Divide-and-conquer approach to the parallel computation of elementary flux modes in metabolic networks

Dimitrije Jevremovic, Daniel Boley
Department of Computer Science & Engineering
University of Minnesota
Minneapolis, MN, USA
Email: jevrem@cs.umn.edu, boley@cs.umn.edu

Carlos P Sosa
IBM, and
Biomed. Infor. & Comp'l. Biology, Univ. of Minnesota
Rochester, MN, USA
Email: cpsosa@us.ibm.com

***Abstract*—Elementary flux modes are an important class of metabolic pathways used to characterize the functioning and behavior of metabolic networks of biochemical reactions in a biological cell. The computation of the elementary flux modes is accomplished by using the so-called Nullspace Algorithm whose high computational cost and memory requirements still limit the computation to relatively small metabolic networks. We combine a "combinatorial" parallelization with a novel divide-and-conquer paradigm into a new implementation of the Nullspace Algorithm with lower memory requirements. We discuss the disadvantages of the combinatorial parallelization and divide-and-conquer ideas and explain why their combination attains more computational power. The improved parallel Nullspace Algorithm is used to compute up to nearly 50 million elementary flux modes for a metabolic network for yeast, a task which was previously not possible using either of the two approaches individually.**

***Keywords*-metabolic network pathways; divide-and-conquer; parallel algorithms; computational biology;**

I. Introduction

Metabolic networks belong to a class of biological networks which allow the representation of biochemical reactions and their relationships within a biological cell or its compartments. One potent way of observing the metabolic network and its interactions is throughout feasible metabolic pathways which satisfy thermodynamic and stoichiometric constraints. Metabolic pathways, in particular a class of *elementary flux modes*, have many different applications in chemical engineering and biochemistry. Elementary flux modes were used in the dissection of a biological cell and analysis of cellular metabolic capabilities [1], [2], phenotype prediction [3], gene knockout studies [4]–[7] and estimation of overall reaction flux distribution [8]–[12].

We propose a new parallel algorithm for the improved computation of *elementary flux modes* which combines an earlier distributed-memory parallel algorithm and a divide-and-conquer idea. The combined algorithm attains lower memory requirements and allows computation of the elementary modes for larger metabolic networks.

This paper is organized as follows. In section II we briefly sketch the background of the metabolic networks and metabolic pathway theory, the elementary flux modes and the Nullspace Algorithm. We also illustrate the Nullspace Algorithm through a simple example network and describe both the combinatorial parallel Nullspace Algorithm and the divide-and-conquer approach. The novel "combined" parallel Nullspace Algorithm is described in section III, and the results of its application to the yeast (*S. cerevisiae*) metabolic network are given in the section IV.

II. Background and related work

A. Metabolic networks and pathways

We use the sample metabolic network in Figure 1 to illustrate our parallel algorithm. This 'toy' network has five internal metabolites (A,B,C,D,P) and nine reactions ($r_1, r_2, r_3, r_4, r_5, r_{6r}, r_7, r_{8r}, r_9$). Each reaction consumes and produces metabolites in fixed proportions. All but two reactions are thermodynamically irreversible, flowing only in the positive direction. Reversible reactions are denoted with a trailing 'r'. Every reaction is characterized by the *reaction rate* (also known as *flux rate*) which numerically gives the rate at which the substrate metabolites are converted to the product metabolites.

The dotted line in Figure 1 marks the boundary between the interior and exterior of the given structure, which may be an entire cell or an internal compartment (organelle). Reactions crossing the network boundary and limited to transport of a particular metabolite between the interior and exterior of the system are known as *exchange reactions* (r_1, r_4, r_{8r}, r_9 in the simple example).

To represent the metabolic network in an analytical way we use the stoichiometry matrix illustrated in equation (2) where rows correspond to metabolites and columns correspond to reactions. Matrix element $N_{i,j}$, if non-zero, gives the molar amount of metabolite i produced (if $N_{i,j} > 0$) or consumed (if $N_{i,j} < 0$) by a unit flux of reaction j. The flux rate for the reaction can be negative only if the reaction is reversible.

At any given moment, some of the reactions in the metabolic network will have non-zero reaction rate, while some others will have zero reaction rate. The complete set

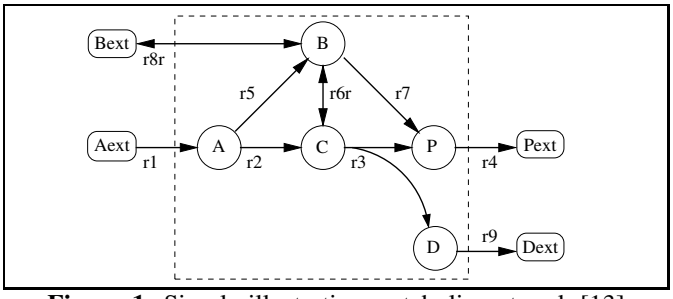

Figure 1: Simple illustrative metabolic network [13]

of flux rates is collected into a *reaction rate (flux rate) vector* r:

$$r = \begin{pmatrix} r_1 & r_2 & r_3 & r_4 & r_5 & r_6 & r_7 & r_8 & r_9 \end{pmatrix}^T \quad (1)$$

The reaction rate vector is equivalent to the concept of a *metabolic pathway*. It was earlier determined that the metabolic network in a biological cell can be considered to be in quasi steady-state when the concentration of each internal metabolite is assumed to be constant [14] and the relationship between the stoichiometry matrix and the reaction rate vector satisfies the constraint (3).

$$N = \begin{array}{c} \\ A \\ B \\ C \\ D \\ P \end{array} \begin{array}{c} \begin{array}{ccccccccc} r_1 & r_2 & r_3 & r_4 & r_5 & r_{6r} & r_7 & r_{8r} & r_9 \end{array} \\ \begin{pmatrix} 1 & -1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & -1 & -1 & 0 \\ 0 & 1 & -1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & -1 & 0 & 0 & 2 & 0 & 0 \end{pmatrix} \end{array} \quad (2)$$

$$N \cdot r = 0 \quad (3)$$

In the typical metabolic network the number of reactions is higher than the number of metabolites so that the constraints (3) have many solutions. Of particular interest is a subset of solutions known as *elementary flux modes (EFM)* [15], [16]. The elementary flux mode is a metabolic pathway or a reaction rate vector with a property that there is no other valid reaction rate vector whose non-zero elements form a subset of the non-zero elements of the given flux vector.

B. Nullspace Algorithm

Currently, the most efficient algorithm for the computation of elementary flux modes is the *Nullspace Algorithm* [17]–[22] which is derived from the "double description method" for the enumeration of extreme rays (or vertices) in the convex polyhedral cone (polytope) [23], [24]. The complexity of the enumeration of extreme rays in the convex cone still remains an open question [25]–[27]. It is however known that the related problem of enumerating vertices in the unbounded polyhedron is NP-hard. Genome-scale metabolic networks [28] may have up to more than 3000 reactions, and the computation of elementary modes in that case still represents a computational challenge.

