Bioinformatics, 32(3), 2016, 398–408
doi: 10.1093/bioinformatics/btv578
Advance Access Publication Date: 10 October 2015
Original Paper

Systems biology

FogLight: an efficient matrix-based approach to construct metabolic pathways by search space reduction

Mehrshad Khosraviani[1], **Morteza Saheb Zamani**[1,*] **and Gholamreza Bidkhori**[2,*,†]

[1]Department of Computer Engineering & IT, Amirkabir University of Technology, Tehran, Iran and [2]Bioinformatics Section, The Lister Institute of Microbiology, Tehran, Iran

*To whom correspondence should be addressed.
†He is currently with VTT Technical Research Centre of Finland.

Associate Editor: Alfonso Valencia

Received on February 9, 2015; revised on September 15, 2015; accepted on October 2, 2015

Abstract

Motivation: A fundamental computational problem in the area of metabolic engineering is finding metabolic pathways between a pair of source and target metabolites efficiently. We present an approach, namely FogLight, for searching metabolic networks utilizing Boolean (AND-OR) operations represented in matrix notation to efficiently reduce the search space. This enables the enumeration of all pathways between metabolites that are too distant for the application of brute-force methods.
Results: Benchmarking tests run with FogLight show that it can reduce the search space by up to 98%, after which the accelerated search for high accurate results is guaranteed. Using FogLight, several pathways between eight given pairs of metabolites are found of which the pathways from CO_2 to ethanol are specifically discussed. Additionally, in comparison with three path-finding tools, namely PHT, FMM and RouteSearch, FogLight can find shorter and more pathways for attempted source-target metabolite pairs.
Contact: szamani@aut.ac.ir, gholamreza.bidkhori@vtt.fi
Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

As time passes on, our knowledge of metabolism is being developed and metabolic databases such as KEGG (Kanehisa *et al.*, 2014), MetaCyc (Caspi *et al.*, 2014), and similar databases are evolving as this information advances. However, our insight is still incomplete and the gaps are being filled gradually.

Metabolic engineering, emerged in the early 1990s, is defined as the directed modulation of metabolic pathways using methods of recombinant technology for the purpose of (over)production or accelerated production of fuels, chemical and pharmaceutical products (Bailey, 1991). Metabolic pathways are the roadmaps in which a biomolecule, namely metabolite, undergoes possible biotransformations, either under the action of enzymatic catalysis or by spontaneous reactions. The problem of synthesizing these roadmaps is more complex than the one referring to a tree-structured search network. Hypergraphs are better representations for metabolic networks. Since finding *k*-shortest hyperpaths of a hypergraph is an NP-complete problem (Ausiello *et al.*, 1992), the problem should be solved heuristically.

Research on metabolic pathways is done in two complementary categories. The analysis of metabolic pathways is motivated by the rapidly increasing quantity of available information on the pathways. Some researchers try to analyze the pathways with certain properties, like elementary flux modes (Schuster and Hilgetag, 1994) and thermodynamic feasibility (Ullah *et al.*, 2009). On the other hand, some algorithms have been proposed to find or predict possible pathways in order to convert a given source metabolite to a given target metabolite. Some of these algorithms are based on one of the branches of artificial intelligence (such as logic programming (Darvas, 1988), evolutionary algorithms (Gerard *et al.*, 2013), machine learning (Dale *et al.*, 2010; Karp and Mavrovouniotis, 1994), etc.), some algorithms have been proposed to find or predict possible pathways in order to convert a given source metabolite to a given target metabolite.

Küffner and his coauthors in (Kuffner *et al.*, 2000) applied an informed searching method, which is similar to (Mavrovouniotis, 1993), on Petri nets derived from metabolic databases to find and enumerate all valid pathways satisfying additional user-defined constraints. Petri nets, directed bipartite graphs and state-transition graphs are the main tools to model metabolic networks and apply some informed and uninformed approaches to them (Croes *et al.*, 2006; Lim and Wong, 2012; McShan *et al.*, 2003; Rahman *et al.*, 2005). In addition to these approaches, various mathematical methods have emerged in the post-genomic era to search for metabolic pathways. In the literature (Beasley and Planes, 2007; Burgard *et al.*, 2001; Jonnalagadda and Srinivasan, 2014; Pey *et al.*, 2011; Pharkya *et al.*, 2004), the authors detailed optimization models, based upon integer linear programming (ILP) to search for plausible metabolic pathways.

Algorithms incorporating concepts from retrosynthesis have been developed to search in the metabolic space, represented by hypergraphs for desired pathways (Campodonico *et al.*, 2014; Carbonell *et al.*, 2012; Cho *et al.*, 2010). These algorithms borrowed an idea from the allied field of synthetic chemistry in which reversed chemical transformations are iteratively applied starting from a target product to reach precursors that are endogenous to the chassis. In Carbonell *et al.* (2012), two methods, one based on elementary modes and the other based on a direct enumeration algorithm were presented. As other approaches using retrosynthesis model, novel metabolic routes have been efficiently screened by probabilistic selection of metabolic pathways in Rodrigo *et al.* (2008) and Yousofshahi *et al.* (2011).

A brute-force method is an exhaustive search approach which systematically enumerates all possible candidates for the exact solution. This is an effective approach in finding pathways but it cannot be used for large networks, as the execution time of the algorithm grows exponentially with the size of the network. One approach to ameliorate the exponential-time problem of the brute-force search is often to reduce the search space. Sometimes a heuristic method is used to obtain dramatic reduction of the candidates to all satisficing solutions and speedup the process of finding the proper pathway. However, on the contrary to brute-force, the heuristic approaches do not guarantee to find a pathway even if there is one. On the other hand, one can reduce the search space by first constructing a small sub-network without losing all candidates, and then applying the brute force method to find all the proper pathways. The approach proposed in this article follows this strategy using an analytical method.

This article presents an innovative solution to find all possible distinct sets of coherent enzyme-catalyzed biochemical reactions (i.e. metabolic pathways) through which the source compound is turned into the target compound. To this end, an analytical processing technique based on matrix operations helps us to provide a search space reduction strategy and find all desired metabolic pathways. Moreover, our proposed algorithm can even find the k-shortest paths by setting proper parameters.

Given that our proposed approach is based on matrix operations, the operational complexity of these operations depends on the matrix dimensions. While the use of bipartite graphs challenges the complexity in this context, we preferred to employ a more appropriate model instead of the bipartite graph. Indeed, using the directed bipartite graph causes our approach to bear large spatial and temporal costs (i.e. size of the memory and the amount of time required to search through it) which stem from intensive increase in the number of vertices and edges. For these reasons, we formulate the biochemical reactions contained in the metabolic pathway databases, e.g. KEGG, MetaCyc, by an AND/OR graph model.

And-Inverter Graph, Majority-Inverter Graph, and AND/OR Graph are three different internal network representations used to provide a suitable environment for modeling in a wide area of problem domains, such as logic optimization and synthesis (Amarú *et al.*, 2014; Färm *et al.*, 2005). Among these graph models, the use of AND/OR graph and the search through it were proposed earlier than the others (Bagchi and Mahanti, 1983). The AND/OR graph is a typical graph in which the types of relations between vertices, which can be either 'AND', 'OR' or a function of them, determine the types of searching process steps that should be followed through. The relations, i.e. 'AND' and 'OR', has been inspired by two simple Boolean gates (i.e. Boolean operations) with the same names, and finally, the graph can be conceived as a Boolean circuit.

Among the aforementioned models, AND/OR graphs can better represent metabolic reactions. Since the introduction of the AND/OR graph, it has been used in various applications and algorithms. To this end, we visualize the network of the biochemical reactions as the Boolean functions consisting of two-variable AND/OR operations. For example, in the metabolic reaction $m_1 + m_2 \rightarrow m_3$, the metabolite m_3 is produced if the metabolites m_1 and m_2 are both present. That is, the relation between the metabolites m_1, m_2 and m_3 is given by the Boolean function $m_3 = m_1$ AND m_2. Conversely, considering two reactions $m_1 \rightarrow m_3$ and $m_2 \rightarrow m_3$, the relation between the corresponding metabolites can be interpreted as the Boolean function $m_3 = m_1$ OR m_2, which means the production of m_3 depends on the existence of m_1 or m_2. Accordingly, before developing the proposed approach to find the pathways in a given metabolic network (Section 2.2), we first describe our AND/OR graph model and its properties against the conventional model in Section 2.1 and the proposed matrix representation in Section 2.2.1. Section 3 is devoted to results and discussion and finally, a conclusion is presented in Section 4.

2 Materials and Methods

2.1 Data model

All elementary biochemical reactions, in the sense that each one takes one basic step (association, dissociation or conversion) to complete, can be represented by one of the two equations of the form (1) or (2) which are used to represent irreversible and reversible reactions, respectively.

$$S_1 + S_2 + \cdots + S_m \rightarrow P_1 + P_2 + \cdots + P_n \tag{1}$$

$$S_1 + S_2 + \cdots + S_m \rightleftharpoons P_1 + P_2 + \cdots + P_n \tag{2}$$

In the above equations, S_i's and P_i's represent the substrates and the products in the biochemical reactions, respectively. Additionally, m and n show the number of metabolites participating as either substrates or products on both sides of the arrows in the equations.

