Walker is with the Department of Chemical and Biological Engineering at Tufts University, Medford, MA 02421. E-mail: mark.walker@tufts.edu.*
- *K. Lee is with the Department of Chemical and Biological Engineering at Tufts University, Medford, MA 02421. E-mail: kyongbum.lee@tufts.edu.*
- *S. Hassoun is with the Department of Computer Science, Tufts University, Medford, MA, 02421. E-mail: soha.hassoun@tufts.edu.*

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCBB.2015.2394470, IEEE/ACM Transactions on Computational Biology and Bioinformatics

production of compounds [10] and maximization of stoichiometric yield [11], [12]. One commonly used analysis tool (e.g. [1], [11], [12], [13]) is elementary flux mode (EFM) analysis, which involves the enumeration of stoichiometrically balanced pathways feasible at steady state [14]. Any steady state flux distribution of a metabolic network can be expressed as a linear combination of the EFMs [13]. Unfortunately, EFM analysis is computationally intractable for large networks [15]. A recent analysis calculated 26 million EFMs for an relatively small *E. coli* network consisting of 106 reactions [16]. Importantly, from a cell engineering perspective, not all EFMs are of interest, as only a subset of EFMs may be engaged in the synthesis of the desired product molecule.

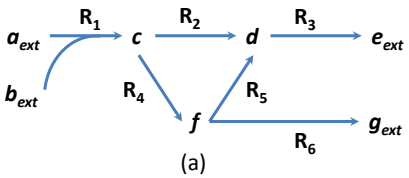
As an alternative to enumeration-based approaches, we investigate a graph-based approach to identify pathways as targets for metabolic engineering. Graph-based algorithms that search for specific attributes such as shortest path and bottleneck path [17] are computationally efficient with runtimes that are polynomial in the size of the graph. The algorithm presented here, which we call *Predictably Profitable Path* (*PreProPath*), searches for pathways that should be *up-regulated* to most predictably improve the yield of a biosynthetic product. Large-scale stoichiometric models of cellular metabolism nearly always have large degrees of freedom. Since the models are underdetermined, it is not possible to specify an operating point, i.e. unique flux distribution; rather, the model circumscribes an operating cone, i.e. flux ranges bounded by physicochemical, regulatory, and measurement-derived constraints. The flux ranges provide a quantitative basis to evaluate the profitability of an engineering intervention, as some interventions will only produce marginal improvements in flux that are subsumed by the uncertainty in the model. The *PreProPath* algorithm identifies a path from a starting substrate to a desired product that most likely contains one or more flux-limiting reactions, where the likelihood is determined by considering the degrees of freedom in the network. The *PreProPath* algorithm extends our earlier work [18] in using graph-based algorithms for pathway analysis while avoiding pathway enumeration. We evaluate the algorithm through two case studies and comparisons with other pathway engineering strategies and examples discussed in the literature.

2 Background and Definitions

2.1 Graph Representation

A metabolic network is represented by an $m \times n$ matrix $\mathbf{S}$, where each column corresponds to a reaction and each row corresponds to a metabolite. $\mathbf{S}$ can be expressed as a directed graph, G, with vertices, V, and edges, E, representing metabolites and reactions respectively. An edge in E may be a hyperedge, connecting two sets of vertices, as a reaction may have multiple reactants and/or products. A path is an alternating sequence of vertices and edges, $v_0, e_0, v_1, e_1, v_2, \ldots, e_{n-1}, v_n$, beginning and ending with vertices. A path between a source and a destination vertex may contain hyperedges such that some vertices associated with the hyperedges may not be part of such path.

Fig. 1(a) illustrates a small hypothetical network. Fig. 1(b) shows the corresponding stoichiometric matrix. Metabolites a_{ext}, b_{ext}, e_{ext}, and g_{ext} are not included in the $\mathbf{S}$ matrix as they are external to the network; however, exchange reactions R_1, R_3, and R_6 are included in the matrix.



(a)

	R_1	R_2	R_3	R_4	R_5	R_6
c	1	-1	0	-1	0	0
d	0	1	-1	0	1	0
f	0	0	0	1	-1	-1

(b)

Fig. 1. An example. (a) Network graph. (b) Corresponding S matrix.

2.2 Flux Variability Analysis

Given a metabolic network, we associate a flux (network flow), v_i, with each reaction i. It is possible to identify the maximum (or minimum) flux within a network by repeatedly applying Flux Balance Analysis (FBA) [19]. This procedure, known as Flux variability analysis (FVA) [20], identifies flux ranges by maximizing and minimizing each network flux subject to stoichiometric, physicochemical (e.g. thermodynamic irreversibility), and measurement constraints [21], [22]. Mathematically, FVA can be expressed as:

$$Maximize\ (or\ Minimize)\ v_i$$
$$subject\ to:$$
$$\mathbf{S} \cdot \mathbf{v} = \mathbf{0}$$
$$v_i^{lb} \leq v_i \leq v_i^{ub}$$

where the equation $\mathbf{S} \cdot \mathbf{v} = \mathbf{0}$ balances the fluxes into and out of a metabolite pool, and constrains the network to operate at quasi-steady state. For exchange reactions, flux bounds, v_i^{lb} and v_i^{ub}, represent the maximum and minimum nutrient uptake or secretion rates. Flux bounds for the rest of the reactions correspond to network constraints relevant to particular operating

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCBB.2015.2394470, IEEE/ACM Transactions on Computational Biology and Bioinformatics

conditions such as reaction directions. The maximum (minimum) flux value identified by FVA for a reaction i is denoted by v_i^{max} (v_i^{min}).

Applying FVA to the **S** matrix in Fig. 1(c) with the following bounds, $v_1 = 10$ and $v_6 \geq 2$, the resulting flux ranges, (v^{min} , v^{max}), for reactions R_1 through R_6 are: (10, 10), (0, 8), (0, 8), (0, 8), (2, 10), (2, 10).

2.3 Edge Weighting During Graph-Based Analysis

Flux values can be utilized as edge weightings, w_e, during graph-based analysis. It is possible to identify a path between a source, s, and a destination, d, utilizing one of the following edge weightings:

- To identify a path capable of carrying maximal flux, edge weights are assigned maximum flux values (v_i^{max}). This approach is optimistic as it assumes that the path is capable of operating under the most favorable conditions, which may not be attainable in practice.
- To guarantee a minimal flux flowing through a path, edge weights are assigned minimum flux values (v_i^{min}). This approach is conservative in identifying flux capabilities. Operationally, an edge along the path may carry a flux higher than the minimal flux v_i^{min}.
- To identify the path with the least flux variability (i.e. operationally providing the most predictable fluxes under specified operating conditions), each edge weight is assigned the flux range, the difference between the maximum and minimum reaction fluxes obtained using FVA. An edge with a low weight here indicates a more predictable operating condition when compared to an edge with a higher weight. Identifying a path from s to d utilizing these edge weights results in the most predictable path as it operates in the tightest flux ranges.