C. Example: Nullspace Algorithm and EFM computation

Using the example of the metabolic network given in Figure 1, we will illustrate the Nullspace Algorithm [17]–[22] and the parallelization ideas. Prior to running the algorithm, the metabolic network and its stoichiometry matrix may be reduced by eliminating redundant reactions, metabolites, and constraints using known methods [19], [21], [29]. The original stoichiometry matrix in equation (2) can thus be reduced to the equivalent matrix in equation (4). It has been shown [19], [21], [29] that such reduced network has the equivalent set of elementary flux modes as the original one, and this preprocessing reduction step yields a more efficient computation of the elementary flux modes.

$$N_{red} = \begin{array}{c} \\ A \\ B \\ C \\ P \end{array} \begin{array}{c} \begin{array}{cccccccc} r_1 & r_2 & r_3 & r_4 & r_5 & r_{6r} & r_7 & r_{8r} \end{array} \\ \begin{pmatrix} 1 & -1 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & -1 & -1 \\ 0 & 1 & -1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 2 & 0 \end{pmatrix} \end{array}. \quad (4)$$

The Nullspace Algorithm begins by computing a basis for the nullspace of N_{red} (the "nullspace matrix"). This is usually accomplished by reducing N_{red} to row echelon form and permuting the columns so that the stoichiometric matrix takes on the form $(-R^{(2)}, I)$. The resulting nullspace matrix then has the form

$$K_{redperm} = \begin{pmatrix} R^{(1)} \\ R^{(2)} \end{pmatrix} = \begin{pmatrix} I \\ R^{(2)} \end{pmatrix} = \begin{array}{c} \begin{array}{c} r_2 \\ r_4 \\ r_5 \\ r_7 \\ r_1 \\ r_3 \\ r_{6r} \\ r_{8r} \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & -2 \\ -1 & 1 & 0 & -2 \\ 1 & -1 & 1 & 1 \end{pmatrix} \end{array} \quad (5)$$

with the rows corresponding to the identity matrix being pushed to the top. The remaining rows are ordered by the increasing number of non-zero elements in the row [19], [21], [23], a heuristic proven to often improve the efficiency of Nullspace Algorithm. We also reorder the columns of N_{red} to to match the row order of (5), obtaining

$$N_{redperm} = \begin{array}{c} \\ A \\ B \\ C \\ P \end{array} \begin{array}{c} \begin{array}{cccccccc} r_2 & r_4 & r_5 & r_7 & r_1 & r_3 & r_{6r} & r_{8r} \end{array} \\ \begin{pmatrix} -1 & 0 & -1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & -1 & -1 \\ 1 & 0 & 0 & 0 & 0 & -1 & 1 & 0 \\ 0 & -1 & 0 & 2 & 0 & 1 & 0 & 0 \end{pmatrix} \end{array}. \quad (6)$$

The columns of the initial nullspace matrix form the initial set of columns which will be iteratively paired as a convex linear combination of two columns to form candidate elementary flux modes. Some of the candidate columns will be accepted and retained in the nullspace matrix, while some others will be rejected using a so-called "algebraic rank test" [18], [20], [21], [30], as will be described below.

Having reduced the size of the initial metabolic network, computed the initial nullspace matrix and reordered its rows according to proven heuristics [19], [21], [23] we may start the core of the Nullspace Algorithm. In Fig. 2 we show the intermediate nullspace matrices obtained at each iteration

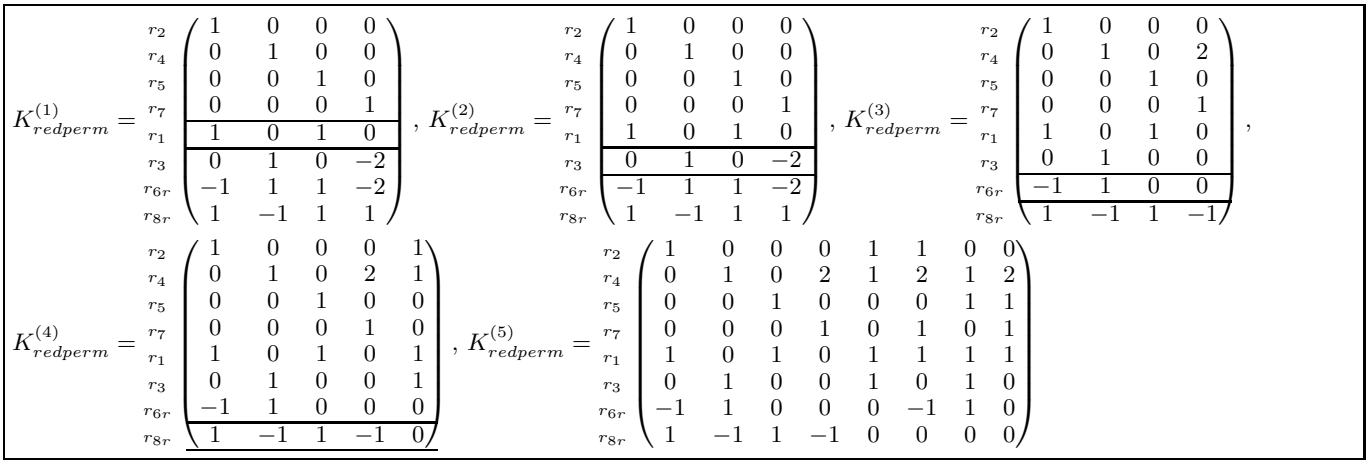

Figure 2: Steps of the Nullspace Algorithm on network of Fig. 1.

of the Nullspace Algorithm. The Nullspace Algorithm starts by processing the first row of the matrix $R^{(2)}$ and ends by processing the last row of the nullspace matrix. In each iteration, we separate the columns according to whether their entry in the row currently being processed is positive, negative, or zero. Columns with a zero entry are simply passed along to the next iteration. Each column with a positive entry is paired with each column with a negative entry to form a convex linear combination such that the combined entry in the current row is zero. The result is a new candidate flux mode.

A rank test [18], [20], [21], [30] is then used to check that the new candidate flux modes are indeed elementary. For every candidate column, we extract the columns from $N_{redperm}$ corresponding to the non-zero entries in the candidate flux mode to form a submatrix. The nullity (dimension of the right nullspace) of this submatrix should be equal to 1 in order to retain the candidate elementary flux mode, or otherwise it will be rejected [18], [20], [21], [30]. If this submatrix has at least two more columns (reactions with non-zero fluxes) than rows (metabolites), then the candidate column has too many non-zeros and therefore is summarily rejected. Otherwise the rank of the submatrix must be computed by using a numerical algorithm such as the LU, QR or SVD [31].

The columns (flux modes) that survive to the next iteration consist of those columns from the previous iteration which had a zero entry in the row currently being processed, plus those columns which had a positive entry, plus the new candidate columns which have passed the rank test. These latter columns necessarily have a zero in the currently processed row. Old columns with negative entries are removed if the reaction corresponding to the current row is irreversible, but these columns are kept if the current reaction is reversible. Because of the fact that during the processing of a row corresponding to a reversible reaction, no column is removed, a usual heuristic is to process reversible reactions last.

We now illustrate in Figure 2 the steps outlined above on the example of Figure 1. We start with the nullspace matrix $K^{(1)}_{redperm}$ and begin the first iteration at row r_1. Since, all the columns have either positive or zero elements in this row, we skip to the next iteration because no candidate columns can be formed. The second iteration is at row r_3 and there is one column with a positive entry and one column with a negative entry. This allows the formation of a single candidate flux mode equal to $\begin{pmatrix}0 & 2 & 0 & 1 & 0 & 0 & 0 & -1\end{pmatrix}^T$.

We then extract the submatrix using the indices of non-zero elements in this candidate column, and compute its nullity. Since the nullity is equal to 1, we retain this column for the next iteration. As the current row corresponds to the irreversible reaction r_3, prior to proceeding to the next iteration, we remove the column having the negative element (-2). The resulting nullspace matrix after the second iteration is given in the matrix $K^{(3)}_{redperm}$. In third iteration, there are again one column with a positive entry and one with a negative entry, which allows the formation of a single candidate elementary flux mode $\begin{pmatrix}1 & 1 & 0 & 0 & 1 & 1 & 0 & 0\end{pmatrix}^T$. This candidate column is retained as well after computing that the nullity is equal to 1.