The first and important step towards finding metabolic pathways is modeling all biochemical reactions and their participating metabolites in the metabolic databases by a computational data structure usable in algorithmic approaches. Obviously, an appropriate model to satisfy this need is a directed graph, but conventional directed graphs cannot represent the intrinsic properties of such a data set. Directed hypergraphs are alternatives to standard directed graphs to represent the facets of the database contents. Thus a model based on the hypergraph theory was suggested in (Pearcy *et al.*, 2014). A hypergraph is a generalization of a graph in which, in contrast to the standard one, its edges called hyperedges can be attached to a set of vertices and not only to two vertices.

Typically, the cofactors or the currency metabolites (Huss and Holme, 2007; Ma and Zeng, 2003; Wagner and Fell, 2001) of the reactions, i.e. the metabolites taking part into large number of reactions, e.g. NADPH, ATP, CO_2, play a part often in the metabolic reactions not as a main compound. The pathways containing the intermediate reaction steps, in which only these molecules are either produced or consumed, should not be considered when an algorithm is looking for some relevant pathways. In consequence, the shortest paths found by some previous algorithms (Latendresse *et al.*, 2014; Rahman *et al.*, 2005) do not correspond to relevant metabolic pathways. Since these molecules are not included explicitly in some pathway databases, we use this feature in our approach to avoid unnecessary complications. Additionally, instead of using the directed bipartite graph (today's conventional model which can be seen in some papers like (Beasley and Planes, 2007; Carbonell *et al.*, 2012; Kuffner *et al.*, 2000)), we formulate the directed hypergraph representing biochemical reactions of the metabolic pathway databases with a directed AND/OR graph model to reduce the space complexity of the search space (A sample AND/OR graph and its corresponding directed bipartite graph can be found in Supplementary Fig. S1). In the following two subsections, we describe our model by recalling some definitions from graph theory.

2.1.1 Definitions and notations (from graph theory)

In this subsection, three basic definitions are described. We use the first two definitions below to define directed AND/OR graphs.

Definition 1: A *labeled digraph* is a 4-tuple $G = (\mathbb{V},\ \mathbb{A},\ \mathbb{L}_{\mathbb{V}}, \mathbb{L}_{\mathbb{A}})$ in which

- $\mathbb{V}$ and $\mathbb{A} = \{(v_i,\ v_j)\ :\ v_i, v_j \in \mathbb{V}\text{and}\ (v_i,\ v_j) \neq (v_j,\ v_i)\}$ are nonempty sets of vertices and directed edges (also called arcs), respectively. The vertex $v_i \in \mathbb{V}$, called *initial vertex*, is the source vertex from which the arc $(v_i,\ v_j)$ starts while the vertex $v_j \in \mathbb{V}$, called *terminal vertex*, is the sink vertex to which the arc $(v_i,\ v_j)$ points.
- $\mathbb{L}_V$ and $\mathbb{L}_A$ describe nonempty sets of the unique identifiers for labeling the vertices and the arcs within G, respectively.

Definition 2: $G' = (\mathbb{V},\ \mathbb{A},\ \mathbb{L}_{\mathbb{V}}, \mathbb{L}_{\mathbb{A}})$ is known as a *labeled multidigraph* if it is permitted to have multiple arcs with the same source and the same sink.

Definition 3: A *directed AND/OR graph* is defined as a labeled multidigraph with no self-loops where some incoming (or outgoing) edges of the vertex $v_i \in \mathbb{V}$ may have identical and non-unique labels.

The labeling technique enables us to define a specific identity for the relationship between the set of edges directed towards/from each of the vertices of the multidigraph G'; this identity reflects a Boolean function using a combination of two main Boolean algebraic operations, namely the conjunction '*AND*' and the disjunction '*OR*' operations. In other words, the set of incoming vertices to vertex v_i with an identical edge label $\ell_e \in \mathbb{L}_A$ in a given directed AND/OR graph represents the members of a conjunction or the inputs to an 'AND' gate. On the other hand, the inputs to an 'OR' gate, with the output v_i, are provided by the set of vertices connected to the incoming edges with distinct labels.

2.1.2 Metabolic AND/OR graph

Here, we look at a modeling of metabolic networks via the directed AND/OR graph. Several remarks follow from the above definitions and comparison between the AND/OR graph and the bipartite graph, both representing metabolic networks.

As stated before, the directed bipartite is a graph with two types of vertices, namely biochemical reactions and metabolites. In contrast, the set of vertices $\mathbb{V}$ in our AND/OR graph model, i.e. $G' = (\mathbb{V},\ \mathbb{A},\ \mathbb{L}_{\mathbb{V}}, \mathbb{L}_{\mathbb{A}})$, includes only metabolites. Additionally, the biochemical reactions are placed in the set $\mathbb{A}$ of arcs of graph G'. Regarding the structure of these two models, the number of vertices, as well as edges, in the metabolic AND/OR graph is less than the ones in the bipartite graph (these claims are further discussed in Supplementary information). Furthermore, taking reversible reactions into account results in exponential growth of the total number of paths.

Finally, $\mathbb{L}_{\mathbb{A}}$ and $\mathbb{L}_{\mathbb{V}}$ are two disjoint nonempty sets of reaction and compound identifiers, respectively, used to label the arcs and vertices of the graph G'. In a database, for instance, each identifier is a 6-character label (e.g. **C00084** or **R00746**) starting with either the letter **C** for compounds (i.e. metabolites) or the letter **R** for reactions and ending with a unique 5-digit number. Considering the given labels of the vertices and the arcs in Figure 1A, which start with either the letter **C** or the letter **R**, the subscripts *a*, *b*, *f*, etc., denote some particular 5-digit numbers.

2.2 The proposed algorithmic approach

Despite the ability of being a general problem-solving technique, the exhaustive search technique can be impractical for large-scale problems due to the combinatorial nature of some problems, in particular, the problem of finding metabolic pathways, and their large demanding search-space. In order to control the complexity of searching through the metabolic networks by the brute-force search and speeding it up, we exploit a matrix-based approach to reduce the size of the search space.

Since matrix-based computation is an indispensable prerequisite for our algorithm, in the following, we suggest a matrix representation for an AND/OR graph used then by the algorithm described in Section 2.2.2.

2.2.1 Matrix representation of an AND/OR graph

In this section, we define a new matrix representation for directed AND/OR graphs. This representation consists of two square matrices as follows:

(1) An asymmetric *n*-by-*n* *binary matrix* $\mathcal{A}_{n\times n}$ corresponding to an AND/OR graph $G' = (\mathbb{V},\ \mathbb{A},\ \mathbb{L}_{\mathbb{V}}, \mathbb{L}_{\mathbb{A}})$ whose entry A_{ij} in row i and column j is defined as:

$$A_{ij} = \begin{cases} 1 & \text{if there is an arc from vertex } v_i \in \mathbb{V} \text{ to } v_j \in \mathbb{V} \text{ for } i \neq j \\ 0 & \text{otherwise} \end{cases}$$

(2) An asymmetric *n*-by-*n* *label matrix* $\mathcal{L}_{n\times n}$ used as a way of distinguishing between conjunction and disjunction relations as well as two different conjunction relations. The entry L_{ij} represents the type of relations, conjunction or disjunction, participating in the arcs $(v_i,\ v_j)$. In other words, each entry L_{ij} participating as an element of a logical conjunction is tagged with a label ℓ_e:

$$L_{ij} = \begin{cases} \ell_e & \text{if arc } e \text{ connected from } v_i \in \mathbb{V} \text{ to } v_j \in \mathbb{V} \text{ participates} \\ & \text{as an element of a conjunction relation} \\ NULL & \text{otherwise} \end{cases}$$