2.4 Definitions

When analyzing a graph, a particular weight of interest is the *bottleneckWeight*$_i$, which limits the maximum amount of flux flowing from s to d along any single path, p_i. Within a graph, and among all paths p_i between s and d, the maximum value among all *bottleneckWeight*$_i$ is referred to as the *bottleneckWeight* [23]. The edge associated with *bottleneckWeight* is a bottleneck edge. More formally,

$$bottleneckWeight = max_{\forall p_i} min_{e \in p_i} w_e$$

Any path between s and d capable of carrying a flux equal to or greater than the *bottleneckWeight* is referred to as a profitable path, as it contains the least restrictive flux-limiting reaction in the network and can be an engineering target that can yield profitable increase in yield.

When utilizing flux ranges as weights, one edge e_1 is less variable than an edge e_2 if $w(e_1)$ is less than $w(e_2)$. A path p_1 is less variable than a path p_2, if the maximum edge weight along p_1 is smaller than the maximum edge weight along p_2. If the maximum edge weights for p_1 and p_2 are equal, then successively smaller maximum weights along each path are compared instead. A path that is least variable is also the most predictable.

2.5 Predictable Profitable Path Conditions

For a given graph, G, a source vertex, s, destination vertex, d, and a flux range associated with each edge (v_i^{min} and v_i^{max}), the *PreProPath* algorithm finds the least variable path that contains the reactions capable of carrying the maximum flux from s to d. More specifically, *PreProPath* identifies a path p as a *predictably profitable path* if it meets the following two conditions:

Condition 1. Path p is profitable: all reactions along p are guaranteed to have a flux carrying capacity equal to or greater than the *bottleneckWeight*.

Condition 2. Path p is predictable: p is the least variable path among all profitable paths p_j.

In our work, we aim to first identify profitable paths within the network using Condition 1, and then to identify the most predictable path among all profitable ones using Condition 2.

To demonstrate these conditions, consider paths $P_1 - P_4$ from s to d in a hypothetical network graph. Each path consists of three edges, with v_i^{max} edge weights representing the maximum possible fluxes obtained using FVA:

$$P_1 = (30, 50, 100)$$
$$P_2 = (30, 70, 120)$$
$$P_3 = (30, 100, 110)$$
$$P_4 = (20, 80, 130)$$

Paths $P_1 - P_3$ have the same largest (among all paths) smallest (within path) weight of 30, which is the *bottleneckWeight*. P_1, P_2, and P_3 are equivalent in terms of flux capacity limits as the largest flux through each of these paths will be at most 30. Paths P_1, P_2, and P_3 therefore satisfy Condition 1; however, P_4 does not.

To demonstrate Condition 2, assume that the following weights are assigned to P_1, P_2, and P_3 based on flux ranges (v_i^{max} - v_i^{min}) found using FVA:

$$P_1 = (2, 3, 8)$$
$$P_2 = (3, 6, 8)$$
$$P_3 = (1, 7, 11)$$

Examining the largest range within each of the three pathways, both P_1 and P_2 have the smallest (among paths) maximum (within path) range of 8. Between P_1 and P_2, P_1 is less variable than P_2, as P_1 has the next

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCBB.2015.2394470, IEEE/ACM Transactions on Computational Biology and Bioinformatics

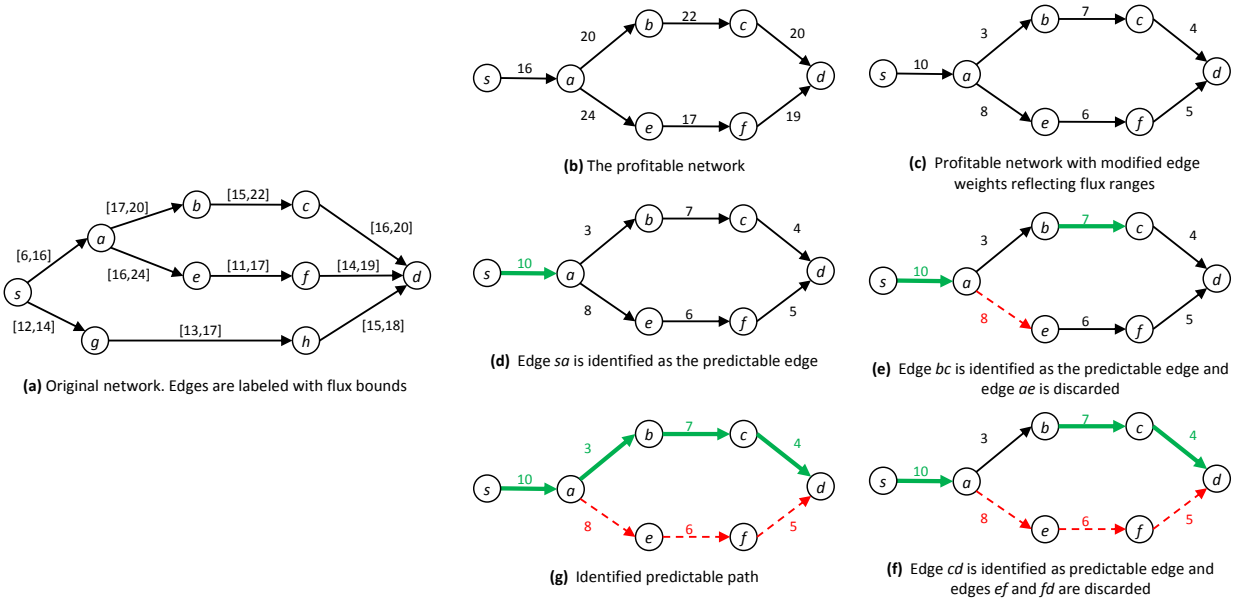


Fig. 2. Example network to illustrate the two searches of the *PreProPath* Algorithm. The green arrows indicate edges that are added to *PreditableProftiableGraph*. The dotted red arrows indicate edges that are removed from *PredictableGraph*.

smallest maximum range (value 3), thus satisfying Condition 2. Among all four paths, P_1 is profitable (capable of carrying flux above the *bottleneckWeight*) and the most predictable because it exhibits the least variability when compared to other profitable paths.

When analyzing a network graph without explicit path enumeration, identifying the least variable path, one with the smallest (among paths) maximum (within path) weight, is not straightforward. A naïve approach is to successively select the lowest edge weight until a path is found from source to destination. Consider for example network in Fig. 2(b), with source s and destination d. Such a naïve approach will result in considering the edges in the following weight order: 3, 4, 5, 6, 7, 8, and then 10. At that point, there is a path from s to d, but it encompasses multiple paths, and in this case, the entire network is selected. Our algorithm, *PreProPath*, selects the edges of the profitable path successively, first selecting the edge with the largest-weight edge necessary to complete the path from s to d, and then selecting the edge with the next largest weight, and so on.

3 *PreProPath* Algorithm

3.1 Algorithm

The *PreProPath* algorithm identifies a predictably profitable path from source to destination by executing two consecutive searches on the network graph. The first search identifies a profitable graph, a subset of the original graph G in which every reaction can operate at or above the flux limit, *bottleneckWeight*. The profitable graph can be found by removing from G all edges having weight less than *bottleneckWeight*. Every path from s to d in the profitable graph thus meets **Condition 1**.

In the second search, the objective is to identify the least variable path in the profitable graph. Edge weights in the profitable graph are set to flux ranges as calculated using FVA. Edges are selected successively (through multiple passes) to build the predictably profitable path. During each pass and in order of increasing edge weights, edges in profitable graph are selected for building a path from s to d. The passes stop when a path from s to d can be established using the selected edges. A post-processing step allows the identification of the path that meets both **Condition 1** and **Condition 2**.