The third iteration corresponds to the reversible reaction r_{6r}, hence the columns having negative elements for the current row are not removed. In the fourth iteration, the nullspace matrix $K^{(4)}_{redperm}$ has two columns with positive and two columns with negative elements for the row corresponding to the reaction r_{8r}. This produces four candidate elementary flux modes $\begin{pmatrix}1 & 1 & 0 & 0 & 1 & 1 & 0 & 0\end{pmatrix}^T$, $\begin{pmatrix}1 & 2 & 0 & 1 & 1 & 0 & -1 & 0\end{pmatrix}^T$,$\begin{pmatrix}0 & 1 & 1 & 0 & 1 & 1 & 1 & 0\end{pmatrix}^T$, $\begin{pmatrix}0 & 2 & 1 & 1 & 1 & 0 & 0 & 0\end{pmatrix}^T$. Two of these columns are duplicates, so only three are probed using the rank test to check whether they are elementary flux modes. As the generation of candidates may yield duplicated columns a

Algorithm 1 $[EFM] = \texttt{NullspaceAlg}(N, K)$

Input:
reduced stoichiometry matrix ($N_{m\times q}$); *initial nullspace matrix* $K_{q\times(q-m)} = \begin{bmatrix} R^{(1)} \\ R^{(2)} \end{bmatrix}$

Output:
matrix of elementary flux modes $EFM_{q\times n_{ems}}$

1: $K^{(q-m+1)} = K$
2: **for** $k = q - m + 1$ to q **do**
3: $candEFM = \texttt{GenerateEFMCands}(K^{(k)})$
4: $candEFM = \texttt{Sort\&RemoveDuplicates}(candEFM)$
5: $candEFM = \texttt{RankTests}(N, candEFM)$
6: $K^{(k)} = \texttt{RemoveNegColumns}(K^{(k)})$
7: $K^{(k+1)} = [K^{(k)}\ candEFM]$
8: **end for**
9: **return** $K^{(q+1)}$

procedure for their removal is to sort the columns by their binary representation and eliminate duplicated columns in one run.

The final nullspace matrix can be directly obtained from the matrix $K^{(5)}_{redperm}$, and the core of the Nullspace Algorithm is complete. The $K^{(5)}_{redperm}$ matrix corresponds to the permuted reduced stoichiometry matrix given in (6). Rows of the matrix $K^{(5)}_{redperm}$ of EFMs are permuted to the original order, and the row corresponding to the redundant reaction r_9 is added back to obtain the matrix EFM in equation (7).

$$EFM = \begin{array}{c} r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \\ r_{6r} \\ r_7 \\ r_{8r} \\ r_9 \end{array} \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 2 & 1 & 2 & 1 & 2 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ -1 & 1 & 0 & 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & -1 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \end{pmatrix} \quad (7)$$

The high-level pseudocode for the Nullspace Algorithm is sketched in Algorithm 1.

D. Combinatorial parallel Nullspace Algorithm

A shared-memory based parallelization of the Nullspace Algorithm was earlier proposed in EFMTools [19], while the distributed memory parallelization based on message passing was implemented [17] as a better fit to the available computer architecture. Our parallelization when run on a single processor was found in [17] to have similar performance to the serial version of EFMTools. We name this distributed memory version as the "combinatorial parallel Nullspace Algorithm" and give its pseudocode in Algorithm 2. The description of functions used in the pseudocode of the algorithm is given in Table I.

Algorithm 2 $[K] = \texttt{ParallelNullspAlg}(N, K, N_{nodes})$

Input:
reduced stoichiometry matrix ($N_{m\times q}$); *initial nullspace matrix* $K_{q\times(q-m)} = \begin{bmatrix} R^{(1)} \\ R^{(2)} \end{bmatrix}$

Output:
bit-valued matrix of elementary modes $EFM_{q\times n_{ems}}$

1: $K^{(q-m+1)} = K$
2: $procId \Leftarrow$ *compute node* ID
3: **for** $k = q - m + 1$ to q **do**
4: $candEFM = \texttt{ParallelGenerateEFMCands}(K^{(k)}, ProcId, N_{nodes})$
5: $candEFM = \texttt{Sort\&RemoveDuplicates}(candEFM)$
6: $candEFM = \texttt{RankTests}(N, candEFM)$
7: {*communicate candEFM and merge* }
8: $candEFM = \texttt{Communicate\&Merge}(candEFM)$
9: $K^{(k)} = \texttt{RemoveNegColumns}(K^{(k)})$
10: $K^{(k+1)} = [K^{(k)}\ candEFM]$
11: **end for**
12: **return** $K^{(q+1)}$

Table I
DESCRIPTION OF FUNCTIONS USED IN ALGORITHMS 1 AND 2

`GenerateEFMCands`:
Generates next set of candidate elementary modes by convex combinations of current set of modes that annihilates the flux of the current reaction.
`ParallelGenerateEFMCands`:
Generates candidate elementary modes local for the current compute node.
`Sort RemoveDuplicates`:
Sorts the candidate elementary flux modes and removes local duplicates.
`RankTests`:
Applies the algebraic rank tests to the candidates local to current compute node.
`RemoveNegColumns`:
If the current iteration corresponds to an irreversible reaction, remove all columns (modes) for which this reaction's flux is negative.
`Communicate Merge`:
Send each compute node's locally computed EFMs to other compute nodes. Merge the incoming EFMs with local set and remove duplicates.

The combinatorial parallel Nullspace Algorithm [17] attained good scalability, reduced overhead in the communication and merge of exchanged candidate elementary flux modes among the compute nodes. However, its disadvantage is that it still requires to store the copies of the current matrix of candidate EFMs (Fig. 2) in the local memory across all compute nodes during the computation, which imposes an upper limit of the size of the networks which can be handled.

E. Divide-and-conquer

In [32], the authors proposed a parallelization based on the divide-and-conquer approach to compute the complete set of the elementary flux modes. The complete set of the elementary flux modes is partitioned across the selected subset of q_{sub} reactions into $2^{q_{sub}}$ disjoint EFM subsets where the zero/nonzero flux pattern of the elementary flux modes in the i^{th} subset corresponds to the binary repre-

sentation of the number i, for $i \in 0, \ldots, 2^{q_{sub}} - 1$. In the example of 8 elementary flux modes from matrix EFM in (7), the partitions across reactions r_{8r} and r_9 will be {6,8}, {1,3,4}, {5,7}, {2} where subset elements are column indices. The reactions for partitioning elementary flux modes can not be randomly selected, as the pre-processing step of reducing metabolic network size will eliminate some of them. This divide-and-conquer approach is also based on the following proposition [30], [32], which can easily be proved by mathematical induction.

Proposition 1: If the Nullspace Algorithm is stopped at its $(q-q')^{th}$ iteration, then the set of elementary flux modes with all the last q' reactions having non-zero flux values coincides with the set of columns in the current nullspace matrix having non-zero flux values in the last q' elements.

We use Proposition 1 to incorporate the divide-and-conquer idea with the combinatorial parallel Nullspace Algorithm (Algorithm 2), as described in the following section.

III. Combined Parallel Nullspace Algorithm

In Algorithm 3, we propose the incorporation of the divide-and-conquer approach in the combinatorial parallel Nullspace Algorithm described in Algorithm 2. Initially, the reaction subset size q_{sub} is selected and the elementary modes are computed for each of the $2^{q_{sub}}$ subsets. In lines 5 and 6 indices of reactions which should have non-zero and zero flux values are extracted according to the binary representation of the current iteration value k. For the i-th subset, a reduced stoichiometry matrix is formed by removing the columns corresponding to the indices $zfRows$ of reactions with zero flux values. This results in the reduced stoichiometry matrix N_i (line 8), and new nullspace matrix K_i is recomputed (line 9). The rows of the nullspace matrix K_i are reordered so that the reactions corresponding to the indices in $nzfRows$ are put at the bottom of the matrix (line 11). We then run the combinatorial parallel Nullspace Algorithm (line 14) described in Algorithm 2 on the pair (N_i, K_i). Rows corresponding to those reactions which should have zero reaction flux values are appended back to the matrix EFM_i (lines 17-21) and the iteration is complete.