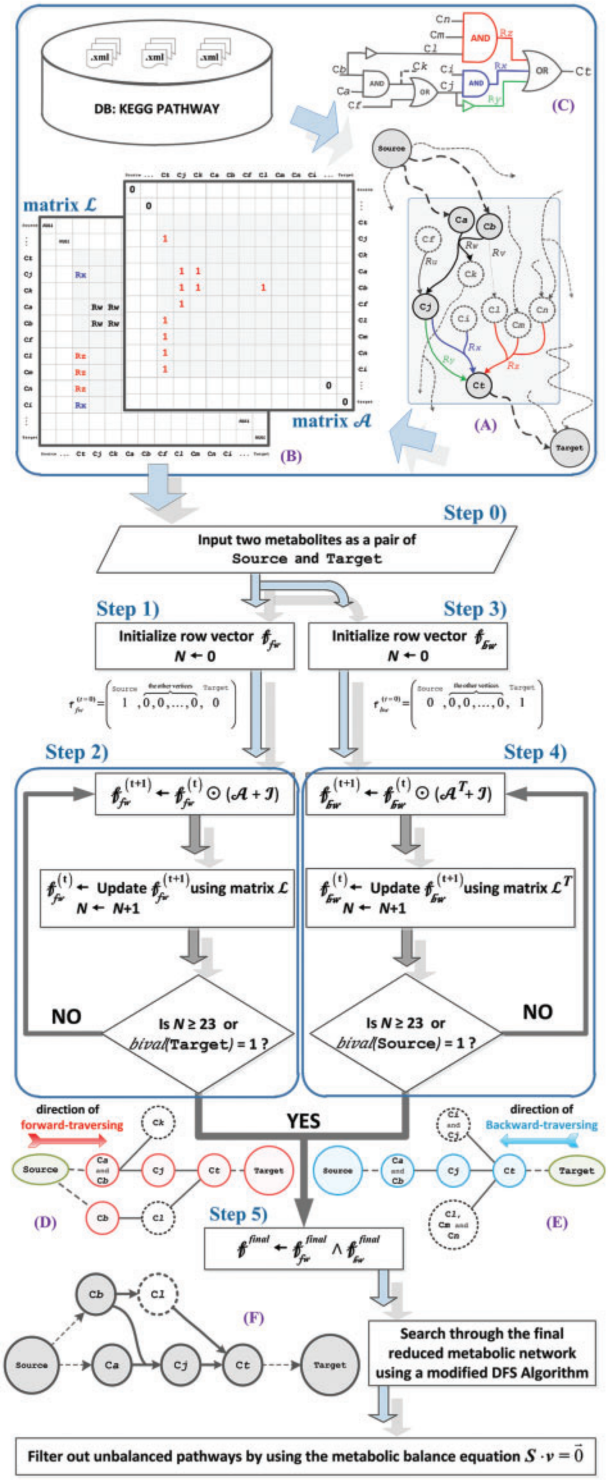
DB: KEGG PATHWAY
.xml
(C)
AND
OR
Rx
Ry
Source
Target
Ca
Cb
Cf
Ck
Cj
Ci
Cl
Cm
Cn
Ct
Rw
Rv
Ru
matrix $\mathcal{L}$
matrix $\mathcal{A}$
(A)
(B)
Step 0)
Input two metabolites as a pair of Source and Target
Step 1)
Initialize row vector $\boldsymbol{f}_{fw}$
$N \leftarrow 0$
Step 3)
Initialize row vector $\boldsymbol{f}_{bw}$
$N \leftarrow 0$
Step 2)
$\boldsymbol{f}_{fw}^{(t+1)} \leftarrow \boldsymbol{f}_{fw}^{(t)} \odot (\mathcal{A}+\mathcal{I})$
$\boldsymbol{f}_{fw}^{(t)} \leftarrow$ Update $\boldsymbol{f}_{fw}^{(t+1)}$ using matrix $\mathcal{L}$
$N \leftarrow N+1$
Is $N \geq 23$ or $bival$(Target) = 1 ?
NO
Step 4)
$\boldsymbol{f}_{bw}^{(t+1)} \leftarrow \boldsymbol{f}_{bw}^{(t)} \odot (\mathcal{A}^T+\mathcal{I})$
$\boldsymbol{f}_{bw}^{(t)} \leftarrow$ Update $\boldsymbol{f}_{bw}^{(t+1)}$ using matrix $\mathcal{L}^T$
$N \leftarrow N+1$
Is $N \geq 23$ or $bival$(Source) = 1 ?
NO
YES
direction of forward-traversing
direction of Backward-traversing
(D)
(E)
Step 5)
$\boldsymbol{f}^{final} \leftarrow \boldsymbol{f}_{fw}^{final} \wedge \boldsymbol{f}_{bw}^{final}$
(F)
Search through the final reduced metabolic network using a modified DFS Algorithm
Filter out unbalanced pathways by using the metabolic balance equation $S \cdot v = \vec{0}$

Fig. 1. Overview of FogLight approach: (**A**) Hypergraph representation of a metabolic network, (**B**) Matrix representation of the network, (**C**) Logic circuit diagram of the boxed subgraph, (**D**) The remaining vertices at the termination of Step 2 (all with the Boolean value '1'); the green elliptical vertex is initialized by the Boolean value '1' in Step 1. Lines 7–12 of Supplementary Figure S4 show the pseudocode of the Boolean SpMV of Step 2 using Definition 6. (**E**) The remaining vertices at the termination of Step 4 (all with the Boolean value '1'); the green vertex is initialized by the Boolean value '1' in Step 3. Lines 20–28 of Supplementary Figure S4 show the pseudocode of the Boolean SpMV-T of Step 4 using Definition 7. (**F**) The reduced appropriate part between Source and Target through which search will proceed. All parts of this figure can be found in Supplementary information

The entries labeled by '*NULL*' values are either participated in a logical disjunction or there is no arc from v_i to v_j in the AND/OR graph G'.

In the metabolic AND/OR graph, the label $\ell_e \in \mathbb{L}_{\mathbb{A}}$ can be the unique name of the reaction e in which the initial and terminal vertices are involved or any other distinctive label.

In brief, the two above matrices specify three types of relations ('OR', 'AND', or no relation) between vertices of the AND/OR graph as follows:

1. If there is an OR relation between the two arcs (v_i, v_j) and $(v_k,\ v_j)$, then $A_{ij} = A_{kj} =$ '1' and $L_{ij} = L_{kj} =$ '*NULL*'.
2. If there is an AND relation between the two arcs (v_i, v_j) and $(v_k,\ v_j)$, then $A_{ij} = A_{kj} =$ '1' and $L_{ij} = L_{kj} = \ \ell_e$.
3. If there is no arc from vertex v_i to vertex v_j, then $A_{ij} =$ '0' and $L_{ij} =$ '*NULL*'.

Figure 1B demonstrates an overview of the whole matrix representation of the AND/OR graph corresponding to the hypergraph of Figure 1A. Each row and column of the two matrices of Figure 1B corresponds to a unique metabolite. The shaded areas of the matrix views are particularly used to display only the boxed subgraph shown in Figure 1A. All non-specified entries in these areas of matrices $\mathcal{A}$ and $\mathcal{L}$ in Figure 1B are respectively set as '0' and "*NULL*".

For example, in Figure 1B, there is an arc from $\mathtt{C_i}$ to $\mathtt{C_t}$ which is represented by $\mathcal{A}[\mathtt{C_i}][\mathtt{C_t}] = 1$ in Figure 1B. Moreover, since there are more than one incoming arc to the terminal vertex $\mathtt{C_t}$ with the label $\mathtt{R_x}$ (i.e. $\mathcal{A}[\mathtt{C_i}][\mathtt{C_t}] = \mathcal{A}[\mathtt{C_j}][\mathtt{C_t}] = 1$), we store it in the corresponding entries of the label matrix $\mathcal{L}$ in Figure 1B, i.e. $\mathcal{L}[\mathtt{C_i}][\mathtt{C_t}] = \mathcal{L}[\mathtt{C_j}][\mathtt{C_t}] = R_x$.

2.2.2 FogLight

A reduced search space consists of a subset of vertices and edges of the initial graph between two given vertices as the source and target points with the aim of finding pathways. In order to do this and speed up the pathway finding in the metabolic networks, we propose an efficient algorithm, called FogLight, using our matrix representation of the metabolic AND/OR graph. To better understand how FogLight reduces the graph (Fig. 1A), we exemplify its steps (Section 2.2.2.2) in Supplementary information.

Before describing our algorithm in details, it is necessary to introduce some mathematical aspects of the materials used in the algorithm as well as the related definitions.

2.2.2.1 Definitions and notations (from linear algebra). The Boolean algebraic operations on binary matrices are analogous to the real matrix operations, except we use the Boolean operators $\wedge$ (logical AND) and $\vee$ (logical OR) on the binary elements instead of multiplication and addition on real numbers, respectively.

Definition 4: Let $\mathcal{A} = [A_{ij}]$ be an n-by-n binary matrix and $\boldsymbol{b} = (b_1, b_2, b_3, \ldots, b_n)$ be an n-dimensional binary row vector. The *Boolean product* of $\boldsymbol{b}$ and $\mathcal{A}$ (denoted by $\boldsymbol{b} \odot \mathcal{A}$) is an n-dimensional binary row vector $\boldsymbol{p} = (p_1, p_2, p_3, \ldots, p_n)$ whose entries are given by $p_j = (b_1 \wedge A_{i1}) \vee (b_2 \wedge A_{i2}) \vee (b_3 \wedge A_{i3}) \vee \cdots \vee (b_n \wedge A_{in})$ for $1 \leq i \leq n$.

Definition 5: Given n Boolean variables $u_1, u_2, u_3, \ldots, u_n$ of a binary vector $\boldsymbol{u}$, *Boolean function vector* $\boldsymbol{f} = (f_1, f_2, f_3, \ldots, f_n)$ is defined as a binary vector whose entry f_i represents a binary value calculated from a Boolean function $F_i(u_1, u_2, u_3, \ldots, u_{i-1}, u_{i+1}, \ldots, u_n)$.

