MATHEMATICS OF COMPUTATION
Volume 83, Number 288, July 2014, Pages 1727–1756
S 0025-5718(2013)02770-6
Article electronically published on December 16, 2013

# EIGENANALYSIS-BASED TASK MAPPING ON PARALLEL COMPUTERS WITH CELLULAR NETWORKS

PENG ZHANG, YUXIANG GAO, JANET FIERSON, AND YUEFAN DENG

Abstract. Through eigenanalysis of communication matrices, we develop a new objective function formulation for mapping tasks to parallel computers with cellular networks. This new formulation significantly speeds up the solution process through consideration of the symmetries in the supply matrix of a network and a transformation of the demand matrix of any application. The extent of the speedup is not easily obtainable through analytical means for most production networks. However, numerical experiments of mapping wave equations on 2D mesh onto 3D torus networks by simulated annealing demonstrate a far superior convergence rate and quicker escape from local minima with our new formulation than with the standard graph theory-based one.

## 1. Introduction

Task mapping is critical for achieving high performance of parallel computers with complex networks such as cellular networks [1]–[10]. All proposed mapping models to date represent parallel computers and applications as graphs. The networking capability of a parallel computer is abstracted as a supply matrix whose entries represent inter-node communication costs while the communication requirements of the tasks of an application are abstracted as a demand matrix whose entries represent inter-task communication loads [10]. The goal of the model is to minimize the objective function, defined as the hop-bytes metric [1, 4, 5, 8–11]. However, this hop-bytes metric is always written as a sum of the element-by-element products of the network supply and application demand matrices. This native and somewhat naive approach completely fails to exploit the symmetries in most cellular networks and thus results in a large number of unnecessary optimization steps for finding a good mapping.

After reformulating the objective function for this task mapping representation, we have discovered many opportunities for speeding up the optimization process and escaping more efficiently from local minima without sacrificing the established convenience in graph theory formulation.

The rest of the paper is organized as follows: In Section 2, we review the graph theory-based mapping formulation and extend it to a new matrix formulation. In Section 3, we introduce and prove the key theorems that lead to an eigenbased quadratic form of the hop-bytes metric. In Section 4, we present the eigenspectra

Received by the editor October 18, 2010 and, in revised form, August 26, 2011 and November 17, 2012.

2010 *Mathematics Subject Classification.* Primary 68M10, 90C06, 90C20, 90C35; Secondary 68M07, 90C90, 65B99, 15A63, 15A18.

*Key words and phrases.* Eigenanalysis, task mapping, large-scale optimization, quadratic programming, graph theory.

of popular networks. In Section 5, we validate and assess the value of our new formulation through analytical and numerical approaches by exploring the symmetries of networks. Conclusions are given in Section 6.

## 2. Graph theory-based mapping formulation

The problem of mapping $n$ tasks of a parallel application onto $p$ nodes of a networked system is well studied [2–6,14,16,17]. It is equivalent to the fundamental quadratic assignment problem (QAP) [10] when $n = p$.

The basic task mapping graph formulation can be reformulated to express the hop-bytes metric in terms of the network supply matrix and the application demand matrix.

A **network topology graph** is a directed graph $G(\boldsymbol{P}, \boldsymbol{L})$, where $\boldsymbol{P}$ and $\boldsymbol{L}$ are the sets of nodes and links. An element $p_i \in \boldsymbol{P}$ represents a node and $p = |\boldsymbol{P}|$ is the total number of nodes. An element $l_{ij} \in \boldsymbol{L}$ represents the connectedness between the node pair $p_i$ and $p_j$, where $l_{ij} = 1$ if there exists a link from $p_i$ to $p_j$ and $l_{ij} = 0$ otherwise. $|\boldsymbol{L}| / |\boldsymbol{P}|$ is the average node degree. The adjacency matrix of such a graph, a binary matrix $L = [l_{ij}] \in R^{p \times p}$, characterizes the connection topology of the network.