Algorithm 3 $[K] = \texttt{CombParallelNullspAlg}(N, K, q_{sub})$

Input:
stoichiometry matrix ($N_{m\times q}$); *initial nullspace of the form* $K_{q\times(q-m)} = \begin{bmatrix} R^{(1)} \\ R^{(2)} \end{bmatrix}$

Output:
matrix of elementary modes $EFM_{q\times n_{ems}}$

1: {*Reduce initial metabolic network* (N, K) *to equivalent smaller network.*}
2: $(N_{red}, K_{red}) \Leftarrow (N, K)$; $EFM = [\,]$;
3: ***for*** $k = 0$ *to* $2^{q_{sub}} - 1$ ***do***
4: {`dec2binvec` *- get binary representation of number as a vector.* $nzfRows$ *- indices of last* q_{sub} *rows which must have non-zero flux.* $zfRows$ *- indices of last* q_{sub} *rows which must have zero flux.*}
5: $nzfRows = q - q_{sub} + \texttt{find}(\texttt{dec2binvec}(k))$
6: $zfRows = q - q_{sub} + \texttt{find}(\neg\texttt{dec2binvec}(k))$
7: {*from last* q_{sub} *rows of nullspace matrix remove rows corresponding to zero reaction flux for the* k^{th} *subproblem* }
8: $N_i = N_{red}; N_i(:, zfRows) = [\,]$;
9: $K_i = \texttt{null}(N_i)$
10: {*reorder rows in* K_i *so that rows corresponding to* $nzfRows$ *are at the bottom* }
11: $K_i = \texttt{ReorderRows}(K_i); N_i = \texttt{ReorderColumns}(N_i)$;
12: $lastRowIter := q - q_{sub}$
13: {*Run parallel algorithm on the pair* (N_i, K_i) *until reaching iteration corresponding to* $lastRowIter^{th}$ *row*}
14: $EFM_i = \texttt{ParallelNullspAlg}(N_i, K_i, N_{nodes}, lastRowIter)$
15: {*keep only those columns in* EFM_i *which have non-zero values in the last* $length(nzfRows)$ *rows* }
16: $selectedRowInd = q - q_{sub} + (1 : \texttt{length}(nzfRows))$
17: $EFM_i = EFM_i(:, \texttt{all}(EFM_i(selectedRowInd, :)))$
18: {*add zero-rows to the* EFM_i *in order to obtain the wanted subset of EFMs*}
19: $numEms_i = \texttt{size}(EFM_i, 2)$
20: $EFM_i(nzfRows, :) = EFM_i(selectedRowInd, :)$
21: $EFM_i(zfRows, :) = \texttt{zeros}(length(zfRows), numEms)$
22: $EFM = [EFM\ EFM_i]$
23: ***end for***
24: ***return*** EFM

A. Example

We now illustrate the divide-and-conquer idea on our sample metabolic network. Once the initial nullspace matrix is found for the stoichiometry matrix, we consider the four subproblems across the two reactions r_{6r} and r_{8r}.

- *Reactions r_{6r}, r_{8r} should both have zero flux values.* Columns corresponding to the reactions r_{6r}, r_{8r} are removed from the matrix $N_{redperm}$ in equation (6) to obtain the matrix given in equation (8).

$$N_{r00} = \begin{array}{c} \\ A \\ B \\ C \\ P \end{array} \begin{array}{c} \begin{array}{cccccc} r_2 & r_4 & r_5 & r_7 & r_1 & r_3 \end{array} \\ \begin{pmatrix} -1 & 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & -1 \\ 0 & -1 & 0 & 2 & 0 & 1 \end{pmatrix} \end{array}. \quad (8)$$

The nullpace matrix corresponding to the matrix N_{r00} is given as K_{r00}.

$$K_{r00} = \begin{array}{c} r_2 \\ r_4 \\ r_5 \\ r_7 \\ r_1 \\ r_3 \end{array} \begin{pmatrix} 2 & 0 \\ 0 & 2 \\ -1 & 1 \\ -1 & 1 \\ 1 & 1 \\ 2 & 0 \end{pmatrix} \times \frac{1}{2}. \quad (9)$$

When the Nullspace Algorithm is run on the pair (N_{r00}, K_{r00}) two elementary flux modes are obtained:

$$EFM_{r00} = \begin{array}{c} r_2 \\ r_4 \\ r_5 \\ r_7 \\ r_1 \\ r_3 \end{array} \begin{pmatrix} 0 & 1 \\ 2 & 1 \\ 1 & 0 \\ 1 & 0 \\ 1 & 1 \\ 0 & 1 \end{pmatrix}. \quad (10)$$

- *Reaction r_{6r} should have zero and reaction r_{8r} should have non-zero flux values.* This requires the removal of the column corresponding to the reaction r_{6r} in the matrix $N_{redperm}$ to obtain matrix N_{r01}

$$N_{r01} = \begin{array}{c} \\ A \\ B \\ C \\ P \end{array} \begin{array}{c} \begin{array}{ccccccc} r_2 & r_4 & r_5 & r_7 & r_1 & r_3 & r_{8r} \end{array} \\ \begin{pmatrix} -1 & 0 & -1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & -1 \\ 1 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & -1 & 0 & 2 & 0 & 1 & 0 \end{pmatrix} \end{array}. \quad (11)$$

The nullpace matrix corresponding to the matrix N_{r01} is given as K_{r01}.

$$K_{r01} = \begin{array}{c} \\ r_2 \\ r_4 \\ r_5 \\ r_7 \\ r_1 \\ r_3 \\ r_{8r} \end{array} \begin{array}{c} \\ \begin{pmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \\ -1 & 1 & 0 \\ 2 & 0 & 2 \\ 2 & 0 & 0 \\ 1 & -1 & 2 \end{pmatrix} \end{array} \times \frac{1}{2}. \quad (12)$$

Running the Nullspace Algorithm until before the row corresponding to the reaction r_{8r} on a pair (N_{r01}, K_{r01}), and extracting those columns having non-zero flux values for the reaction r_{8r} we obtain two elementary flux modes:

$$EFM_{r01} = \begin{array}{c} \\ r_2 \\ r_4 \\ r_5 \\ r_7 \\ r_1 \\ r_3 \\ r_{8r} \end{array} \begin{array}{c} \\ \begin{pmatrix} 0 & 0 \\ 2 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ -1 & 1 \end{pmatrix} \end{array}. \quad (13)$$

- *Reaction r_{6r} should have non-zero and reaction r_{8r} should have zero flux values.* This requires the removal of the column corresponding to the reaction r_{8r} in the matrix $N_{redperm}$ to obtain matrix N_{r10}.

$$N_{r10} = \begin{array}{c} \\ A \\ B \\ C \\ P \end{array} \begin{array}{c} \begin{array}{ccccccc} r_2 & r_4 & r_5 & r_7 & r_1 & r_3 & r_{6r} \end{array} \\ \begin{pmatrix} -1 & 0 & -1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & -1 \\ 1 & 0 & 0 & 0 & 0 & -1 & 1 \\ 0 & -1 & 0 & 2 & 0 & 1 & 0 \end{pmatrix} \end{array}. \quad (14)$$

The nullspace matrix corresponding to the matrix N_{r10} is given as K_{r10}.

$$K_{r10} = \begin{array}{c} \\ r_2 \\ r_4 \\ r_5 \\ r_7 \\ r_1 \\ r_3 \\ r_{6r} \end{array} \begin{array}{c} \\ \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ -1 & 1 & -1 \\ 1 & 0 & 1 \\ 2 & -1 & 2 \\ 1 & -1 & 2 \end{pmatrix} \end{array}. \quad (15)$$