Considering Figure 1A and its corresponding logic circuit diagram made up of basic logic 'AND' and 'OR' gates in Figure 1C, the Boolean value of f_{c_t} and f_{c_j} are obtained by the functions F_{c_t} and F_{c_j}, respectively, where each $bival(\mathtt{C_x})$ represents the binary value of vertex $\mathtt{C_x}$.

$$F_{c_t} = \Big(\big(bival(\mathtt{C_i}) \wedge bival(\mathtt{C_j})\big) \vee \big(bival(\mathtt{C_j})\big) \vee \big(bival(\mathtt{C_l}) \wedge bival(\mathtt{C_m}) \wedge bival(\mathtt{C_n})\big)\Big)$$

$$F_{c_j} = \Big(\big(bival(\mathtt{C_a}) \wedge bival(\mathtt{C_b})\big) \vee bival(\mathtt{C_f})\Big)$$

Definition 6: Assuming an asymmetric real sparse matrix, an efficient storage format, called *Compressed Sparse Row (CSR)* format, is widely used in *sparse matrix-vector multiplication (SpMV)*.

Let n_{nz} denote the number of nonzero entries of an n-by-n sparse binary matrix $\mathcal{A} = [A_{ij}]$. CSR storage format is used to store $\mathcal{A}$ into three arrays with the following characteristics:

1. The first array is *val*, of length n_{nz}, and holds all nonzero entries of $\mathcal{A}$ as they are traversed in a row-wise fashion.
2. An integer n_{nz}-ary array, namely *col_idx*, is the second array which contains the column indices of nonzero entries A_{ij} of the original binary matrix $\mathcal{A}$. That is, if $val[k] = A_{ij} = 1$, then $col\_idx[k] = j$, for $1 \le k \le n_{nz}$.
3. The last array is an integer array named *rptr* which is used to store the indices of the beginning of each row in both arrays *val* and *col_idx*; it means that if $A_{ij} = 1$, then $rptr[i] \le k < rptr[i+1]$, for $1 \le k \le n+1$.

However, the above-mentioned format can be modified to save binary sparse matrices efficiently. To this end, we choose not to save the array *val* since its all elements are '1' and the Boolean multiplication process of SpMV does not need it in practice. We name this new storage as *binary CSR* (*b-CSR*) format which is defined by only two arrays *col_idx* and *rptr*.

Definition 7: In contrast to b-CSR, another storage format, namely *binary Compressed Sparse Column* (*b-CSC*), is defined to be used in the *transposed variant of Boolean SpMV* (*BSpMV-T*). A sparse binary matrix $\mathcal{A}^T = [A_{ji}]$ can be stored into the following two arrays (see Supplementary Fig. S3A (S3B) for the b-CSR (b-CSC) storage format of the upper matrix of Fig. 1B):

1. The first array is *row_idx* which contains the row indices of nonzero entries a_{ji} of $\mathcal{A}$. That is, if $val[k] = A_{ji} = 1$, then $row\_idx[k] = i$, for $1 \le k \le n_{nz}$.
2. Column indices of the first nonzero element in each row of matrix $\mathcal{A}^T$ are referred to by an integer value saved in array *cptr*.

2.2.2.2 Matrix-based algorithm. Using the matrix modeling of the AND/OR graph, we propose a matrix-based algorithm to prune the unrelated off-the-path pathways between the two given vertices of source and target (Fig. 1, Step 0). For this purpose, the graph is independently traversed in two opposite directions, i.e. one from the vertex `Source` and the other from the vertex `Target`. Since Steps 1–2 and Steps 3–4 are done independently (as shown in Fig. 1), each of them can be implemented either sequentially or in parallel.

In the AND/OR graph, the vertices are traversed from the given source, level by level and the entries of the vector are assigned by a Boolean value according to its corresponding Boolean functions; note that all vertices (except the source vertex) have been initialized by '0'. A similar traversal is performed from the target. Finally, only vertices assigned twice by the Boolean value of '1' are marked as the intermediate vertices to find the paths. Hereafter, in order to find paths between the pair of source and target, in the second stage, we search through the reduced space by successive Depth-First Traversal (with ability to backtrack) considering the AND-OR relations. These processes are repeated until all valid paths are enumerated. The above processes make the search for a path (or paths) much more efficient.

Our proposed algorithm (Fig. 1, Steps 1–5) proceeds in the following steps (the detailed pseudocode is shown in Supplementary Fig. S4):

(Step 1) *Forward-initialization*: To march in time, the initial solution ${\mathscr{f}_{fw}}^{(t=0)}$ must be known. That is, at $t = 0$, the initial Boolean values of all the vertices must be specified. To this end, by choosing a starting vertex $v_i \in \mathbb{V},\ 1 \le i \le n$ and assigning the Boolean values of '1' to it and '0' to the other ones, the Boolean function vector $\mathscr{f}_{fw}$, for forward traversal is initially constructed from the AND/OR graph.

(Step 2) *Forward-traversing*: Here, The strategy is to march in time from the initial values (i.e. ${\mathscr{f}_{fw}}^{(t=0)}$) to some final ones (i.e. ${\mathscr{f}_{fw}}^{final}$). Considering a Boolean value of '1' assigned only to the given source substrate in the previous step, the Boolean values of the other vertices in the graph (which are all initialized to '0') are re-evaluated. The re-evaluation process formulated by Eq. 3 is done until the maximum default depth limit N is reached or the target vertex is met during the vertex- and edge-traversing. This equation can simply be written as an iterative procedure depending on two primary parameters, namely an n-dimensional Boolean function vector $\mathscr{f}_{fw}$ and an n-by-n matrix with the binary value of '1' on its main diagonal and the values of the entries in the binary matrix $\mathcal{A}_{n\times n}$ elsewhere.

$${\mathscr{f}_{fw}}^{(t+1)} = {\mathscr{f}_{fw}}^{(t)} \ \odot\ (\mathcal{A}_{n\times n} + \mathfrak{I}_{n\times n}) \qquad (3)$$

where $\mathcal{A}_{n\times n}$ is the binary matrix representing the connections in the given AND/OR graph and $\mathfrak{I}_{n\times n}$ is a binary identity matrix with the values of '1' on its main diagonal. Considering Eq. 3, we were able to march in time by constructing the vector $\mathscr{f}_{fw}$ at the next time-step (denoted by ${\mathscr{f}_{fw}}^{(t+1)}$) using the fact that we know its values at the previous time-step.

(Step 3) *Backward-initialization*: Next, by choosing an ending point $v_j \in \mathbb{V},\ 1 \le j \le n$, $\mathscr{f}_{bw}$ is defined as a Boolean vector composed of the initial binary values to be assigned to all the vertices.

(Step 4) *Backward-traversing*: In this step, the re-evaluation process is done independent of the last Boolean values obtained from Step 3. Considering '1' as the Boolean value of the given vertex corresponding to the target product, the Boolean values of the other vertices in the graph (which are all initialized to '0') are re-evaluated. The re-evaluation process formulated by Eq. 4 is done N times, where N is the maximum depth limit obtained from the first step (value of N was chosen as 23 in our experiments (Yousofshahi *et al.*, 2011)).

To find the next values of ${\mathscr{f}_{bw}}^{(t+1)}$ from the current values in ${\mathscr{f}_{bw}}^{(t)}$, we use

$${\mathscr{f}_{bw}}^{(t+1)} = {\mathscr{f}_{bw}}^{(t)} \ \odot\ (\mathcal{A}^T_{n\times n} + \mathfrak{I}_{n\times n}) \qquad (4)$$

where $\mathcal{A}^T$ is the transpose of the binary matrix $\mathcal{A}_{n\times n}$.

(Step 5) *Boolean conjunction*: The meet of the two Boolean values obtained from Steps 2 and 4 for each vertex $v_j \in \mathbb{V}$ leads to the calculation in this step, as formulated below:

$$\boldsymbol{f}^{final} = \boldsymbol{f}_{fw}{}^{final} \wedge \boldsymbol{f}_{bw}{}^{final} \tag{5}$$

Here, using the graph representation of a given metabolic network, the above five steps are briefly explained. Figure 1A shows this graph in which all the vertices, including `Source` and `Target`, are the metabolites and denoted by the labels of the compound identifiers. The hyperedges have also been labeled with the reaction identifiers.

Given the two compound identifiers as `Source` and `Target`, the algorithm attempts to find paths between them. It utilizes the five above steps to prune off-the-path branches and mark a small portion of the large network by the Boolean value of ‘1’. The subgraph consisting of the marked vertices is then used in the second stage for searching the path exhaustively. Finally, the set of found pathways, namely $\mathbb{P}$, are checked to see if each individual pathway $p \in \mathbb{P}$ satisfies the steady-state condition subject to the stoichiometry constraints. To this end, the following flux balance equation (i.e. a system of N_{mfp} metabolites involved in N_{rfp} reactions in the pathway) is employed:

$$\mathcal{S} \cdot \boldsymbol{v} = 0 \tag{6}$$

where $\mathcal{S}$ is the $N_{\mathrm{mfp}} \times N_{\mathrm{rfp}}$ stoichiometry matrix, corresponding to each pathway $p \in \mathbb{P}$, and $\boldsymbol{v}$ is the N_{rfp}-dimensional reaction rate vector (also called flux vector), whose *i*th component represents the rate (or flux) of reaction *i*. At the end if for some of the pathways, there is no vector $\boldsymbol{v}$ which meets the flux balance constraint, it is filtered out from the final set of results.