**Definition 1.** A **supply matrix of a network** refers to the distance matrix of $G(\boldsymbol{P}, \boldsymbol{L})$, i.e., a matrix $S = [s_{ij}] \in R^{p \times p}$ whose element $s_{ij}$ is defined as

$$s_{ij} = \begin{cases} 0 & \text{for } i = j, \\ \min\left\{n | L_{ij}^n \neq 0\right\} & \text{otherwise.} \end{cases} \tag{1}$$

Here $L^n = \underbrace{L \times \cdots \times L}_{n}$ and $L_{ij}^n$ is the $i$-th row and $j$-th column element of $L^n$. Thus, $s_{ij}$ represents the shortest node-to-node distance from $p_i$ to $p_j$, measured in hops.

An **application graph** is a weighted graph $G(\boldsymbol{T}, \boldsymbol{H})$, where $\boldsymbol{T}$ and $\boldsymbol{H}$ are the sets of tasks and communication loads. An element $t_i \in \boldsymbol{T}$ represents a task and $n = |\boldsymbol{T}|$ is the total number of tasks. A nonnegative element $h_{ij} \in \boldsymbol{H}$ represents the total size of all messages in bytes communicated from $t_i$ to $t_j$.

**Definition 2.** A **demand matrix of an application** refers to the weighted connection matrix of $G(\boldsymbol{T}, \boldsymbol{H})$, i.e., a matrix $D = [d_{ij}] \in R^{n \times n}$ whose element $d_{ij} = h_{ij}$.

A **mapping** refers to the allocation of $n$ tasks onto $p$ nodes and is defined as an embedding $G(\boldsymbol{P}, \boldsymbol{L})$ to $G(\boldsymbol{T}, \boldsymbol{H})$, an injection from $\boldsymbol{T}$ to $\boldsymbol{P}$ such that every $h_{ij} \in \boldsymbol{H}$ corresponds to a path in $S$ and such a mapping relates to an injection map

$$\sigma = \begin{pmatrix} 1 & 2 & \cdots & n \\ i_1 & i_2 & \cdots & i_n \end{pmatrix} \quad \text{or} \quad \sigma(k) = i_k, \tag{2.1}$$

where $i_k \in \{1, 2, \cdots, p\}$ and $k \in \{1, 2, \cdots, n\}$. In this notation, $\sigma(k) = i_k$ implies that task $k$ is assigned to node $i_k$. When $n = p$, this injection map becomes a permutation $\sigma$. The inverse of a mapping $\sigma^{-1}$ is written as

$$\sigma^{-1} = \begin{pmatrix} i_1 & i_2 & \cdots & i_n \\ 1 & 2 & \cdots & n \end{pmatrix} \quad \text{or} \quad \sigma^{-1}(i_k) = k. \tag{2.2}$$

**Definition 3.** A **mapping matrix** is a binary matrix $M = [m_{ij}] \in R^{p\times n}$ whose column vector $m_k$ is

$$m_k = \begin{bmatrix} m_{1k} \\ \vdots \\ m_{pk} \end{bmatrix} = e_{i_k} \in R^p, \tag{3}$$

where $m_{ij} = 1$ implies task $j$ is assigned to node $i$, and $m_{ij} = 0$ otherwise. $e_{i_k}$ is the $i_k$-th column of an identity matrix $I \in R^{p\times p}$. Thus, $M = [e_{i_1}, e_{i_2}, \cdots, e_{i_n}] \in R^{p\times n}$. When $n = p$, the mapping permutation $\sigma$ defines a bi-injective map between an identity matrix $I \in R^{n\times n}$ and $M \in R^{n\times n}$:

$$\sigma : I = \begin{bmatrix} e_1, & e_2, & \cdots, & e_n \end{bmatrix} \leftrightarrow M = \begin{bmatrix} e_{i_1}, & e_{i_2}, & \cdots, & e_{i_n} \end{bmatrix}. \tag{4}$$

**Definition 4.** The **hop-bytes metric** measures the quality of a mapping [2,3,17]. Its expression in the graph representation is

$$f_{S,D}(\sigma) = \sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}\cdot s_{\sigma(i)\sigma(j)} \quad \text{or, equivalently,} \tag{5}$$

$$f_{S,D}(\sigma) = \sum_{i=1}^{n}\sum_{j=1}^{n} d_{\sigma^{-1}(i)\sigma^{-1}(j)}\cdot s_{ij}$$