Similarly, using the Nullspace Algorithm and running it until before the reaction corresponding to the row r_{6r}, two elementary flux modes are obtained:

$$EFM_{r10} = \begin{array}{c} \\ r_2 \\ r_4 \\ r_5 \\ r_7 \\ r_1 \\ r_3 \\ r_{6r} \end{array} \begin{array}{c} \\ \begin{pmatrix} 1 & 0 \\ 2 & 1 \\ 0 & 1 \\ 1 & 0 \\ 1 & 1 \\ 0 & 1 \\ -1 & 1 \end{pmatrix} \end{array}. \quad (16)$$

- *reactions r_{6r} and r_{8r} have both non-zero fluxes.*

$$N_{r11} = \begin{array}{c} \\ A \\ B \\ C \\ P \end{array} \begin{array}{c} \begin{array}{cccccccc} r_2 & r_4 & r_5 & r_7 & r_1 & r_3 & r_{6r} & r_{8r} \end{array} \\ \begin{pmatrix} -1 & 0 & -1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & -1 & -1 \\ 1 & 0 & 0 & 0 & 0 & -1 & 1 & 0 \\ 0 & -1 & 0 & 2 & 0 & 1 & 0 & 0 \end{pmatrix} \end{array}. \quad (17)$$

$$K_{r11} = \begin{array}{c} \\ r_2 \\ r_4 \\ r_5 \\ r_7 \\ r_1 \\ r_3 \\ r_{6r} \\ r_{8r} \end{array} \begin{array}{c} \\ \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & -2 \\ -1 & 1 & 0 & -2 \\ 1 & -1 & 1 & 1 \end{pmatrix} \end{array}. \quad (18)$$

In this last case, the Nullspace Algorithm is run on a pair of matrices (N_{r11}, K_{r11}) until before the row corresponding to the reaction r_{6r}, and the elementary flux modes having non-zero values for both reactions r_{6r} and r_{8r} are extracted. This yields two elementary flux modes:

$$EFM_{r11} = \begin{array}{c} \\ r_2 \\ r_4 \\ r_5 \\ r_7 \\ r_1 \\ r_3 \\ r_{6r} \\ r_{8r} \end{array} \begin{array}{c} \\ \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \\ -1 & 1 \\ 1 & -1 \end{pmatrix} \end{array}. \quad (19)$$

We see that the union of elementary flux modes obtained for all four cases coincides with the elementary flux modes given in Fig. 2. In our combined parallel Nullspace Algorithm, each of the four subtasks would be run independently using the combinatorial parallel Nullspace Algorithm given earlier in Algorithm 2 and described in [17]. □

IV. Results

We used the "Calhoun" parallel platform of the Minnesota Supercomputing Institute and Blue Gene/P of IBM to test our combined parallel Nullspace Algorithm.

Blue Gene/P parallel configuration with up to 512 compute nodes was used to perform the computation [33]. To provide a better understanding of the results of executing combined parallel Nullspace Algorithm and software we give a brief overview of the Blue Gene/P platform. The smallest component in the system is the chip. Single chip has a PowerPC 450 quad-core processor with 4GB of memory. Each processor core runs at a frequency of 850 MHz, and each processor core can perform four floating-point operations per cycle, giving a theoretical peak performance of 13.6 gigaFLOPS/chip. The chip constitutes the *compute card*. The *I/O card* is the next building block. This card is physically very similar to the compute card. However, the I/O card has the integrated Ethernet enabled for communication with the outside world. The I/O cards and the compute cards form a so-called *node card*. The node card has 2 rows of 16 compute cards and 0-2 I/O nodes depending on the I/O configuration. Further, a midplane has 16 node cards. A

$R4$: F6P + ATP $\Longrightarrow$ FDP + ADP
$R5$: FDP $\Longrightarrow$ F6P
$R9$: PYR + ATP $\Longrightarrow$ PEP + ADP
$R10$: PEP + ADP $\Longrightarrow$ PYR + ATP
$R12$: GL3P + FAD_mit $\Longrightarrow$ DHAP + FADH_mit
$R26$: GL3P $\Longrightarrow$ GLY
$R15$: G6P + 2 NADP $\Longrightarrow$ 2 NADPH + CO2 + RL5P
$R21$: ACCOA + OA $\Longrightarrow$ COA + CIT
$R23$: ICIT + NADP $\Longrightarrow$ CO2 + NADPH + AKG
$R24$: AKG_mit + NAD_mit + COA_mit $\Longrightarrow$ CO2 + NADH_mit + SUCCOA_mit
$R27$: FUM + FADH $\Longrightarrow$ SUCC + FAD
$R33$: PYR + COA $\Longrightarrow$ ACCOA + FOR
$R37$: PYR + ATP + CO2 $\Longrightarrow$ ADP + OA
$R38$: PYR $\Longrightarrow$ ACEADH + CO2
$R40$: ACEADH + NADH $\Longrightarrow$ ETOH + NAD
$R41$: ACEADH + NADP $\Longrightarrow$ AC + NADPH
$R42$: OA + ATP $\Longrightarrow$ PEP + CO2 + ADP
$R43$: PEP + CO2 $\Longrightarrow$ OA
$R46$: ICIT $\Longrightarrow$ GLX + SUCC
$R47$: ACCOA + GLX $\Longrightarrow$ COA + MAL
$R53$: ACEADH + NAD $\Longrightarrow$ AC + NADH
$R54$: ATP $\Longrightarrow$ ADP
$R58$: NADH + NAD_mit $\Longrightarrow$ NAD + NADH_mit
$R59$: NH3ext $\Longrightarrow$ NH3
$R60$: GLY $\Longrightarrow$ GLYext
$R62$: GLCext + PEP $\Longrightarrow$ G6P + PYR
$R63$: AC $\Longrightarrow$ ACext
$R64$: LAC $\Longrightarrow$ LACext
$R65$: FOR $\Longrightarrow$ FORext
$R66$: ETOH $\Longrightarrow$ ETOHext
$R67$: SUCC $\Longrightarrow$ SUCCext
$R68$: O2ext $\Longrightarrow$ O2
$R69$: CO2 $\Longrightarrow$ CO2ext
$R70$: 7437 G6P + 611 G3P + 437 R5P + 130 E4P + 500 PEP + 2060 PYR + 45 ACCOA_mit + 362 ACCOA + 733 AKG + 1232 OA + 1158 NAD + 434 NAD_mit + 6413 NADPH + 1568 NADPH_mit + 40141 ATP + 5587 NH3 $\Longrightarrow$ 1000 BIO + 247 CO2 + 45 COA_mit + 362 COA + 1158 NADH + 434 NADH_mit + 6413 NADP + 1568 NADP_mit + 40141 ADP
$R72$: PYR_mit + COA_mit + NAD_mit $\Longrightarrow$ ACCOA_mit + NADH_mit + CO2
$R73$: OA_mit + ACCOA_mit $\Longrightarrow$ CIT_mit + COA_mit
$R75$: ICIT_mit + NAD_mit $\Longrightarrow$ AKG_mit + NADH_mit + CO2
$R76$: ICIT_mit + NADP_mit $\Longrightarrow$ AKG_mit + NADPH_mit + CO2
$R77$: ICIT + NADP $\Longrightarrow$ AKG + NADPH + CO2
$R82$: MAL_mit + NADP_mit $\Longrightarrow$ PYR_mit + NADPH_mit + CO2
$R85$: ETOH_mit + COA_mit + 2 ATP_mit + 2 NAD_mit $\Longrightarrow$ ACCOA_mit + 2 ADP_mit + 2 NADH_mit
$R86$: ACEADH_mit + NAD_mit $\Longrightarrow$ AC_mit + NADH_mit
$R87$: ACEADH_mit + NADP_mit $\Longrightarrow$ AC_mit + NADPH_mit
$R93$: ADP + ATP_mit $\Longrightarrow$ ADP_mit + ATP
$R98$: FUM_mit + SUCC $\Longrightarrow$ SUCC_mit + FUM
$R100$: SUCC $\Longrightarrow$ SUCC_mit
$R101$: AKG + MAL_mit $\Longrightarrow$ AKG_mit + MAL