As seen in Step 2, the Boolean value of each vertex of the network is assigned in each time-step by applying the Boolean function of the vertex-inputs of the vertex from the previous level (see Eq. 3); this is done by employing matrices $\mathcal{A}$ and $\mathcal{L}$. For instance, the Boolean values of $\boldsymbol{f}_{fw}$ at $t = 1$ depend on the known values of that Boolean function vector at the previous time (denoted by $\boldsymbol{f}_{fw}{}^{(t=0)}$) which is initialized in Steps 1. In this step, the Boolean value of `Source` in $\boldsymbol{f}_{fw}{}^{(t=0)}$ is initialized by ‘1’ and the others by ‘0’. Then, by substituting it in Eq. 3, and then, using matrix $\mathcal{L}$, the new Boolean values of $\boldsymbol{f}_{fw}$ at the next time-step, i.e. $t = 1$, are re-evaluated. This process is repeated in the forward direction until the Boolean value of `Target` becomes ‘1’ or the depth limit is reached. As a result of this process in forward direction, the Boolean values of the vertices $\mathtt{C_a}$, $\mathtt{C}_b$, $\mathtt{C_i}$, $\mathtt{C_j}$, $\mathtt{C_k}$, $\mathtt{C_l}$ and $\mathtt{C}_t$ between two vertices `Source` and `Target` will be ‘1’ (as shown in Fig. 1D). The same procedure is performed from `Target` to `Source` to mark the vertices in reverse direction (Fig. 1E). Finally, only the vertices with both forward and backward marks as ‘1’ are kept which are $\mathtt{C_a}$, $\mathtt{C}_b$, $\mathtt{C_j}$, $\mathtt{C_l}$ and $\mathtt{C}_t$ (as shown in Fig. 1F). These vertices and their corresponding arcs form on-the-path pathways from `Source` to `Target`. As seen in Figure 1F, the vertex with the label $\mathtt{C_l}$ is assigned twice by the Boolean value of ‘1’ and consequently, it is kept in the subgraph but the pathway containing this vertex is discarded in the second stage.

3 Results and Discussion

Metabolic engineering has the potential to produce fairly large quantities of a wide variety of chemicals from readily available materials. To achieve this goal, many metabolic pathways or product-specific enzymes have been created, modified and engineered and then transferred and combined into the microbial hosts.

FogLight was run in two stages to search through the metabolic network containing two different sets of organisms in order to find the shortest and the all paths from a source to a target over a constant period of time (i.e. 2000 s). The new shortest paths found can be used to define minimal gene sets for designing artificial genomes. The number of pathways found by FogLight between two distinct materials is summarized in Table 1 where each row corresponds to one pair of source-target metabolites. N/A in the table means that no pathways were found in that period of time, e.g. rows 2 and 5 and N/I means that the metabolic network of the selected organisms does not consist of the source or target metabolites, (e.g. row 1 in which Enterobacteriaceae family has no Triacylglycerol).

As a result of the first stage of FogLight, a reduced space (subnetwork) is obtained by pruning off-the-path arcs of the initial (unprocessed) network. The amount of this reduction was reported in the fourth and eighth columns of Table 1. Clearly, no space reduction is possible when the brute-force method is only used. In this table, the percentage of reduction in search space is calculated in terms of the number of edges reduced in the initial network. In fact, the number of edges to be processed in the brute force algorithm can became too large and counting them will be impractical. For example, if the average branching factor of the graph is 4 and the number of levels to be searched is 20, then this number will amount 4^{20}.

In this table, minimal reaction set is the minimum number of biochemical reactions (i.e. shortest paths) for production of a target metabolite from a source. This minimum itself and the number of it have been reported in columns 6 and 5, respectively. Details of the shortest pathways found by FogLight within the reaction network (the compound network) related to conversions from a given source to a target (listed in Table 1) can be found in Supplementary Table S1 (S2). Additionally, the values of reaction fluxes for the steady-state pathways have been reported in the 4th column of Supplementary Table S1. For verification of the steady-state condition of these pathways, an N_{rfp}-by-N_{rfp} system of linear equations in the form of Eq. 6 is solved and the vector value(s) of $\boldsymbol{v}$ that satisfy it are obtained.

Considering the metabolic network containing all organisms, FogLight was run to find the shortest paths and their results were compared with the results of the brute-force method, as an optimum global search technique with high runtime cost. The results show that in most cases, the optimal solutions (for depth-limit ≥ 6) can be found by FogLight in much less time (by one order of magnitude) in comparison with the brute-force method. The only case where FogLight takes more time is the last row of Table 1. This is because the shortest path is very short in this case and brute force can find it in a short time whereas FogLight attempts to reduce the graph in the first stage and this has a runtime overhead.

FogLight can be highly beneficial in finding relevant non-natural pathways when it is given a source and a target metabolite. As shown in Table 1, we compared FogLight to PHT (Rahman *et al.*, 2005) (the new version released in 2011), FMM (Chou *et al.*, 2009) and RouteSearch (Latendresse *et al.*, 2014), three web-based path-finding tools. While RouteSearch uses EcoCyc/MetaCyc as the source database, PHT and FMM have been developed to reconstruct metabolic pathways from one metabolite to another based mainly on KEGG database.

Among the above three tools, RouteSearch and FMM, like our approach, can search for finding paths with different lengths all at once without preliminary setting of their length. Their results, including the information about the shortest paths and all paths

Table 1. Comparing FogLight with brute-force algorithm and three web-based path-finding tools

			Shortest path-finding				All path-finding		
Source	Target	Method	Space reduction percentage	No. of minimal reaction set	Size of minimal reaction set	Total time[d] (s) ($\times 10^2$)	Space reduction percentage	Total no. of pathways	Maximum pathway length
Triacylglycerol	Fructose	FogLight [a,c]	97.42	3	6	0.66	51.84	37	10
		Brute-force Alg.[a]	0	3	6	4.77	0	N/A	N/A
		PHT [a]	–	N/A	N/A	N/A	–	–	–
		FMM [a]	–	3	6	0.08 [f]	–	34	10
		FogLight [b]	N/I	N/I	N/I	N/I	N/I	N/I	N/I
		RouteSearch [b]	–	N/I	N/I	N/I	–	N/I	N/I
Glucose	NADPH	FogLight [a,c]	70.10	1	10	0.14	52.38	18	10
		Brute-force Alg.[a]	0	N/A	N/A	N/A	0	N/A	N/A
		PHT [a]	–	N/A	N/A	N/A	–	–	–
		FMM [a]	–	N/A	N/A	N/A	–	N/A	N/A
		FogLight [b]	63.59	1	10	0.07	38.79	13	10
		RouteSearch [b]	–	2[e]	3	0.09 [f]	–	61[e]	5
Glucose	Ethanol	FogLight [a,c]	89.11	1	6	0.27	51.91	74	10
		Brute-force Alg.[a]	0	1	6	1.45	0	N/A	N/A
		PHT [a]	–	1	10	0.26 [f]	–	–	–
		FMM [a]	–	9	8	0.08 [f]	–	10	9
		FogLight [b]	78.49	2	7	0.42	38.79	8	14
		RouteSearch [b]	–	2[e]	4	0.10 [f]	–	73[e]	6
Glucose	Arginine	FogLight [a,c]	82.50	3	8	0.91	51.82	33	14
		Brute-force Alg.[a]	0	3	8	5.28	0	N/A	N/A
		PHT [a]	–	3	8	0.30 [f]	–	–	–
		FMM [a]	–	10	9	0.08 [f]	–	11	10
		FogLight [b]	67.33	1	9	0.94	38.79	20	14
		RouteSearch [b]	–	1[e]	5	0.09 [f]	–	38[e]	6
Glucose	Valine	FogLight [a,c]	90.88	1	8	0.71	51.95	9	11
		Brute-force Alg.[a]	0	N/A	N/A	N/A	0	N/A	N/A
		PHT [a]	–	1	8	0.26 [f]	–	–	–
		FMM [a]	–	3	9	0.08 [f]	–	3	9
		FogLight [b]	67.29	1	9	0.92	38.79	5	13
		RouteSearch [b]	–	8[e]	5	0.09 [f]	–	66[e]	6
Glucose	Acetate	FogLight [a,c]	93.36	1	5	0.08	51.67	28	9
		Brute-force Alg.[a]	0	1	5	0.07	0	N/A	N/A
		PHT [a]	–	1	5	0.26 [f]	–	–	–
		FMM [a]	–	3	6	0.08 [f]	–	42	10
		FogLight [b]	86.12	1	6	0.13	37.50	15	7
		RouteSearch [b]	–	8[e]	5	0.10 [f]	–	1 (+79[e])	6
Glucose	Tryptophan	FogLight [a,c]	96.31	1	6	0.09	51.80	11	9
		Brute-force Alg.[a]	0	1	6	0.13	0	N/A	N/A
		PHT [a]	–	1	6	0.29 [f]	–	–	–
		FMM [a]	–	1	15	0.08 [f]	–	6	17
		FogLight [b]	88.73	1	7	0.05	38.79	25	13
		RouteSearch [b]	–	2[e]	4	0.09 [f]	–	78[e]	6
CO_2	Ethanol	FogLight [a,c]	98.48	2	3	0.05	51.63	18	8
		Brute-force Alg.[a]	0	2	3	0.03	0	13	7
		PHT [a]	–	1 (+3[e])	4	0.27 [f]	–	–	–
		FMM [a]	–	N/A	N/A	N/A	–	N/A	N/A
		FogLight [b]	93.60	3	5	0.38	38.79	35	9
		RouteSearch [b]	–	2 (+6[e])	3	0.09 [f]	–	16 (+75[e])	9

For comparisons between our approach (FogLight) and brute-force algorithm; the metabolic search spaces were assumed as follows:

[a] For the metabolic network containing all organisms.