The pseudo code for the *PreProPath* algorithm is presented in Fig. 3. On line 1, a weighted graph G is created from the **S** matrix. The edge weights are set to either v_{max} or v_{min} depending on the metabolic engineering application and the appropriateness of utilizing an optimistic or conservative approach. On line 2, the *bottleneckWeight* is found in G using the single source-single destination bottleneck algorithm (e.g. [23]).

The first search spans lines 3 through 7. Starting with an empty graph *profitableGraph*, edges with weight equal to or greater than *bottleneckWeight* are added to *profitableGraph*. Each edge in *profitableGraph* is then assigned the flux range as a weight in preparation for the second search.

The second search spans lines 8 through 13. An empty graph *predictableProfitableGraph* is created. The maximum connecting edge, predictable, in *profitableGraph* is iteratively selected and re-

***PreProPath* Algorithm(S, s, d, v_{min}, v_{max})**
1. Create a graph G from **S** using either v_{max} or v_{min} as edge weights
2. Identify *bottleneckWeight* for paths from s to d
3. Create an empty graph *profitableGraph*
4. for each edge e in G
5. if(weight(e) $\geq$ *bottleneckWeight*)
6. add edge e to *profitableGraph*
7. for each edge e, set edge weights in *profitableGraph* to be ($v_{max}[e] - v_{min}[e]$)
8. Create an empty graph *predictableProfitableGraph*
9. while there does not exist a path from s to d in *predictableProfitableGraph*
10. $predictable$ = maximum connecting edge in *profitableGraph* from s to d
11. remove $predictable$ and all edges with weight greater than weight of $predictable$ from *profitableGraph*
12. add $predictable$ to *predictableProfitableGraph*
13. return path from s to d in *predictableProfitableGraph*

Fig. 3. Pseudo code for *PreProPath* Algorithm.

moved from *profitableGraph*. This maximum connecting edge is found by first sorting all edge weights in *profitableGraph*, and then successively adding edges in increasing weight to an initially empty graph until a path from s to d is found. The last added edge is the maximum connecting edge. All edges with weights greater than the weight of the maximum connecting edge are removed from *profitableGraph*. The predictable edge is then added to *predictableProfitableGraph*. The process is repeated until a path from s to d is found in *predictableProfitableGraph*. The iterative process successively builds the predictable profitable path in *predictableProfitableGraph*, one edge at a time in order of decreasing variability. The returned path on line 13 is the *predictableProfitablePath* from s to d.

An example illustrating the two searches of the *PreProPath* algorithm is shown in Fig. 2. The original network, with each edge weight representing the flux bounds, is shown in Fig. 2(a). The first search identifies a profitable network. In this example, we utilize v_{max} as our edge weights for the first search.

By inspection, it can be seen that there are three parallel paths from source node s to target node d with the following node sequences: (i) s, a, b, c, d; (ii) s, a, e, f, d; and (iii) s, g, h, d. The bottleneck edge for (i) is sa with weight 16. This edge is also the bottleneck for path (ii). For path (iii), sg is the bottleneck edge. For the entire network, the bottleneck edge is sa, because this edge has a greater weight than sg. Edge sa is identified on line 2 in the algorithm as the bottleneck edge.

Per lines 4-6 of the algorithm, edges with weight greater than the weight of sa will be added to *profitableGraph*. Edges gh and hd do not appear in *profitableGraph*, as they have weights less than sa. The *profitableGraph* is shown in Fig. 2(b). Per line 7 of the algorithm, the edge weights in *profitableGraph* are modified to reflect flux ranges, shown in Fig. 2(c).

Fig. 2(d-g) illustrates the execution of the second search (lines 8-13 in the algorithm) on the graph in Fig. 2(c). First, the edge with weight 10, colored green in Fig. 2(d), is identified as the maximum connecting edge. To identify this edge, the algorithm examines the edges in order of ascending weights. Edges ab (weight 3), cd (4), ... ae (8), and sa (10) are considered in order. Once sa is considered, there exists a path from s to d, and sa is selected as a potential edge for the predictable profitable path.

Edge sa is removed from *profitableGraph* and added to the *predictableProfitableGraph*. Next, the edge with weight 7, colored green in Fig. 2(e), is identified as the maximum connecting edge. This edge is removed from *profitableGraph* and added to the *predictableProfitableGraph*. Additionally, edge ae, indicated as a dotted red line in Fig. 2(e), is removed from *profitableGraph*. This edge is removed because the algorithm cannot possibly identify ae as a maximum connecting edge in future iterations of the while loop. The *profitableGraph* is shown in Fig. 2(f) and Fig. 2(g) after two additional iterations through the while loop. After the second search terminates, there is only one path (colored green) from s to d. This is the predictable profitable path.

3.2 Correctness Proof

Theorem 1: Given a stoichiometric matrix, minimum and maximum flux ranges, and a starting vertex s and an ending vertex d, *PreProPath* returns a path from s to d. Further, *PreProPath* returns a predictable profitable path.

Proof: First, we argue that there exists a path in *profitableGraph* by contradiction. At the end of the first search, *PreProPath* identifies a subgraph of G, *profitableGraph*, in which every edge has a weight greater than or equal to the weight of the bottleneck edge. Assume that no path exists from s to d. This implies that there is an edge e' on the path from s to d with a weight less than the weight of the bottleneck edge, and that this edge was not added to the *profitableGraph*. By definition of the *bottleneckWeight*,

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCBB.2015.2394470, IEEE/ACM Transactions on Computational Biology and Bioinformatics

e' should have been identified as the bottleneck edge, and we have a contradiction. Thus, there exists a path from s to d in *profitableGraph*.

We next argue that *PreProPath* identifies a path in *predictableProfitableGraph* from s to d. This can be shown by construction. Every edge in *profitableGraph* is potentially an edge in *predictableProfitableGraph*. During each step of the second search, a new edge is added to *predictableProfitableGraph*. The steps repeat until a path from s to d is identified in *predictableProfitableGraph*. Because *profitableGraph* has a path from s to d, the second search is guaranteed to identify a path from s to d.

Further, we argue that the path identified by the second search is a predictable profitable path. By Condition 1, *profitableGraph* has edges with weights greater than the *bottleneckWeight*. Every path in *profitableGraph* is therefore a profitable path. By construction, every edge in the *profitableGraph* is a potential edge in the *predictableProfitableGraph*. Thus any path identified in the second search must be profitable.

By contradiction, we next argue that the second search returns a predictable path. Assume that the path p returned from the second search is not predictable. That is, there exists another path p' in *profitableGraph* that is less variable than p. By definition, p' has an edge e' with a maximum edge weight that is less than the weight of the maximum-weight edge e along p. If both p' and p have equal maximum edge weights, then p' has at least one edge e' with less weight than the weight of an edge e in p. This implies that the second search added an edge e' that completes a path from s to d to *predictableProfitableGraph* with smaller weight than edge e. This contradicts how the second search is executed. Thus, *PreProPath* is guaranteed to return a path that is predictable and profitable. □

4 Results

We applied the *PreProPath* algorithm to two test cases involving a small- and a large-scale metabolic model with 61 and 2,382 reactions, respectively. When computing the minimum flux, maximum flux, and flux ranges, we included cofactor and currency metabolites in the stoichiometric balances. However, when applying the *PreProPath* algorithm on the graph representation of the metabolic model, we removed the nodes corresponding to cofactors. This allows *PreProPath* to identify conversion routes from s to d that carry the carbon flux, as suggested by earlier work [24], [25]. With the removal of the cofactors, the two test cases had 51 and 2267 reactions, respectively.