## 3. Eigenanalysis-based mapping theorems

Several eigenanalysis-based theorems governing the new task mapping formulations are organized as follows:

(1) **Theorem 1**: Extension of the hop-bytes metric from the graph theory formulation to the new matrix formulation;
(2) **Theorem 2**: A weighted quadratic form of the hop-bytes metric through orthogonal diagonalization of a symmetric network supply matrix;
(3) **Theorem 3**: Extension of Theorem 1 by transforming the demand matrix of any applications;
(4) **Theorem 4**: The quadratic form of the hop-bytes metric or the objective function.

**Theorem 1.** *Given a network supply matrix $S \in R^{p\times p}$, an application demand matrix $D \in R^{n\times n}$ and an injective mapping matrix $M \in R^{p\times n}$, the hop-bytes metric in Definition* 4 *is*

$$f_{S,D}(M) = \mathrm{tr}\big(S^T M D M^T\big). \tag{6}$$

*Remark.* In Theorem 1, $\big(MDM^T\big)_{ij}$ is the total size of all messages sent from node $i$ to node $j$ for a given mapping $M$. The term $\big(S^T MDM^T\big)_{ii}$ is the sum of the hop-bytes of all messages traveling to node $i$ from all other nodes. Thus, the trace is the total hop-bytes from all nodes to all nodes as defined in Definition 4.

**Corollary 1.1.** *Given the same conditions as in Theorem* 1*, the hop-bytes in Definition* 4 *can also be expressed as*

$$f_{S,D}(M) = \mathrm{tr}\big(M^T S^T M D\big). \tag{7}$$

This is obvious because trace is invariant under cyclic permutation.