[b] For the metabolic network containing family Enterobacteriaceae.

[c] Details of the connected reaction and compound networks of the shortest pathways can be found in Supplementary Tables S1 and S2, respectively.

[d] In FogLight, the search space is reduced in less than 2 s on a Quad-Core 3.8 GHz Intel Core i5-3570 with 8 GB of physical memory.

[e] The numbers indicate the number of biologically irrelevant paths from all of the found pathways.

[f] These runtimes are highly dependent on their server's hardware characteristics and Internet traffic as the experiments are performed on the web.

found by those two, have are presented in Table 1. However, since Pathway Hunter Tool (PHT) needs the user to set the path length, we searched for the shortest paths and therefore, the last two columns of the table are left empty for this tool. In some cases shown in the table, the final results of PHT are similar or inferior to the results obtained by FogLight.

We also noticed that some of the pathways found by these PHT, FMM, and especially RouteSearch are biologically-irrelevant, due to the existence of one or more currency metabolites as the intermediate compounds in the pathways. The number of these pathways has been differentiated in parentheses by the superscript letter ‘e’ in Table 1.

As an illustration, let us consider the third row of Table 1 in which the results (i.e. for the pathways found between glucose and ethanol) of FogLight have been compared with Brute-force Algorithm and the three above-mentioned web-based tools. In this particular example, brute-force, as an infallible but time-consuming technique, was employed and found one shortest biologically-relevant pathway with a length of 6. For comparison, we found the same result by using FogLight in shorter time. Our strategy is to reduce the large metabolic network into a sub-network according to glucose and ethanol (i.e. the given source-target pair) and search exhaustively, but in shorter time, through the reduced network (i.e. nearly %11 of the original network). PHT and FMM have respectively detected 1 and 9 shortest paths with lengths of 10 and 8 which are longer than the optimum length of 6 found by FogLight.

Considering a metabolic network containing family Enterobacteriaceae, FogLight searched through %21.5 of the whole search-space and detected two shortest paths of length 7 while RouteSearch discovered two shorter paths (i.e. with length of 4). However, the shortest paths, and even all the paths of length 5 and 6, found by RouteSearch are biologically-irrelevant and implausible because these paths pass through ADP and NADH, two of the currency metabolites listed in Table 2.

The time complexity analysis of FogLight shows that in the worst-case, the growth of execution time is linearly dependent on the number of reactions. That is, it is always possible to find a constant coefficient K, for which the following relation is satisfied:

$$T_{FogLight} < K.N_{br}$$

where $T_{FogLight}$ is FogLight’s execution time and N_{br} is the number of reactions. As shown in Figure 2, the order of execution time of FogLight is much less than that of the brute-force method in terms of the time complexity (discussed further in Supplementary information).

The first seven rows of this table enumerate the pathways through which glucose is converted to the remarked metabolites by considering two metabolic networks of all organisms versus Enterobacteriaceae. The reason for selecting these usual metabolites (i.e. glucose, arginine, etc.) as the inputs to FogLight is to show that our approach can search correctly and rapidly through the metabolic networks and find some new pathways which did not even exist naturally. On the other hand, we selected a pair of source and target metabolites between which there is no natural pathway (shown in the last row of Table 1) and searched through the networks to find whether there are any pathways between them or not.

Table 2 Currency metabolites and their corresponding IDs

KEGG compound identifier	Compound name
C00001	H_2O
C00002	ATP
C00003	NAD+
C00004	NADH
C00005	NADPH
C00006	NADP+
C00007	O_2
C00010	CoA
C00014	NH_3
C00080	H+
C00008	ADP

Figure 3 shows the entire metabolic network containing the pathways of all species (Fig. 3A) and the appropriate part extracted by our algorithm (Fig. 3B).

These two selected metabolites play key role in development of renewable biofuels (ethanol, isobutanol, isoprenol, etc.) which have drawn significant attention in recent years (Machado and Atsumi, 2012; Tran *et al.*, 2014; Zheng *et al.*, 2013). In addition to natural biofuel-producing systems, recent advances in metabolic engineering have made industries able to produce biofuels through several non-native pathways. Since ethanol (labeled as C00469 in Fig. 3B) is currently the most widely produced and utilized biofuel of the market, the remainder of this section focuses on the use of our approach to find ethanol-production pathways from carbon dioxide (labeled as C00011 in Fig. 3B). Moreover, the numbers of pathways found between CO_2 and ethanol and the search time have been reported in Table 1 as the last pair of source and target.

As depicted by the black ellipses in Figure 3A along with its corresponding graph in Figure 3B, the entire metabolic network composed of pathways of the entire set of organisms registered in the KEGG PATHWAY database has been reduced to a fast-and-easily-searchable space using our algorithm (Fig. 1).

Six natural carbon fixation pathways have been known so far (Fast and Papoutsakis, 2012), of which two are seen in Figure 3B. The first carbon fixation pathway is Calvin-Benson cycle used by plants, algae and cyanobacteria. Cyanobacteria possess some endowments for photoautotrophic conversion of CO_2 into the biofuel products such as ethanol. In addition, they have relatively simple genetic background and well-characterized tools for engineering (Machado and Atsumi, 2012).

Figure 3B illustrates some ethanol-production pathways found by our algorithm, one of which branched from the point of 3-phosphoglycerate (C00197) at the Calvin-Benson cycle to produce ethanol by using a set of enzymes consisting of rubisco (catalyzes reaction R00024), phosphoglycerate mutase (catalyzes R01518), enolase (catalyzes R00658), pyruvate kinase (catalyzes R00200), pyruvate decarboxylase (catalyzes R00224) and alcohol dehydrogenase (catalyzes R00754). While the first four reactions of this

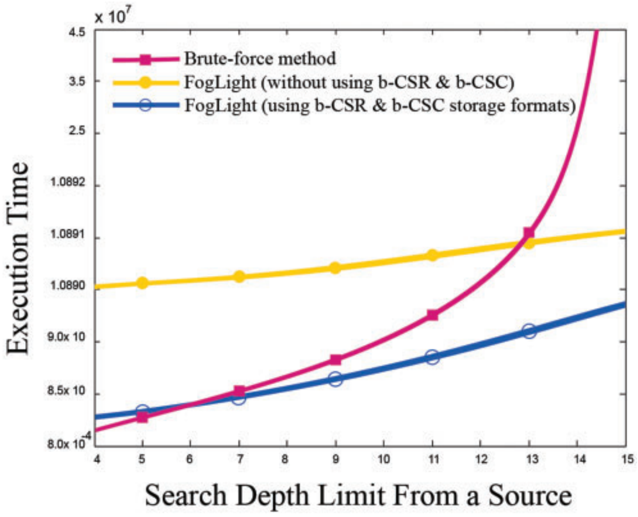

Fig. 2. Execution time comparison between Brute-force and FogLight with/without b-CSR and b-CSC storage formats