Figure 3: *S. cerevisiae* Metabolic Network I with 62 metabolites and 78 reactions: the irreversible reactions

$R3_r$: G6P $\Longleftrightarrow$ F6P
$R6_r$: FDP $\Longleftrightarrow$ G3P + DHAP
$R7_r$: G3P $\Longleftrightarrow$ DHAP
$R8_r$: G3P + NAD + ADP $\Longleftrightarrow$ PEP + ATP + NADH
$R13_r$: DHAP + NADH $\Longleftrightarrow$ GL3P + NAD
$R16_r$: RL5P $\Longleftrightarrow$ R5P
$R17_r$: RL5P $\Longleftrightarrow$ X5P
$R18_r$: R5P + X5P $\Longleftrightarrow$ G3P + S7P
$R19_r$: X5P + E4P $\Longleftrightarrow$ F6P + G3P
$R20_r$: G3P + S7P $\Longleftrightarrow$ E4P + F6P
$R22_r$: CIT $\Longleftrightarrow$ ICIT
$R25_r$: SUCCOA_mit + ADP_mit $\Longleftrightarrow$ ATP_mit + COA_mit + SUCC_mit
$R28_r$: FUM $\Longleftrightarrow$ MAL
$R29_r$: MAL + NAD $\Longleftrightarrow$ NADH + OA
$R30_r$: PYR + NADH $\Longleftrightarrow$ NAD + LAC
$R32_r$: ACCOA + 2 NADH $\Longleftrightarrow$ ETOH + 2 NAD + COA
$R36_r$: ATP + AC + COA $\Longleftrightarrow$ ADP + ACCOA
$R74_r$: CIT_mit $\Longleftrightarrow$ ICIT_mit
$R78_r$: ACEADH_mit + NADH_mit $\Longleftrightarrow$ ETOH_mit + NAD_mit
$R79_r$: SUCC_mit + FAD_mit $\Longleftrightarrow$ FUM_mit + FADH_mit
$R80_r$: FUM_mit $\Longleftrightarrow$ MAL_mit
$R81_r$: MAL_mit + NAD_mit $\Longleftrightarrow$ OA_mit + NADH_mit
$R88_r$: CIT + MAL_mit $\Longleftrightarrow$ CIT_mit + MAL
$R89_r$: MAL + SUCC_mit $\Longleftrightarrow$ MAL_mit + SUCC
$R90_r$: CIT + ICIT_mit $\Longleftrightarrow$ CIT_mit + ICIT
$R92_r$: AC_mit $\Longleftrightarrow$ AC
$R94_r$: PYR $\Longrightarrow$ PYR_mit
$R95_r$: ETOH $\Longrightarrow$ ETOH_mit
$R96_r$: MAL_mit $\Longrightarrow$ MAL
$R97_r$: ACCOA_mit $\Longrightarrow$ ACCOA
$R102r$: OA $\Longleftrightarrow$ OA_mit

Figure 4: *S. cerevisiae* Metabolic Network I with 62 metabolites and 78 reactions: the reversible reactions.

additional internal metabolite:
GLC
added reactions:
$R1$: GLC + ATP $\Longrightarrow$ G6P + ADP
$R14$: GLY + ATP $\Longrightarrow$ GL3P + ADP
$R56$: 24 ADP + 20 NADH_mit + 10 O2 $\Longrightarrow$ 24 ATP + 20 NAD_mit
$R57$: 24 ADP + 20 FADH + 10 O2 $\Longrightarrow$ 24 ATP + 20 FAD
$R61$: GLCext $\Longrightarrow$ GLC
reactions made reversible:
$R54_r$: ATP $\Longleftrightarrow$ ADP
$R60_r$: GLY $\Longleftrightarrow$ GLYext
$R63_r$: AC $\Longleftrightarrow$ ACext
modified reaction:
$R62$: GLC + PEP $\Longrightarrow$ G6P + PYR

Figure 5: *S. cerevisiae* Metabolic Network II with 63 metabolites and 83 reactions: differences from Network I.

rack holds 2 midplanes, for a total of 32 node cards or 1024 compute cards. A full petaflop system contains 72 racks. Finally, the compute nodes may be configured at boot time to operate in one of three modes: a) *symmetric multiprocessing mode* b) *virtual node mode* and c) *dual mode*. *Symmetric-multiprocessing mode* runs the main process on one processor and can spawn up to 3 threads on the remaining processors. In *dual mode*, the CPUs with rank 0 and 2 run a main program process, and each can spawn an additional thread. *Virtual node mode* runs the underlying program on all four processors, without additional threading.

"Calhoun" is an SGI Altix XE 1300 Linux cluster. The cluster consists of 256 compute nodes, each containing two quad-core 2.66 GHz Intel Xeon "Clovertown"-class processors sharing 16 GB of main memory. In total, "Calhoun" consists of 2048 compute cores and 4 TB of main memory. Compute node is consists of two multi-chip modules (MCMs) each composed of two dies. These dies are two separate pieces of silicon connected to each other and arranged on a single module. Each die has two processor cores that share a 4 MB L2 cache. Each MCM communicates with the main memory in the system via a 1,333 MHz front-side bus (FSB). All of the systems within "Calhoun" are interconnected with a 20-gigabit non-blocking InfiniBand fabric used for interprocess communication (IPC). The InfiniBand fabric is a high-bandwidth, low-latency network, the intent of which is to accommodate high-speed communication for large MPI jobs. The nodes are also interconnected with two 1-gigabit ethernet networks for administration and file access, respectively.

Algorithms 1, 2 and 3 are implemented in software which

is distributed under the GNU General Public License (GPL). Source code and documentation are freely available at the web site: http://elmocomp.sourceforge.net/.

We used two metabolic networks of *S. cerevisiae* : Network I of dimensions 62×78 (35×55) and Network II of dimension 63×83 (40×61), respectively, where values in parentheses correspond to the size of the reduced metabolic network after elimination of redundant constraints. The list of reactions in the Metabolic Network I is given in Figures 3 and 4. Metabolic Network II is the same except for the modifications listed in Figure 5. In these figures reversible reactions are denoted with suffix "r" and external metabolites with suffix "ext". Elementary flux modes were computed for Network I using the combinatorial parallel Nullspace Algorithm (Algorithm 2) and the combined parallel Nullspace Algorithm (Algorithm 3) on Intel Xeon (Clovertown) machine. The results of the computation using Algorithm 2 are given in table II, while the results of using Algorithm 3 are given in table III. In table III the row $subset$ identifies the subproblem in the divide-and-conquer partitioning, and the zero-flux pattern in the two reactions $R89_r$ and $R74_r$ used ($\not{R}$ and R denote that the reaction R has zero and non-zero flux value in the given EFM subproblem, respectively). To compute the elementary flux modes for the subproblems in the combined parallel Nullspace Algorithm we used 16 cores across 4 compute nodes and compared that results with the column in Table II corresponding to 16 cores. The divide-and-conquer splitting in the Algorithm 3 decreased the number of intermediate candidate modes from 159,599,700,951 to 81,714,944,316, what resulted in the effective reduction of the computation time from 208.98 seconds ((Table II)) to 141.6 seconds.