TABLE 1. Symbols and definitions

| Symbols | Definitions |
|---|---|
| $\alpha_{ij}$ or $A_{ij}$ or $(A)_{ij}$ | Element of matrix $A = [a_{ij}]$ in $i$-th row and $j$-th column |
| $A = [a_{ij}]$ | Entry in $i$-th row and $j$-th column of $A$ is $a_{ij}$ |
| $a[i]$ | $i$-th element of a vector $a$ |
| $G(\boldsymbol{P}, \boldsymbol{L})$ | A network graph with $\boldsymbol{P}$ the set of nodes and $\boldsymbol{L}$ the set of links |
| $S = [s_{ij}] \in R^{p \times p}$ | A network supply matrix with entry $s_{ij}$ as the shortest hop distance from node $i$ to $j$ |
| $G(\boldsymbol{T}, \boldsymbol{H})$ | An application graph with $\boldsymbol{T}$ the set of tasks and $\boldsymbol{H}$ the set of communication loads |
| $D = [d_{ij}] \in R^{n \times n}$ | An application demand matrix with entry $d_{ij}$ as the message size sent from task $i$ to $j$ |
| $\sigma$ | A mapping in which task $k$ is assigned to node $\sigma(k)$, $k \in \{1, 2, \cdots, n\}$ |
| $\sigma^{-1}$ | An inverse mapping in which node $i$ is assigned with task $\sigma^{-1}(i)$, $i \in \{1, 2, \cdots, p\}$ |
| $M = [m_{ij}] \in R^{p \times n}$ | A binary mapping matrix with entry $m_{ij} = 1$ implying task $j$ is assigned to node $i$ |
| $f_{S,D}(\sigma)$ | The hop-bytes metric given $S$, $D$, and $\sigma$ in the graph theory-based representation |
| $f_{S,D}(M)$ | The hop-bytes metric provided $S$, $D$, and $M$ in the new matrix representation |
| $\mathrm{tr}(A)$ | Trace of a matrix $A = [a_{ij}] \in R^{n \times n}$, $\mathrm{tr}(A) = \sum_{i=1}^{n} a_{ii}$ |
| $\Lambda = \mathrm{diag}(\lambda_1, \cdots \lambda_n)$ | Diagonal matrix $\Lambda \in R^{n \times n}$ with diagonal elements $\Lambda_{ii} = \lambda_i$, $i \in \{1, 2, \cdots, n\}$ |
| $A^T$ | Transpose of a matrix $A$ |
| $\alpha_1, \cdots, \alpha_n$ | Eigenvalues of a network supply matrix $S \in R^{n \times n}$ |
| $\alpha = [\alpha_1, \cdots, \alpha_n]^T$ | A vector whose entries are eigenvalues of $S$ |
| $\Lambda_s = \mathrm{diag}(\alpha_1, \cdots, \alpha_n)$ | A diagonal matrix whose diagonal elements are eigenvalues of $S$ |
| $q_1, \cdots, q_n$ | Orthonormal eigenvectors of a network supply matrix $S \in R^{n \times n}$ |
| $Q = [q_1, \cdots, q_n]$ | A matrix whose column vectors are orthonormal eigenvectors of $S$ |
| $B$ | A matrix $B = \left(D + D^T\right)/2 \in R^{n \times n}$ in which $D \in R^{n \times n}$ is an application demand matrix |
| $\beta_1, \cdots, \beta_n$ | Eigenvalues of $B \in R^{n \times n}$ |
| $\beta = [\beta_1, \cdots, \beta_n]^T$ | A vector whose entries are eigenvalues of $B$ |
| $\Lambda_b = \mathrm{diag}(\beta_1, \cdots, \beta_n)$ | A diagonal matrix whose diagonal elements are eigenvalues of $B$ |
| $p_1, \cdots, p_n$ | Orthonormal eigenvectors of $B$ |
| $P = [p_1, \cdots, p_n]$ | A matrix whose column vectors are orthonormal eigenvectors of $B$ |
| $[v]_S$ | Coordinate vector of $v$ relative to the basis $S = \{u_1, u_2, \cdots, u_n\}$ so $[v]_S = S^T v$ |
| $\|\|v\|\|_\Lambda$ | Weighted vector norm: $\|\|v\|\|_\Lambda = \sqrt{\sum_{i=1}^{n} \sigma_i v_i^2}$ where $v = [v_1, \cdots, v_n]^T$, $\Lambda = \mathrm{diag}(\sigma_1, \cdots, \sigma_n)$ |
| $\|v\|$ | Dimension of a vector $v$ |
| $\mathbf{1}_N$ | An $N$-dimensional vector containing all 1's |
| $1_{n \times n}$ | A square matrix containing all 1's |
| $I_{n \times n}$ | An identity matrix and $I_{n \times n} \in R^{n \times n}$ |

*Remark.* In Corollary 1.1, $\left(M^T S^T M\right)_{ij}$ is the hop distance from task $i$ to task $j$ for a given mapping $M$. The term $\left(M^T S^T M D\right)_{ij}$ is the sum of the hop-bytes to task $i$ from all other tasks. Thus, similar to the approach from the node perspective, the trace is the total hop-bytes from all tasks to all tasks as defined in Definition 4.

**Symmetric assumption**: For all networks we consider, links are always bidirectional and the distance between a node pair is equal regardless of transmission direction. Such a network is symmetric and so is its supply matrix. We also assume $n = p$, converting the other case of $n < p$ when necessary by padding with noncommunicating "dummy" tasks. Without loss of generality, we assume:

(1) The network supply matrix is symmetric: $S = S^T$;
(2) The number of tasks is equal to the number of nodes: $n = p$, and thus the mapping matrix $M$ is a permutation matrix.

With such assumptions, we have,

**Lemma 1.1.** *In Theorem* 1*, the hop-bytes metric is*

$$f_{S,D}(M) = \mathrm{tr}\left(SMDM^T\right). \tag{8}$$

**Lemma 1.2.** *The mapping matrix is an orthonormal matrix, i.e.,*

$$M^{-1}M = M^T M = I_{n\times n}. \tag{9}$$

**Lemma 1.3.** *Given a supply matrix $S \in R^{n\times n}$, a demand matrix $D \in R^{n\times n}$ and a mapping matrix $M \in R^{n\times n}$, there exists a matrix $A \in R^{n\times n}$ orthonormal such that*

$$f_{S,D}(A) = \mathrm{tr}\left(\Lambda_s A D A^T\right), \tag{10}$$

*where $\Lambda_s = diag(\alpha_1, \cdots, \alpha_n)$ and $\alpha_i$, $i \in \{1, \cdots, n\}$, are the eigenvalues of $S$.*