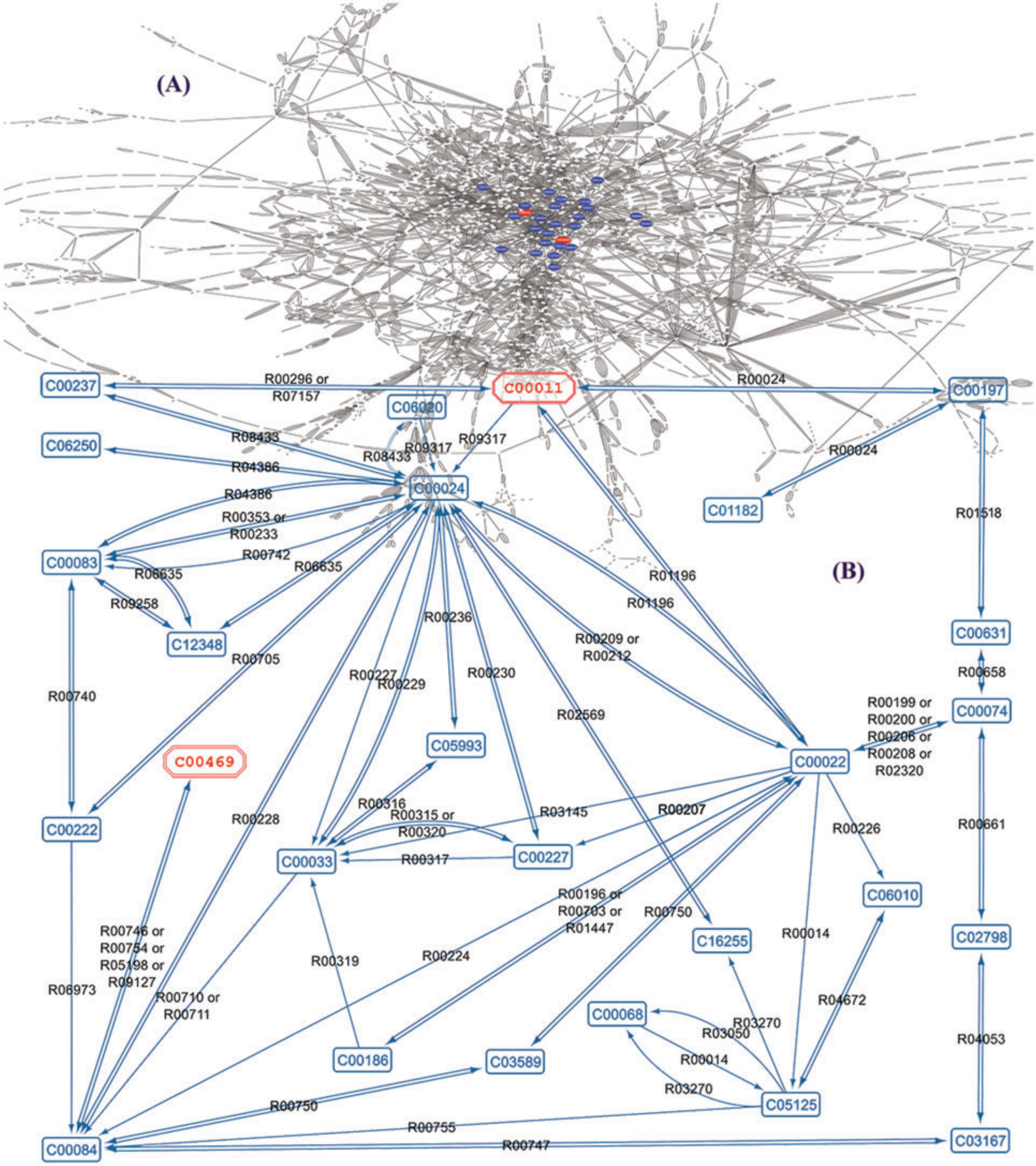

Fig. 3. Entire metabolic network (**A**) containing the pathways of the entire set of organisms, and (**B**) the appropriate part between carbon dioxide and ethanol extracted by FogLight approach; the source/target metabolite is shown with a red single/double-lined octagon and the intermediates with blue rounded rectangles

pathway (i.e. **R00024**, **R01518**, **R00658**, and **R00200**) naturally leads to the formation of pyruvate from CO_2 in cyanobacteria, reactions **R00224** (equally, **R00014+R00755**) and **R00754** are used to reduce pyruvate to ethanol in a non-native manner (Lee, 2011).

An alternative to carbon fixation by the Calvin-Benson cycle is the Wood-Ljungdahl (WL) pathway through which carbon dioxide is anaerobically reduced to form Acetyl-CoA. The key enzyme of WL pathway (shown in Fig. 3B) is CO dehydrogenase/acetyl-CoA synthase (CODH/ACS) which is used to synthesize Acetyl-CoA (i.e. **R09317** or equally, **R07157+R08433**). Considering the production of ethanol from this point (i.e. **C00024**), some number of possible paths and their corresponding reactions can be observed in Figure 3B. Consequently, microbial production of ethanol can be provided by manipulation of a suitable organism genome to encode for appropriate metabolic enzymes of each of the observed pathways. Therefore, it may be possible to chemoautotrophically produce ethanol from CO_2 in the anaerobic bacteria containing WL pathway by combination of the genes coding for the mentioned enzymes.

The other pathway found by our approach includes the shortest one in which enzyme pyruvate synthase (also called ferredoxin 2-oxidoreductase) used as a main catalyzer for CoA-acetylating (i.e. reaction **R01196**). In this pathway, CO_2 along with another substrate is converted into pyruvate and then into ethanol through either two or three enzymatic reactions.

In addition to pyruvate synthase, another mechanism exists for fermentative pyruvate turnover (as shown in Fig. 3B) which uses pyruvate formate-lyase (catalyzes **R00212**) as a main catalyzer to convert pyruvate to ethanol via acetyl-CoA and acetaldehyde (Schomburg and Michal, 2012).

Some other pathways from **C00011** to **C00469** besides the above mentioned pathways can be seen in Figure 3B and have been found by our approach. The pathways differ in the number of reaction steps or their enzymatic types. These differences and varieties are due to the use of numerous enzymatic reactions of different organisms as an including component of the found pathways. Detecting these various metabolic pathways and their participating enzymes for the desired traits, ones are able to create genetically modified organisms through the mutation of genes.

After obtaining the pathways, they should be verified according to stoichiometry constraints with the aim of satisfying steady-state condition. Some of the pathways found by our algorithm have been illustrated in Figure 4. Two pathways which convert CO_2 (C00011) to ethanol (C00469) through the network are as follows:

$$\textit{Pathway}\ 1: \texttt{C00011} \overset{\texttt{R01196}}{\longrightarrow} \texttt{C00022} \overset{\texttt{R00224}}{\longrightarrow} \texttt{C00084} \overset{\texttt{R00746}}{\longrightarrow} \texttt{C000469}$$

$$\textit{Pathway}\ 2: \texttt{C00011} \overset{\texttt{R09317}}{\longrightarrow} \texttt{C00024} \overset{\texttt{R00228}}{\longrightarrow} \texttt{C00084} \overset{\texttt{R00746}}{\longrightarrow} \texttt{C000469}$$

The flux values of reactions on Pathways 1 and 2 were calculated to satisfy Eq. 6 (see Supplementary Table S1 for details). However for the following pathway, there was no flux vector which can satisfy this equation.

$$\textit{Pathway}\ 3: \texttt{C00031} \overset{\texttt{R00305}}{\longrightarrow} \texttt{C00198} \overset{\texttt{R01519}}{\longrightarrow} \texttt{C00257} \overset{\texttt{R01538}}{\longrightarrow} \texttt{C00204}$$

$$\overset{\texttt{R08570}}{\longrightarrow} \texttt{C00022} \overset{\texttt{R00217}}{\longrightarrow} \texttt{C00036} \overset{\texttt{R00357}}{\longrightarrow} \texttt{C00049} \overset{\texttt{R01954}}{\longrightarrow} \texttt{C03406} \overset{\texttt{R01086}}{\longrightarrow} \texttt{C00062}$$

Nevertheless, the imbalance of this pathway was found to be due to the involvement of currency metabolites (shown by dotted ellipses in the figure) in the pathway. By removing these metabolites from the stoichiometry matrix, the steady-state condition was met. The details of matrix calculations can be found in Supplementary information.

In the previous paragraphs, the structure of the different found pathways from CO_2 to ethanol has been described. Therefore, FogLight can efficiently find useful synthetic metabolic pathways in order to be utilized in genetic and metabolic engineering.

4 Conclusion

In order to look for metabolic pathways through metabolic networks, we proposed an approach in which the search space is considerably reduced and the computational cost is decreased to make the problem complexity manageable.

It is possible to lose some admissible candidates as the search-space is narrowed down heuristically as previous approaches suggest. Therefore, our algorithm in the first stage, employs an analytical approach to prune the off-the-path branches of the AND/OR graph of the metabolic network by a matrix-based technique. Then, in the second, it searches for the metabolic pathways through the final reduced network by brute-force method.

The space reduction technique proposed in this article can be used by other techniques which search through metabolic networks, like the heuristic approaches mentioned above, but for the reduced network. Therefore, such approaches can take advantage of this search space reduction to produce similar results as theirs in shorter time. This is because our space reduction technique does not miss any admissible pathways from the given source to target after the graph is trimmed. However, the brute-force method used as the second stage in our algorithm gives all the pathways without the disadvantage of losing any of them as in other approaches. Results demonstrate that our approach is much faster than the brute-force search algorithm, even considering search through the metabolic networks of all organisms.

Using FogLight, we were able to find some metabolic pathways for production of ethanol from carbon dioxide while few of them have been discussed. The found and discussed metabolic pathways are verified by the enzymes, reactions and the partial pathways reported in the literature (Lee, 2011; Schomburg and Michal, 2012). We also compared the complexity of this algorithm with the brute-force search algorithm in looking for all possible pathways leading to a target from a source. For the pathways with the length of greater than six, results show that our approach is much faster than brute-force algorithm and our experiments confirmed the efficiency of our approach.