4.1 Ethanol Production in *E. coli*

The first test case examined the production of ethanol from glucose in *E. coli*. As a first-generation biofuel, ethanol garnered significant attention from the metabolic engineering research community [26]. The network model used in the case study was constructed to represent *E. coli* growing on a minimal medium with glucose as the sole carbon and energy source [27]. The model comprised the following metabolic pathways: glycolysis, pentose phosphate pathway, TCA cycle, anapleurotic reactions, redox-associated reactions, oxidative phosphorylation/maintenance reactions, membrane transport reactions, and biomass synthesis. This model was used to calculate the metabolic flux distributions in *E. coli* at several steady states (Appendix A). For each reaction in the model, a lower and upper bound was calculated by minimizing and maximizing the flux subject to stoichiometric and measurement constraints. Experimental data for glucose uptake and growth rates were taken from published studies involving chemostat cultures [28]. The growth rates corresponded to the following approximate doubling times: 200, 100, 80, 60, 50, 40 and 30 min. Overall, the relative magnitudes of flux ranges with respect to glucose uptake were similar across different growth rates. The reactions with higher maximal flux values were found in glycolysis and pathways producing formate, ethanol and acetate as shown in Fig. 4. At all growth rates, the largest flux ranges were calculated for conversion of phosphoenolpyruvic acid (PEP) to pyruvate (PYR), followed by fructose-6-phosphate (F6P) to fructose-1,6-biphosphate (F16P), and malate (MAL) to oxaloacetate (OAA).

The flux data corresponding to different growth rates were at first separately analyzed. We used the maximum reaction flux (upper bound) as the edge weight to determine the flux-limiting reaction. Reactions with weights smaller than the flux-limiting step were pruned to isolate the profitable network. For every growth rate, the profitable network comprised glycolysis reactions (Fig. 5). As there were no competing pathways in the profitable network, the subsequent search for the predictable pathway led to the trivial result.

We also investigated a pessimistic approach of us-

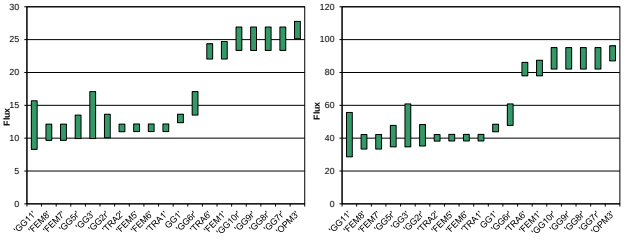


Fig. 4. Flux distributions (mmol/gDW.hr) in *E. coli* network for different growth rates. Flux distributions are plotted for growth rates of (A) 200 and (B) 50 minutes. Reactions are arranged in ascending order of flux lower bound.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCBB.2015.2394470, IEEE/ACM Transactions on Computational Biology and Bioinformatics

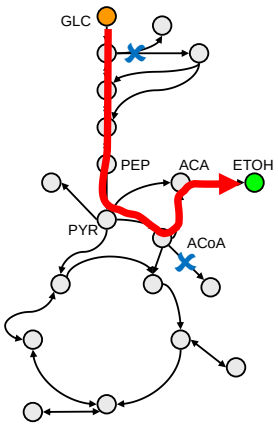


Fig. 5. *E. coli* network for ethanol production. Upper bounds of flux range are used as weights for the identification of profitable network. Red lines highlight pathways in the profitable network. The competing pathways deleted by Trinh et al. [1] are marked with X.

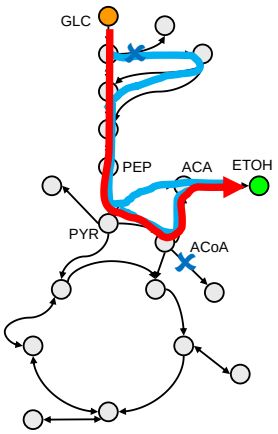


Fig. 6. *E. coli* network for ethanol production. Lower bounds of flux range are used as weights for the identification of profitable network. Red and blue lines highlight competing pathways in the profitable network for the production of ethanol from glucose. The red pathway is more predictable when compared to blue. The competing pathways deleted by Trinh et al. [1] are marked with X.

ing the minimal reaction flux (lower bound) as the edge weight to identify the profitable network. For every growth rate, the resulting profitable network comprised glycolysis and the pentose phosphate pathways (Fig. 6). The subsequent search for the predictable pathway based on flux ranges eliminated the pentose phosphate pathway, which contained reactions with larger flux ranges compared to glycolysis.

To determine if there was a predictably profitable consensus pathway across different growth rates, the above analysis was repeated using pooled data. For each reaction, the lower and upper flux bounds were

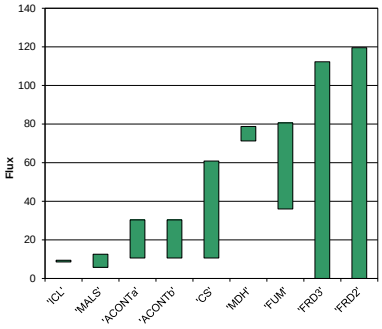


Fig. 7. Flux distributions (mmol/gDW.hr) of reactions in TCA cycle of *E. coli* network. Reactions are arranged in ascending order of flux upper bound.

set to the minimum of the lower bounds and maximum of the upper flux bounds respectively, irrespective of the growth rate. As was the case for each of the different growth rates, glycolysis is more predictably profitable compared to the pentose phosphate pathway.

4.2 Succinate Production in *E. coli*

In the second test case, we analyzed succinate production from glucose using a genome-scale model of *E. coli* metabolism (*i*AF1260) [29]. Succinate is a commercially valuable chemical used as a precursor for numerous industrial products, including pharmaceuticals and biodegradable polymers [30].

The upper and lower bounds of the reaction fluxes in the model were calculated by constraining a subset of internal and external fluxes using previously reported measurements for the MG1655 strain of *E. coli* assuming an error range of ± 5 percent on the measured fluxes [31]. The upper bounds of ethanol transport reactions were reduced, similar to a previous study [32]. Based on these flux ranges, the profitable network comprised the reactions of glycolysis and the TCA cycle. In this network, the flux ranges of reactions in the reductive arm of the TCA cycle, involving the conversion of oxaloacetate (OAA) to malate, fumarate, and eventually to succinate, were smaller than the flux ranges of the remaining reactions in the TCA cycle (Fig. 7). Consequently, the most predictably profitable synthesis route consisted of 14 reactions spanning the reactions of glycolysis and the reductive arm of the TCA cycle (Fig. 8).