*Proof.* According to the finite-dimensional spectral theorem for a real symmetric matrix $S$, there exists a real orthonormal matrix $Q = [q_1, \cdots, q_n]$ such that $Q^{-1}SQ = Q^T SQ = \Lambda_s = diag(\alpha_1, \cdots, \alpha_n)$ where $\alpha_i$ and $q_i$ are eigenpairs (eigenvalues and associated orthonormal eigenvectors) of $S$. Substituting $S = Q\Lambda_s Q^T$ into equation (8) yields

$$f_{S,D}(M) = \mathrm{tr}\left(Q\Lambda_s Q^T MDM^T\right) = \mathrm{tr}\left(\Lambda_s Q^T MDM^T Q\right). \tag{11}$$

Let $A = Q^T M$. $A$ is an orthonormal matrix since $Q$ and $M$ are both orthonormal matrices. This produces equation (10). □

**Theorem 2.** *Given the same conditions as in Lemma* 1.3*, the hop-bytes metric is represented in a weighted quadratic form:*

$$f_{S,D}(M) = \sum_{i=1}^{n} \alpha_i \cdot {q_i}^{*T} D q_i^* = \sum_{i=1}^{n} \alpha_i \cdot [q_i]_M^T D [q_i]_M, \tag{12}$$

*where $q_i^* = M^T q_i = [q_i]_M$, and $\alpha_i$ and $q_i$ are eigenpairs of $S$.*

*Proof.* By Lemma 1.3, the hop-bytes metric is $f_{S,D}(A) = \mathrm{tr}\left(\Lambda_s A D A^T\right)$ where $\Lambda_s = diag(\alpha_1, \cdots, \alpha_n)$, $A = Q^T M$, and the column vectors of $Q$ form the eigenspaces of

$S$. Let the column vectors of $A^T$ be $a_i \in R^n$. Thus, we have

$$A = Q^T M = \begin{bmatrix} q_1^T \\ q_2^T \\ \vdots \\ q_n^T \end{bmatrix} M = \begin{bmatrix} q_1^T M \\ q_2^T M \\ \vdots \\ q_n^T M \end{bmatrix} \implies a_i^T = q_i^T M. \tag{13}$$

On the other hand, we have

$$f_{S,D}(A) = \operatorname{tr}\left( \Lambda \begin{bmatrix} a_1^T \\ a_2^T \\ \vdots \\ a_n^T \end{bmatrix} D[a_1, a_2, \cdots, a_n] \right) \tag{14}$$

$$= \operatorname{tr}\left( \begin{bmatrix} \alpha_1 & 0 & \cdots & 0 \\ 0 & \alpha_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \alpha_n \end{bmatrix} \begin{bmatrix} a_1^T D a_1 & * & \cdots & * \\ * & a_2^T D a_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & * \\ * & \cdots & * & a_n^T D a_n \end{bmatrix} \right)$$

$$= \alpha_1 a_1^T D a_1 + \cdots + \alpha_n a_n^T D a_n.$$

Combining equations (13) and (14) produces $f_{S,D}(M) = \alpha_1 q_1^T M D M^T q_1 + \cdots + \alpha_n q_n^T M D M^T q_n$. Letting $q_i^* T = M^T q_i = [q_i]_M$ completes the proof. □