The set of EFMs for Metabolic Network II was computed using the combined parallel Nullspace Algorithm (Algorithm 3) on the Blue Gene/P parallel platform (Table IV) using 256 processors in SMP mode. Initially, we tried computing the elementary modes for this network using the combinatorial parallel Nullspace Algorithm (Algorithm 2), but due to high memory requirements the computation had to be abandoned at 59^{th} iteration, two iterations before completion. We used this as a guidance to estimate the number of the partitioning reactions in a divide-and-conquer approach to compute the elementary modes using the combined parallel Nullspace Algorithm. The reactions which were used as the partitioning subset comprised of the last three reactions in the reordered nullspace matrix $\{R60_r, R90_r, R54_r\}$, where the reaction $R60_r$ corresponds to the last row in the nullspace matrix. However, for partitions $\not{R60_r}\not{R90_r}R54_r$ (interrupted during the execution of the last iteration step) and $\not{R60_r}R90_rR54_r$ we were not able to complete the computation irrespective of the number of compute nodes used due to high memory requirements. This required the partitioning of the two subproblems by addition of one more reaction to the subset. We performed further splitting within the two subsets using four instead of three reactions and computed the elementary flux modes for the partitions across the last four reactions in the nullspace matrix corresponding to $\not{R60_r}\not{R90_r}R54_r\not{R22_r}$, $\not{R60_r}\not{R90_r}R54_rR22_r$, $\not{R60_r}R90_rR54_r\not{R22_r}$, $\not{R60_r}R90_rR54_rR22_r$. For the case of subset ID=1 in a three-reaction split we were able to estimate the number of generated intermediate candidate modes to be equal to 21,268,414,872,504. By addition of the fourth reaction to the partitioning subset the number of candidate elementary modes, as shown in Table IV, was reduced to 4,447,206,371,897 (=4,340,558,712,549+106,647,659,348). For the case of subset ID=3, we were not able to compute the number of generated candidate modes in the three-reaction split, as the computation was interrupted two iteration steps before the end, but the number of generated candidate modes in the four-reaction split is given in Table IV.

A. Time scalability

Computation time is proportional to the number of generated intermediate elementary modes. Divide-and-conquer approach usually leads to the decrease of the cumulative number of intermediate modes compared to the unsplit problem, and the execution times are proportional to these numbers of modes. It is yet unclear how to select the subset of reactions in divide-and-conquer that may maximally decrease the number of intermediate candidate elementary flux modes.

B. Memory scalability

The combinatorial parallel Nullspace Algorithm has the disadvantage that it requires the storage of the current nullspace matrix in the local memory across all compute nodes at each step. Hence, until that bottleneck is removed, the combinatorial parallel Nullspace Algorithm may only be used for problems where the current nullspace matrix may fit in the local memory of the compute node. The divide-and-conquer feature of the combined parallel Nullspace algorithm "fits" the larger problem to the available architecture, where combinatorial parallel algorithm only could not be applied. However, cumulative memory requirements for all subproblems compared to the original problem remain the same.

C. Discussion

The divide-and-conquer approach requires the selected of the initial subset of reactions to be used in the partitioning. If the size of this subset is large, the number of partitions to explore may be impractically high. Currently the estimation of the minimal number of partitions that should be used in elementary mode computations is a manual procedure. An automated method to select the subset and estimate the approximate number of elementary modes for a given reaction partition would be helpful to make the combined parallel Nullspace Algorithm a fully automated procedure. In order

Table II
PARALLEL COMPUTATION OF EFMS ON *S. cerevisiae* METABOLIC NETWORK I USING ALGORITHM 2 ON INTEL XEON MACHINE

# nodes	1	2	1	1	4	8	16
# cores per node	1	1	4	8	4	4	4
total # cores	1	2	4	8	16	32	64
memory per core	12gb	12gb	3gb	1.5gb	3gb	3gb	3gb
gen. cand (sec)	2744.76	1383.93	688.60	349.05	179.04	95.44	46.83
rank test (sec)	112.88	77.42	52.80	33.98	20.38	12.21	8.01
communicate (sec)	0	0.06	0.09	0.18	0.17	0.19	0.18
merge (sec)	0	0.68	1.01	1.40	1.45	1.62	1.74
total time (sec)	2894.40	1490.85	761.29	404.33	208.98	115.46	61.87
Total # candidate modes: 159,599,700,951 Total # EFM: 1,515,314							

Table III
PARALLEL COMPUTATION OF EFMS ON *S. cerevisiae* METABOLIC NETWORK I USING ALGORITHM 3 ON INTEL XEON (CLOVERTOWN) MACHINE WITH PARTITIONING ACROSS REACTIONS $\{R89_r, R74_r\}$ USING 16 PROCESSORS

subset	~~$R89_r$~~ ~~$R74_r$~~	~~$R89_r$~~ $R74_r$	$R89_r$ ~~$R74_r$~~	$R89_r R74_r$
# EFM	274,919	599,344	207,533	433,518
gen. cand (sec)	17.50	57.36	17.29	24.61
rank test (sec)	2.96	7.18	2.34	3.78
comm (sec)	0.05	0.10	0.05	0.10
merge (sec)	0.16	0.44	0.11	0.36
total time (sec)	21.97	67.77	20.79	31.07
Cumulative total time: 141.6 secs Total # EFM: 1,515,314				
Total # candidate modes: 81,714,944,316				

Table IV
PARALLEL COMPUTATION OF EFMS ON *S. cerevisiae* METABOLIC NETWORK II USING ALGORITHM 3 ON BLUE GENE/P PARALLEL PLATFORM WITH PARTITIONING ACROSS REACTIONS $\{R54_r, R90_r, R60_r\}$ AND USING 256 COMPUTE NODES

subset ID	binary partition subset	# candidate modes	# EFM	time (sec)
0	~~$R60_r$~~ ~~$R90_r$~~ ~~$R54_r$~~	6,214,645,617,622	5,461,652	1575.23
1	~~$R60_r$~~ ~~$R90_r$~~ $R54_r$ ~~$R22_r$~~	4,340,558,712,549	5,192,050	1105.26
	~~$R60_r$~~ ~~$R90_r$~~ $R54_r$ $R22_r$	106,647,659,348	713,038	53.45
2	~~$R60_r$~~ $R90_r$ ~~$R54_r$~~	15,066,250,207,733	9,565,657	3342.69
3	~~$R60_r$~~ $R90_r R54_r$ ~~$R22_r$~~	4,644,781,999,541	7,768,777	1380.74
	~~$R60_r$~~ $R90_r R54_r$ $R22_r$	378,647,219,526	2,070,396	178.15
4	$R60_r$ ~~$R90_r$~~ ~~$R54_r$~~	692,798,105,813	2,004,634	255.18
5	$R60_r$ ~~$R90_r$~~ $R54_r$	5,498,326,647,776	5,902,918	1437.18
6	$R60_r R90_r$ ~~$R54_r$~~	1,634,149,803,325	3,170,692	548.96
7	$R60_r R90_r R54_r$	1,724,004,561,529	7,920,995	766.17
Total # EFM:49,764,544 Total time: 2h 57min 23 secs				

to give an intuition about the computational complexity of this problem it would be worth mentioning that to enumerate all the elementary modes having non-zero flux for a specific reaction is NP-hard [26], [27]. In addition, to decide if there exists an elementary mode with non-zero fluxes for two or more given reactions is NP-hard as well.

V. CONCLUSION

This paper gives an improved parallel Nullspace Algorithm to compute the elementary flux modes in the metabolic network. The earlier combinatorial parallel Nullspace Algorithm [17] which had high memory requirements was combined with the divide-and-conquer idea [32] for the computation of the elementary flux modes. The efficiency of applying the divide-and-conquer approach depends on the proper selection of the reactions subset used to partition the space of elementary flux modes into disjoint subsets computed independently. The combination of two algorithmic ideas may reduce the computation time by lowering the total number of intermediate candidate elementary modes and fit the larger problems to the available parallel architecture where previously the combinatorial parallel Nullspace Algorithm failed. Using this method, we were able to complete the computation of the nearly 50 million elementary flux modes on the variation of the yeast metabolic network. Future work should focus on several points. First, the current nullspace matrix should not be stored across all the compute nodes in the combinatorial parallel Nullspace Algorithm, but should be partitioned in an efficient way instead. Second, design of the strategy of selecting the reaction subset used in the divide-and-conquer part of the algorithm that generates minimal number of intermediate elementary mode candidates would lead to the improvement of the computation time. Third, different algorithmic paradigms such as partitioning of the metabolic network graph as an alternative to the divide-and-conquer approach exposed in this paper should also be considered.