4.3 Increasing the Flux through the Profitable Pathway

To evaluate the predictably profitable pathway identified by our algorithm as a target for succinate overproduction, we investigated the impact of overexpressing one or more enzymes in the pathway. Similar to a previous study [33], we calculated the smallest level of guaranteed succinate production

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCBB.2015.2394470, IEEE/ACM Transactions on Computational Biology and Bioinformatics

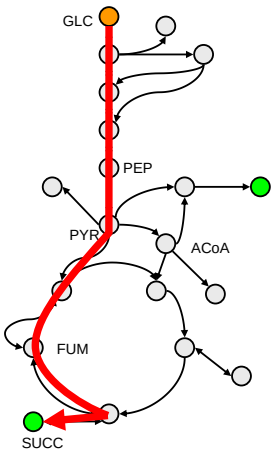


Fig. 8. *E. coli* network for succinate production. Red lines highlight pathways for the production of succinate from glucose.

by solving a linear program whose objective is to minimize the succinate flux. Over-expression of an enzyme was modeled by raising the lower bound of the corresponding reaction flux. Flux ranges were computed using FVA. Glucose uptake was set to a nominal value of 100 mmol/gDW.hr and allowed to vary ± 5 percent. All flux ranges computed by FVA are provided in Appendix B. We perform two sets of experiments where the lower bound of the biomass flux was set to 1 percent and to 5 percent of the wild-type (*i*AF1260) value (3 mmol/gDW.hr).

We first investigated the impact of over-expressing a single enzyme, i.e. increasing the lower bound of a reaction flux. An increase in the minimal succinate flux was found for three enzymes in the profitable pathway; these were enolase (ENO), fumarate reductase (FRD3), and phosphoenol pyruvate carboxylase (PPC) (Table 1). The magnitude of the minimal succinate flux depended on the enzyme and varied with the amount of increase in the lower bound (Fig. 9 and Fig. 10). However, the lower bound could not be increased without limit. For all three enzymes, a threshold was found beyond which the linear program became infeasible. The widest range of feasible solutions was found for FRD3 (Table 1). Over-expressing FRD3 also afforded the highest minimal succinate flux. Near the upper limit of the lower bound for FRD3, the minimal succinate flux reached 99 percent and 96 percent of the theoretical maximum for 1 percent and 5 percent of the wild-type biomass flux, respectively. Over-expressing PPC resulted in the lowest minimal succinate flux. However, even intervention yielded significant increases in succinate flux, above 48 percent and 44 percent of the theoretical maximum, when the lower bounds for the biomass flux were set to 1 percent and 5 percent of the wild-type flux, respectively.

We next investigated whether over-expressing an enzyme, one at a time, outside of the profitable pathway could also lead to an increase in the minimal succinate flux. The only enzyme over-expressions able to produce succinate at a level similar to that obtained by over-expressing enzymes in the predictably profitable pathway were oxidative phosphorylation reactions acting as cellular transport reactions.

Finally, we characterized the impact of over-expressing pairs of enzymes (Fig. 11 and Fig. 12). The calculations were performed exhaustively, since the predictably profitable pathway consisted of only 10 reactions, excluding exchange reactions. Approximately half of the 45 unique combinations (55 percent) resulted in a non-zero minimal flux of succinate. The best combinations (supporting a minimal succinate flux exceeding 75 percent of the theoretical maximum) involved at least one of the three enzymes identified from the single over-expression analysis (ENO, FRD3, and PPC). In addition, fumarase (FUM) was also identified as an attractive engineering target, specifically in combination with FRD3 or PPC for 1 percent biomass production. In these cases, the main contribution of FUM was to enlarge the effective over-expression range of the other enzyme. For example, when the lower bound of FUM flux is placed within the range of the wild type, only a narrow range is available for PPC over-expression to achieve a higher minimal succinate flux (exceeding 75 percent of the theoretical maximum). In contrast, reducing FUMs lower bound to below 20 mmol/gDW.hr widens the PPC over-expression range four-fold (Fig. 11b). Similarly, the lower bound of PPC flux when reduced below

TABLE 1
Flux Ranges of Enzymes Increases Minimum Succinate Yield. Units are mmol/gDW.hr.

Enzyme	Wild-type flux range	At least 1% biomass production				At least 5% biomass production			
		Max. succinate yield	Flux range for at least x% of max. theoretical yield of succinate			Max. succinate yield	Flux range for at least x% of max. theoretical yield of succinate		
			x = 33	x = 50	x = 75		x = 33	x = 50	x = 75
ENO	172 – 205.7	89	251 – 260	254 – 260	258 – 260	85	229 – 258	250 – 258	252 – 258
FRD3	0 – 112.3	99	94 – 210	124 – 210	168 – 210	96	96 – 209	126 – 209	117 – 209
PPC	81.5 – 88.2	48	335 – 356	–	–	44	327 – 344	–	–

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCBB.2015.2394470, IEEE/ACM Transactions on Computational Biology and Bioinformatics

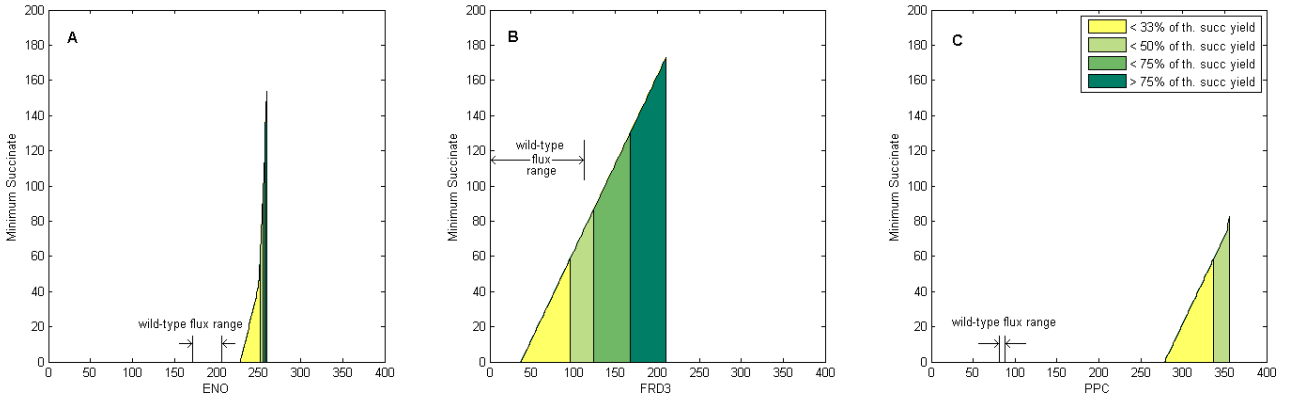


Fig. 9. Minimum guaranteed succinate flux for single intervention for at least 1% biomass production. Succinate flux (mmol/gDW.hr) is plotted against lower bound of reaction flux (mmol/gDW.hr).