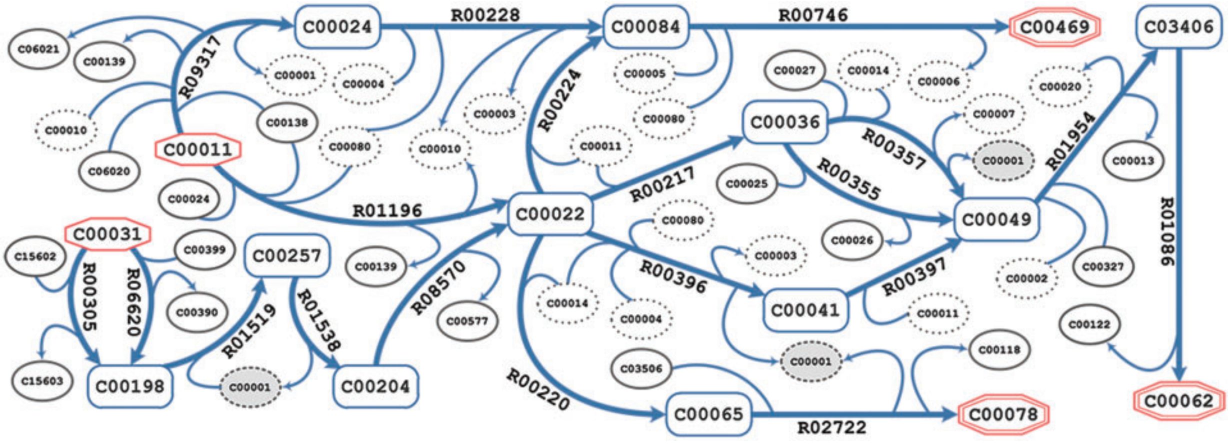

Fig. 4. Pathways from carbon-dioxide and glucose to ethanol (C00469), arginine (C00062) and tryptophan (C00078) found by FogLight

Acknowledgements

We express our special thanks to Dr. Sayed-Amir Marashi for his useful comments during the course of this article.

Conflict of Interest: none declared.

References

Amarú,L. *et al.* (2014) Majority-inverter graph: a novel data-structure and algorithms for efficient logic optimization. In: *Proceedings of the 51st Annual Design Automation Conference*. San Francisco, CA, USA: ACM, pp. 1–6.

Ausiello,G. *et al.* (1992) Optimal traversal of directed hypergraphs. In: *Technical Report TR–92–073*. Berkeley, CA: International Computer Science Institute.

Bagchi,A. and Mahanti,A. (1983) Admissible heuristic search in AND/OR graphs. *Theor. Comput. Sci.*, **24**, 207–219.

Bailey,J.E. (1991) Toward a science of metabolic engineering. *Science*, **252**, 1668–1675.

Beasley,J.E. and Planes,F.J. (2007) Recovering metabolic pathways via optimization. *Bioinformatics*, **23**, 92–98.

Burgard,A.P. *et al.* (2001) Minimal reaction sets for *Escherichia coli* metabolism under different growth requirements and uptake environments. *Biotechnol. Prog.*, **17**, 791–797.

Campodonico,M.A. *et al.* (2014) Generation of an atlas for commodity chemical production in *Escherichia coli* and a novel pathway prediction algorithm, GEM-Path. *Metab. Eng.*, **25**, 140–158.

Carbonell,P.*et al.* (2012) Enumerating metabolic pathways for the production of heterologous target chemicals in chassis organisms. *BMC Syst. Biol.*, **6**, 10.

Caspi,R. *et al.* (2014) The MetaCyc database of metabolic pathways and enzymes and the BioCyc collection of Pathway/Genome Databases. *Nucleic Acids Res.*, **42**, D459–D471.

Cho,A. *et al.* (2010) Prediction of novel synthetic pathways for the production of desired chemicals. *BMC Syst. Biol.*, **4**, 35.

Chou,C.-H. *et al.* (2009) FMM: a web server for metabolic pathway reconstruction and comparative analysis. *Nucleic Acids Res.*, **37**, W129–W134.

Croes,D. *et al.* (2006) Inferring meaningful pathways in weighted metabolic networks. *J. Mol. Biol.*, **356**, 222–236.

Dale,J.M. *et al.* (2010) Machine learning methods for metabolic pathway prediction. *BMC Bioinformatics*, **11**, 15.

Darvas,F. (1988) Predicting metabolic pathways by logic programming. *J. Mol. Graphics*, **6**, 80–86.

Färm,P. *et al.* (2005) Logic optimization using rule-based randomized search. In: *Proceedings of the 2005 Asia and South Pacific Design Automation Conference*. Shanghai, China: ACM, p. 998–1001.

Fast,A.G. and Papoutsakis,E.T. (2012) Stoichiometric and energetic analyses of non-photosynthetic CO2-fixation pathways to support synthetic biology strategies for production of fuels and chemicals. *Curr. Opin. Chem. Eng.*, **1**, 380–395.

Gerard,M.F. *et al.* (2013) An evolutionary approach for searching metabolic pathways. *Comput. Biol. Med.*, **43**, 1704–1712.

Huss,M. and Holme,P. (2007) Currency and commodity metabolites: their identification and relation to the modularity of metabolic networks. *IET Syst. Biol.*, **1**, 280–285.

Jonnalagadda,S. and Srinivasan,R. (2014) An efficient graph theory based method to identify every minimal reaction set in a metabolic network. *BMC Syst. Biol.*, **8**, 28.

Kanehisa,M. *et al.* (2014) Data, information, knowledge and principle: back to metabolism in KEGG. *Nucleic Acids Res.*, **42**, D199–D205.

Karp,P.D. and Mavrovouniotis,M.L. (1994) Representing, analyzing, and synthesizing biochemical pathways. *IEEE Expert*, **9**, 11–21.

Kuffner,R. *et al.* (2000) Pathway analysis in metabolic databases via differential metabolic display (DMD). *Bioinformatics*, **16**, 825–836.

Latendresse,M. *et al.* (2014) Optimal metabolic route search based on atom mappings. *Bioinformatics*, **30**, 2043–2050.

Lee,J.W. (2011) Designer organisms for photosynthetic production of ethanol from carbon dioxide and water. *US Patent No. 7973214 B2*.

Lim,K. and Wong,L. (2012) CMPF: class-switching minimized pathfinding in metabolic networks. *BMC Bioinformatics*, **13**, S17.

Ma,H. and Zeng,A.-P. (2003) Reconstruction of metabolic networks from genome data and analysis of their global structure for various organisms. *Bioinformatics*, **19**, 270–277.

Machado,I.M. and Atsumi,S. (2012) Cyanobacterial biofuel production. *J. Biotechnol.*, **162**, 50–56.

Mavrovouniotis,M.L. (1993) Identification of qualitatively feasible metabolic pathways. *Artif. Intell. Mol. Biol.*, 325–364.

McShan,D.C. *et al.* (2003) PathMiner: predicting metabolic pathways by heuristic search. *Bioinformatics*, **19**, 1692–1698.

Pearcy,N. *et al.* (2014) Hypergraph models of metabolism. *Int. J. Biol. Vet. Agric. Food Eng.*, **8**, 784–788.

Pey,J. *et al.* (2011) Path finding methods accounting for stoichiometry in metabolic networks. *Genome Biol.*, **12**, R49.

Pharkya,P. *et al.* (2004) OptStrain: a computational framework for redesign of microbial production systems. *Genome Res.*, **14**, 2367–2376.

Rahman,S.A. *et al.* (2005) Metabolic pathway analysis web service (Pathway Hunter Tool at CUBIC). *Bioinformatics*, **21**, 1189–1193.

Rodrigo,G. *et al.* (2008) DESHARKY: automatic design of metabolic pathways for optimal cell growth. *Bioinformatics*, **24**, 2554–2556.

Schomburg,D. and Michal,G. (2012) *Biochemical Pathways: An Atlas Of Biochemistry And Molecular Biology*. Hoboken, N.J.: John Wiley & Sons.

Schuster,S. and Hilgetag,C. (1994) On elementray flux modes in biochemical reaction systems at steady state. *J. Biol. Syst.*, **02**, 165–182.

Tran,K.T. *et al.* (2014) Metabolic engineering of *Escherichia coli* to enhance hydrogen production from glycerol. *Appl. Microbiol. Biotechnol.*, **98**, 4757–4770.

Ullah,E. *et al.* (2009) An algorithm for identifying dominant-edge metabolic pathways. In: *ICCAD*. IEEE, pp. 144–150.

Wagner,A. and Fell,D.A. (2001) The small world inside large metabolic networks. *Proc. Biol. Sci. R. Soc.*, **268**, 1803–1810.

Yousofshahi,M. *et al.* (2011) Probabilistic pathway construction. *Metab. Eng.*, **13**, 435–444.

Zheng,Y. *et al.* (2013) Metabolic engineering of Escherichia coli for high-specificity production of isoprenol and prenol as next generation of biofuels. *Biotechnol. Biofuels*, **6**, 57.