ACKNOWLEDGMENT

This work was supported by NSF grant 0534286, IBM Ph.D. fellowship program and Biomedical Informatics and Computational Biology Program of the University of Minnesota, Rochester. We would like to thank Friedrich Srienc of the Chemical Engineering and Materials Science Department, University of Minnesota, Twin Cities for providing the metabolic network data. We are grateful for the resources and technical support from the Minnesota Supercomputing Institute and IBM Rochester Blue Gene Center, as well as to Cindy Mestad and Steven Westerbeck for their support.

REFERENCES

[1] S. Schuster, D. Fell, and T. Dandekar, "A general definition of metabolic pathways useful for systematic organization and analysis of complex metabolic networks," *Nature Biotechnology*, vol. 18, no. 3, pp. 326–332, 2000.

[2] S. Wiback and B. Palsson, "Extreme pathway analysis of human red blood cell metabolism," *Biophysics Journal*, vol. 83, no. 2, pp. 808–818, 2002.

[3] J. Stelling, S. Klamt, K. Bettenbrock, S. Schuster, and E. D. Gilles, "Metabolic network structure determines key aspects of functionality and regulation," *Nature*, vol. 420, no. 6912, pp. 190–193, 2002.

[4] U. Haus, S. Klamt, and T. Stephen, "Computing knock-out strategies in metabolic networks," *Journal of Comp. Biology*, vol. 15, no. 3, pp. 259–268, 2008.

[5] C. Trinh, P. Unrean, and F. Srienc, "A minimal Escherichia coli cell for most efficient ethanol production from hexoses and pentoses," *Applied and Environ. Microbiology*, vol. 74, no. 12, pp. 3634–3643, 2008.

[6] C. Trinh and F. Srienc, "Metabolic engineering of Escherichia coli for efficient conversion of glycerol to ethanol," *Applied and Environmental Microbiology*, vol. 75, no. 21, pp. 6696–6705, 2009.

[7] O. Hädicke and S. Klamt, "CASOP: A computational approach for strain optimization aiming at high productivity," *Journal of Biotechnology*, vol. 147, no. 2, pp. 88–101, 2010.

[8] J. Schwartz and M. Kanehisa, "A quadratic programming approach for decomposing steady-state metabolic flux distributions onto elementary modes," *Bioinformatics*, vol. 21, pp. ii204–ii205, 2005.

[9] J. Schwartz and M. Kanehisa, "Quantitative elementary mode analysis of metabolic pathways: the example of yeast glycolysis," *BMC Bioinformatics*, vol. 7, no. 186, 2006.

[10] Q. Zhao and H. Kurata, "Maximum entropy decomposition of flux distribution at steady state to elementary modes," *J. of Bioscience and Bioeng.*, vol. 107, no. 1, pp. 84–89, 2009.

[11] Q.Zhao and H.Kurata, "Genetic modification of flux for flux prediction of mutants," *Binformatics Systems biology*, vol. 25, no. 13, pp. 1702–1708, 2009.

[12] Q.Zhao and H.Kurata, "Use of maximum entropy principle with lagrange multipliers extends the feasibility of elementary mode analysis," *J. of Bioscience and Bioeng.*, vol. 110, no. 2, pp. 254–261, 2010.

[13] C. Trinh, A. Wlaschin, and F. Srienc, "Elementary mode analysis: a useful metabolic pathway analysis tool for characterizing cellular metabolism," *Appl. Microbiol. Biotechnol*, vol. 22, no. 5, pp. 813–826, 2009.

[14] B. Clarke, "Stoichiometric network analysis," *Cell Biophysics*, vol. 12, pp. 237–53, 1988.

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

[16] S. Schuster and C. Hilgetag, "On elementary flux modes in biochemical reaction systems at steady state," *Journal of Biological Systems*, vol. 2, no. 2, pp. 165–182, 1994.

[17] D. Jevremovic, C. T. Trinh, F. Srienc, , C. Sosa, and D. Boley, "Parallelization of nullspace algorithm for the computation of elementary flux modes," Univ. of Minnesota, Computer Science and Eng. Dept. Tech. Rep. 10-028, 2010.

[18] D. Jevremovic, C. T. Trinh, F. Srienc, and D. Boley, "A simple rank test to distinguish extreme pathways from elementary modes in metabolic networks," Univ. of Minnesota, Computer Science and Eng. Dept. Tech. Rep. 08-029, 2008.

[19] M. Terzer and J. Stelling, "Large Scale computation of elementary flux modes with bit pattern trees," *Bioinformatics*, 2008.

[20] R. Urbanczik and C. Wagner, "An improved algorithm for stoichiometric network analysis: theory and applications," *Bioinformatics*, vol. 21, no. 7, pp. 1203–1210, 2005.

[21] J. Gagneur and S. Klamt, "Computation of elementary modes: a unifying framework and the new binary approach," *BMC Bioinformatics*, vol. 5, no. 175, 2004.

[22] C. Wagner, "Nullspace approach to determine the elementary modes of chemical reaction systems," *J. Phys. Chem.*, vol. 108, no. 7, pp. 2425–2431, 2004.

[23] K. Fukuda and A. Prodon, "Double description method revisited," in *Combinatorics and Computer Science*, M. Deza, R. Euler, and I. Manoussakis, Eds. Springer, 1996, pp. 91–111, also tech. report, Mathematics, ETH, 1995. [Online].

[24] T. Motzkin, H. Raiffa, G. Thompson, and R. Thrall, "The double description method," in *Contributions to theory of games*, H. Kuhn and A. Tucker, Eds. Princeton University Press, 1953, vol. II, pp. 51–73.

[25] L. Khachiyan, E. Boros, K. Borys, K. Elbassioni, and V. Gurvich, "Generating all vertices of a polyhedron is hard," *Proceedings of the seventeenth annual ACM-SIAM symposium on discrete algorithms*, pp. 758 – 765, 2006.

[26] V. Acuña, F. Chierichetti, V. Lacroix, A. Marchetti-Spaccamela, M. Sagot, and L. Stougie, "Modes and cuts in metabolic networks: Complexity and algorithms," *BioSystems*, vol. 95, no. 1, pp. 51–60, 2009.

[27] V. Acuña, A. Marchetti-Spaccamela, M. Sagot, and L. Stougie, "A note on the complexity of finding and enumerating elementary modes," *Biosystems*, vol. 99, no. 3, pp. 210–214, 2010.

[28] J. Schellenberger, J. Park, T. Conrad, and B. Palsson, "BiGG: a biochemical genetic and genomic knowledgebase of large scale metabolic reconstructions," *BMC Bioinformatics*, vol. 11, no. 213, 2010.

[29] D. G. Luenberger, *Linear and nonlinear programming*, 2nd ed. Springer, 2003.

[30] D. Jevremovic, C. Trinh, F. Srienc, and D. Boley, "On algebraic properties of extreme pathways in metabolic networks." *Journal of Comp. Biology*, vol. 17, no. 2, pp. 107–119, 2010.

[31] G. H. Golub and C. F. V. Loan, *Matrix computations*, 3rd ed. John Hopkins Univ. Press, 1996.

[32] S. Klamt, J. Gagneur, and A. von Kamp, "Algorithmic approaches for computing elementary modes in large biochemical reaction networks," *Systems Biology, IEE Proceedings*, vol. 152, no. 4, pp. 249–255, 2005.

[33] C. Sosa and B. Knudsen, "IBM System Blue Gene Solution: Blue Gene/P Application Development," http://www.redbooks.ibm.com/abstracts/sg247287.html, 2008.