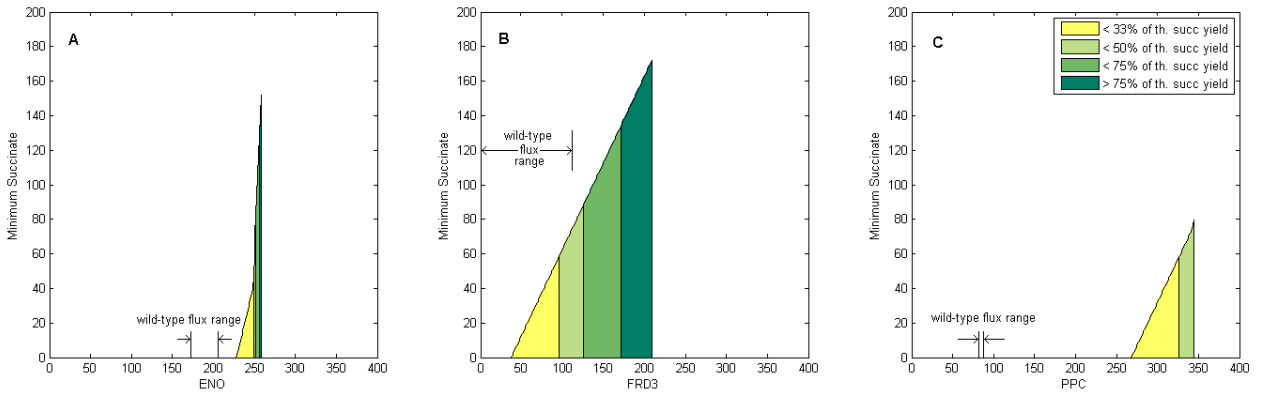


Fig. 10. Minimum guaranteed succinate flux for single intervention for at least 5% biomass production. Succinate flux (mmol/gDW.hr) is plotted against lower bound of reaction flux (mmol/gDW.hr).

150 mmol/gDW.hr, widens the ENO over-expression range five-fold (Fig. 11c). In all cases supporting a minimal succinate flux greater than 75 percent of the theoretical maximum, the increase in the minimal succinate flux positively correlated with an increase in the lower bound of ENO, FRD3 or PPC. In this regard, the double over-expressions did not identify any new enzymes for flux increase. Therefore, we did not further investigate additional combinations involving triple or quadruple over-expressions.

The engineering targets identified by our approach varied with different biomass production rates due to the different requirements for biomass precursors. For example, FUM was not identified as a potential enzyme for intervention when the lower bound for biomass flux was set to 5 percent of the wild-type flux. The over-expression ranges also depended on the growth rates. For higher growth rate, the over-expression level of the enzymes was lower compared to lower growth rate, reflecting a lower yield of succinate for a faster growing cell. For example, the flux range of PPC was found to be 335-356 mmol/gDW.hr for 1 percent biomass production compared to 327-344 mmol/gDw.hr for 5 percent biomass production.

5 Conclusion and Discussion

We have developed an efficient computational method to identify engineering targets for increased production of compounds in biochemical networks. The method is "uncertainty-aware" as it considers degrees of freedom in the model and multiple metabolic states arising because of different uptake rates. The algorithm is based on guided search and avoids exhaustive exploration of all pathways in the network. The effectiveness of the method was demonstrated by applying it to two test cases. In the first test case, the algorithm identified pathways for the maximum production of target compounds across different steady-state flux distributions reflecting different growth rates. In the second test case, we characterized the over-expression of enzymes along the succinate-producing pathway in *E. coli* that was identified by our algorithm as the predictable profitable path. An important feature of the *PreProPath* algorithm is that it can take into account different flux states, as determined by measured rates of metabolite exchange with

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCBB.2015.2394470, IEEE/ACM Transactions on Computational Biology and Bioinformatics

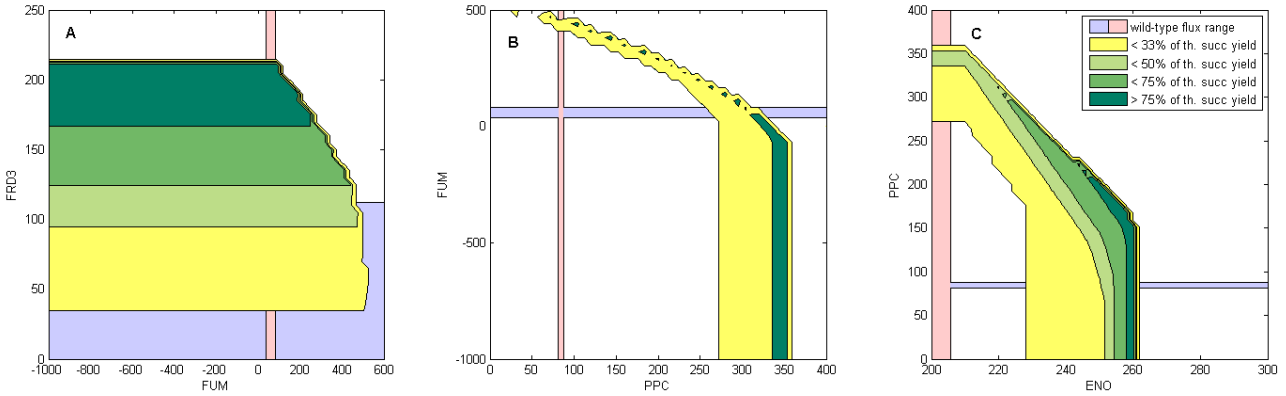


Fig. 11. Contour plot of minimum guaranteed succinate yield for at least 1% biomass production. Minimum succinate yield is plotted for lower bounds of reaction flux (mmol/gDW.hr) of the two intervened enzymes. The operating range of reaction fluxes in wild-type characterized strain is also shown.

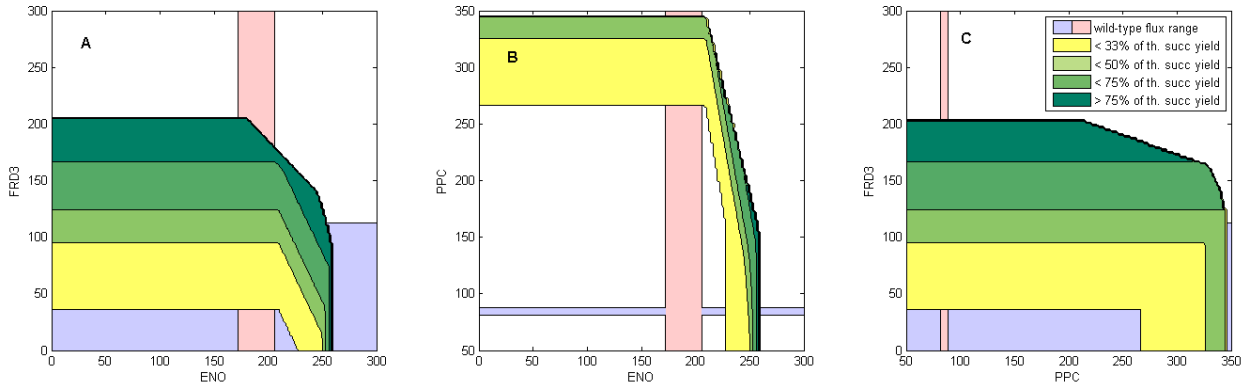


Fig. 12. Contour plot of minimum guaranteed succinate yield for at least 5% biomass production. Minimum succinate yield is plotted for lower bounds of reaction flux (mmol/gDW.hr) of the two intervened enzymes. The operating range of reaction fluxes in wild-type characterized strain is also shown.

the medium, when searching for pathways with particular attributes. In the first test case, *PreProPath* identified the same pathway, glycolysis, across different growth rates, underscoring the singular importance of this pathway in ethanol production.

PreProPath is effective in analyzing biochemical pathways without direct enumeration of all possible pathways. Our algorithm is an alternative to explicitly enumerating all elementary pathways followed by search for a pathway with a specific property (predictability and profitability, in this case). To increase ethanol production in *E. coli* as in our first test case, Trinh et al. identified and then analyzed over 15,000 EFMs to determine gene knockout targets [1]. To narrow down the candidate pathways, EFMs that do not contribute to ethanol production were eliminated. The remaining ethanol-producing EFMs were then grouped into six "families" based on the type of sugar substrate. Using eight gene knockouts, pathways competing with ethanol producing pathways were removed. Our identified predictably profitable pathway is the same one identified by Trinh et al. We have computed the number of EFMs using two tools, gEFM [34] and EFMTool [16]. The runtime for gEFM was smaller at 1,636 seconds. With network compression, the runtime for EFMTool was smaller at 54.68 seconds. The runtime for *PreProPath*, without network compression, was less than one second. All programs were executed on a 2.83GHz Intel Xeon E5440 CPU with a 12MB cache.

In our case studies, we looked at bottleneck reactions that refer to flux-limiting reactions. A reaction can be "flux-limiting" due to various reasons such as reaction kinetics or regulatory effects. Our method simply identifies such a reaction within the context of a specific network at a particular flux distribution. We draw a distinction between flux-limiting and rate-limiting. A flux-limiting reaction does not necessarily correspond to a "rate-limiting" reaction, which was believed to be the slowest step in a series of reactions, and was often associated with the first committed step of a pathway. Metabolic Control Analysis [33],

applicable only in the context of small network perturbations, shows that such flux-limiting reactions exist if the first reaction step is completely insensitive to its product, which is not typically the case.

The results of the case studies suggest that our algorithm can efficiently guide the search for pathway engineering targets. While the results are promising, they also pointed to limitations of the present analysis. First, the analysis does not distinguish between degrees of freedom in a model arising from insufficient equality constraints and variances associated with measurements. These two different sources of uncertainty can both lead to flux variability, which forms the basis of our algorithm. However, the relative magnitudes of these uncertainties directly influence the results. For example, the second case study showed that it is possible to obtain a different predictably profitable pathway depending on the metabolic state. Clearly, metabolic states can only be distinguished meaningfully, if the uncertainties in the measurements are sufficiently small. One way to discriminate between the variances arising from the two different sources of uncertainty is through sensitivity analysis, for example based on Monte Carlo simulations, which systematically assess the impact of measurement errors. Second, the analysis assumed that a reaction with a small value for its flux range is more profitable to genetically boost than another with a higher range value. Additionally, it is assumed that the flux values are uniformly distributed between the minimum and maximum flux values. The algorithm presented here can be adapted to utilize edge weights that correspond to a different desired objective. For example, if Monte Carlo sampling is utilized and flux distributions are available, then incorporating spreads in standard deviation of flux distributions in edge weightings may be more informative about predictability than flux ranges. Third, using FVA, the flux for each edge is maximized and minimized independently of the flux state of other edges constrained only by the stoichiometric balances and the physicochemical upper and lower bounds on the fluxes (e.g. reaction irreversibility). Clearly, not all reactions can attain their maximum or minimum flux values simultaneously. Our profitability and predictability analysis can be made more accurate by computing tighter flux bounds attained through additional constraints or utilizing a sampling based approach. The additional constraints could be derived from measurements on flux distributions at different metabolic states, for example, through isotopic labeling experiments. The sampling would respect dependencies and ensure consistent metabolic state throughout the network. The algorithm presented here can be adapted to utilize edge weights that correspond to a different desired objective. The Java implementation of the algorithm is available via GitHub at https://github.com/eullah01/PreProPath.

APPENDIX A

Flux data of *E. coli* for ethanol production.

APPENDIX B

Flux data of *E. coli* for succinate production.

ACKNOWLEDGMENTS

This work was supported by the National Science Foundation under Grant no. 0829899.

REFERENCES

[1] C. T. Trinh, P. Unrean, and F. Srienc, "Minimal *Escherichia coli* cell for the most efficient production of ethanol from hexoses and pentoses," *Appl Environ Microbiol*, vol. 74, no. 12, pp. 3634–43, 2008.

[2] J. H. Park and S. Y. Lee, "Towards systems metabolic engineering of microorganisms for amino acid production," *Curr Opin Biotechnol*, vol. 19, no. 5, pp. 454–60, 2008.

[3] S. K. Ng, D. I. Wang, and M. G. Yap, "Application of destabilizing sequences on selection marker for improved recombinant protein productivity in cho-dg44," *Metab Eng*, vol. 9, no. 3, pp. 304–16, 2007.

[4] S. Atsumi, A. F. Cann, M. R. Connor, C. R. Shen, K. M. Smith, M. P. Brynildsen, K. J. Chou, T. Hanai, and J. C. Liao, "Metabolic engineering of *Escherichia coli* for 1-butanol production," *Metab Eng*, vol. 10, no. 6, pp. 305–11, 2008.

[5] T. Hanai, S. Atsumi, and J. C. Liao, "Engineered synthetic pathway for isopropanol production in *Escherichia coli*," *Appl Environ Microbiol*, vol. 73, no. 24, pp. 7814–8, 2007.

[6] M. Inui, M. Suda, S. Kimura, K. Yasuda, H. Suzuki, H. Toda, S. Yamamoto, S. Okino, N. Suzuki, and H. Yukawa, "Expression of *Clostridium acetobutylicum* butanol synthetic genes in *Escherichia coli*," *Appl Microbiol Biotechnol*, vol. 77, no. 6, pp. 1305–16, 2008.

[7] L. P. Yomano, S. W. York, K. T. Shanmugam, and L. O. Ingram, "Deletion of methylglyoxal synthase gene (mgsa) increased sugar co-metabolism in ethanol-producing *Escherichia coli*," *Biotechnol Lett*, vol. 31, no. 9, pp. 1389–98, 2009.

[8] E. J. Steen, Y. Kang, G. Bokinsky, Z. Hu, A. Schirmer, A. McClure, S. B. Del Cardayre, and J. D. Keasling, "Microbial production of fatty-acid-derived fuels and chemicals from plant biomass," *Nature*, vol. 463, no. 7280, pp. 559–62, 2010.

[9] A. F. Cann and J. C. Liao, "Production of 2-methyl-1-butanol in engineered *Escherichia coli*," *Appl Microbiol Biotechnol*, vol. 81, no. 1, pp. 89–98, 2008.

[10] P. Unrean, C. T. Trinh, and F. Srienc, "Rational design and construction of an efficient *E coli* for production of diapolycopendioic acid," *Metab Eng*, vol. 12, no. 2, pp. 112–22, 2010.

[11] J. C. Liao, S. Y. Hou, and Y. P. Chao, "Pathway analysis, engineering, and physiological considerations for redirecting central metabolism," *Biotechnol Bioeng*, vol. 52, no. 1, pp. 129–40, 1996.

[12] S. Schuster, T. Dandekar, and D. A. Fell, "Detection of elementary flux modes in biochemical networks: a promising tool for pathway analysis and metabolic engineering," *Trends Biotechnol*, vol. 17, no. 2, pp. 53–60, 1999.

[13] H. Kurata, Q. Zhao, R. Okuda, and K. Shimizu, "Integration of enzyme activities into metabolic flux distributions by elementary mode analysis," *BMC Syst Biol*, vol. 1, p. 31, 2007.

[14] S. Schuster, D. A. Fell, and T. Dandekar, "A general definition of metabolic pathways useful for systematic organization and analysis of complex metabolic networks," *Nat Biotechnol*, vol. 18, no. 3, pp. 326–32, 2000.

[15] S. Klamt and J. Stelling, "Combinatorial complexity of pathway analysis in metabolic networks," *Mol Biol Rep*, vol. 29, no. 1-2, pp. 233–6, 2002.

[16] M. Terzer and J. Stelling, "Large-scale computation of elementary flux modes with bit pattern trees," *Bioinformatics*, vol. 24, no. 19, pp. 2229–35, 2008.

[17] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 2nd ed. Cambridge: The MIT Press, 2001.
[18] E. Ullah, K. Lee, and S. Hassoun, "A weighted graph algorithm for identifying dominant-edge metabolic pathways," *International Conference on Computer-Aided Design (ICCAD)*, pp. 144–150, 2009.
[19] D. A. Fell and J. R. Small, "Fat synthesis in adipose tissue. an examination of stoichiometric constraints," *Biochem J*, vol. 238, no. 3, pp. 781–6, 1986.
[20] R. Mahadevan and C. H. Schilling, "The effects of alternate optimal solutions in constraint-based genome-scale metabolic models," *Metab Eng*, vol. 5, no. 4, pp. 264–76, 2003.
[21] H. P. J. Bonarius, G. Schmid, and J. Tramper, "Flux analysis of underdetermined metabolic networks: the quest for the missing constraints," *Trends in Biotechnology*, vol. 15, no. 8, pp. 308–314, 1997.
[22] A. Varma and B. O. Palsson, "Metabolic flux balancing: Basic concepts, scientific and practical use," *Nat Biotech*, vol. 12, no. 10, pp. 994–998, 1994.
[23] H. N. Gabow and R. E. Tarjan, "Algorithms for two bottleneck optimization problems," *Journal of Algorithms*, vol. 9, no. 3, pp. 411–417, 1988.
[24] D. Croes, F. Couche, S. J. Wodak, and J. van Helden, "Metabolic pathfinding: inferring relevant pathways in biochemical networks," *Nucleic Acids Res*, vol. 33, no. Web Server issue, pp. W326–30, 2005.
[25] P. Gerlee, L. Lizana, and K. Sneppen, "Pathway identification by network pruning in the metabolic network of *Escherichia coli*," *Bioinformatics*, vol. 25, no. 24, pp. 3282–8, 2009.
[26] J. M. Clomburg and R. Gonzalez, "Biofuel production in *Escherichia coli*: the role of metabolic engineering and synthetic biology," *Appl Microbiol Biotechnol*, vol. 86, no. 2, pp. 419–34, 2010.
[27] R. Carlson and F. Srienc, "Fundamental *Escherichia coli* biochemical pathways for biomass and energy production: identification of reactions," *Biotechnol Bioeng*, vol. 85, no. 1, pp. 1–19, 2004.
[28] ——, "Fundamental *Escherichia coli* biochemical pathways for biomass and energy production: creation of overall flux states," *Biotechnol Bioeng*, vol. 86, no. 2, pp. 149–62, 2004.
[29] A. M. Feist, C. S. Henry, J. L. Reed, M. Krummenacker, A. R. Joyce, P. D. Karp, L. J. Broadbelt, V. Hatzimanikatis, and B. O. Palsson, "A genome-scale metabolic reconstruction for *Escherichia coli* K-12 MG1655 that accounts for 1260 ORFs and thermodynamic information," *Mol Syst Biol*, vol. 3, p. 121, 2007.
[30] S. H. Hong and S. Y. Lee, "Importance of redox balance on the production of succinic acid by metabolically engineered *Escherichia coli*," *Appl Microbiol Biotechnol*, vol. 58, no. 3, pp. 286–90, 2002.
[31] A. M. Sanchez, G. N. Bennett, and K. Y. San, "Batch culture characterization and metabolic flux analysis of succinate-producing *Escherichia coli* strains," *Metab Eng*, vol. 8, no. 3, pp. 209–26, 2006.
[32] S. Ranganathan, P. F. Suthers, and C. D. Maranas, "Optforce: an optimization procedure for identifying all genetic manipulations leading to targeted overproductions," *PLoS Comput Biol*, vol. 6, no. 4, p. e1000744, 2010.
[33] D. A. Fell, "Metabolic control analysis: a survey of its theoretical and experimental development," *Biochem J*, vol. 286 (Pt 2), pp. 313–30, 1992.
[34] E. Ullah, C. Hopkins, S. Aeron, and S. Hassoun, "Decomposing biochemical networks into elementary flux modes using graph traversal," in *Proceedings of the International Conference on Bioinformatics, Computational Biology and Biomedical Informatics*, ser. BCB'13. New York, NY, USA: ACM, 2013, pp. 211:211–211:218. [Online]. Available: http://doi.acm.org/10.1145/2506583.2506620

Ehsan Ullah received his PhD in Computer Science from Tufts University and Masters Degree in Electrical Engineering from University of Engineering and Technology, Lahore, Pakistan. His research interests computational tools and algorithms for Systems Biology. He is a member of ACM, IET, IEEE and IEEE Computer Society.

Mark Walker received his BS degree in chemical engineering from Tufts University in 2010. He is currently working towards a PhD in biomedical engineering under Dr. Raimond Winslow at the Institute for Computational Medicine, Johns Hopkins University. His research interests include multi-scale modeling of biological systems and high performance computing. He is also interested in developing web-based tools that enable transparent computational research in the biomedical sciences.

Kyonbum Lee is Associate Professor and Chair of the Chemical and Biological Engineering Department at Tufts University. He obtained his B.S. in chemical engineering from Stanford University (1995) and Ph.D. in chemical engineering from the Massachusetts Institute of Technology (2002). His research interests are in metabolic engineering, tissue engineering, and systems biology. A major thrust area is the development and use of metabolomics technologies for the study of metabolic diseases.

Soha Hassoun received the PhD degree from the Department of Computer Science and Engineering from the University of Washington, Seattle, and the Masters Degree from the Department of Electrical Engineering and Computer Science from the Massachusetts Institute of Technology. She is an associate professor and chair of the Department of Computer Science at Tufts University. Her research interests electronic design automation and computational tools for Systems Biology. Soha was a recipient of an NSF CAREER Award and ACM/SIGDA Distinguished Service Awards for creating the Ph.D. forum at Design Automation Conference (DAC), and the CADAthlon at ICCAD. She serves and has served on a number of technical and executive committees for several conferences and workshops including serving as the technical program chair for the Design Automation Conference (DAC) in 2012 and 2011, and the International Conference on Computer-Aided Design (ICCAD) in 2005. She is a Tau Beta Pi fellow. She is a member of ACM, AiChE, and a senior member of IEEE.