*Remark.* In Theorem 2, $M^T q_i = [q_i]_M$ is the coordinate vector of $q_i$ relative to the basis $M = \{e_{i_1}, e_{i_2}, \cdots, e_{i_n}\}$. It is also a permutation of elements of a supply matrix's eigenvector $q_i$ and

$$q_i^* = M^T q_i = \begin{bmatrix} q_i[i_1] \\ q_i[i_2] \\ \vdots \\ q_i[i_n] \end{bmatrix} = \begin{bmatrix} q_i[\sigma(1)] \\ q_i[\sigma(2)] \\ \vdots \\ q_i[\sigma(n)] \end{bmatrix}, \tag{15}$$

where $q_i[\sigma(k)]$ is the $\sigma(k)$-th element of the vector $q_i$. Then we have

$$f_{S,D}(M) = \sum_{i=1}^{n} \alpha_i {q_i^*}^T D q_i^*, \tag{16}$$

where ${q_i^*}^T D q_i^* = \sum_{l<m} (d_{lm} + d_{ml}) q_i^*[l] q_i^*[m] = \sum_{l<m} (d_{lm} + d_{ml}) q_i[\sigma(l)] q_i[\sigma(m)]$ and thus we obtain the following:

**Corollary 2.1.** *Given the same conditions as in Theorem 2, we have*

$$f_{S,D}(\sigma) = \sum_{i=1}^{n} \sum_{l<m} \alpha_i (d_{lm} + d_{ml}) q_i[\sigma(l)] q_i[\sigma(m)] \quad \textit{or, equivalently,} \tag{17}$$

$$f_{S,D}(\sigma) = \sum_{i=1}^{n} \sum_{l<m} \alpha_i \left( d_{\sigma^{-1}(l)\sigma^{-1}(m)} + d_{\sigma^{-1}(m)\sigma^{-1}(l)} \right) q_i(l) q_i(m).$$

The network is assumed to be symmetric so we can obtain an equivalent form of the hop-bytes metric by symmetrizing the demand matrix in Theorem 3 [10].

**Theorem 3.** *Given a network supply matrix $S \in R^{n\times n}$, an application demand matrix $D \in R^{n\times n}$ and a mapping matrix $M \in R^{n\times n}$, the hop-bytes metric is*

$$f_{S,D}(M) = tr\left(SMBM^T\right), \tag{18}$$

*where $B = \left(D + D^T\right)/2 \in R^{n\times n}$ is a symmetric matrix.*

*Proof.* Since the distance between any two nodes in a symmetric network is the same regardless of data transmission direction, $B$ and $D$ impact the hop-bytes metric identically. □

The bound estimation for the quadratic part of the QAP [8,10] defines a property of the hop-bytes metric on the mapping problem stated in Theorem 3 as:

**Property 3.1.** Given the same conditions as in Theorem 3 with $\alpha_1 \geq \alpha_2 \geq \cdots \geq \alpha_n$ the eigenvalues of $S$ and $\beta_1 \geq \beta_2 \geq \cdots \geq \beta_n$ the eigenvalues of $B$, we have

$$\sum_{i=1}^{n} \alpha_i \beta_{n+i-1} \leq f_{S,D}(M) \leq \sum_{i=1}^{n} \alpha_i \beta_i. \tag{19}$$

These new formulations and theorems for task mapping in the eigen representation can be summarized as

**Theorem 4** (Eigen Representation Theorem)**.** *Given:*

1. *A network supply matrix $S \in R^{n\times n}$ with eigenpairs $\alpha_i$ and $q_i$;*
2. *An application demand matrix $D \in R^{n\times n}$;*
3. *A matrix $B = \left(D + D^T\right)/2$ with eigenpairs $\beta_i$ and $p_i$.*

*The hop-bytes metric as in Definition 4 in the eigen representation is:*

$$f_{S,D}(M) = \alpha^T C \beta. \tag{20}$$

*where $C = [c_{ij}] \in R^{n\times n}$, $c_{ij} = \left(q_i^T M p_j\right)^2 = \left([q_i]_M \cdot p_j\right)^2$, $\alpha = [\alpha_1, \cdots, \alpha_n]^T$, $\beta = [\beta_1, \cdots, \beta_n]^T$, and $M$ is a mapping matrix.*

*Proof.* The symmetric matrix $B$ is orthonormally similar to a diagonal matrix $\Lambda_b$ such that $P^{-1}BP = P^TBP = \Lambda_b = \operatorname{diag}(\beta_1, \cdots, \beta_n)$, where $\beta_i$, $i \in \{1, \cdots, n\}$, and the column vectors of $P = [p_1, \cdots, p_n]$ are the eigenvalues and eigenvectors of $B$. Similarly, the symmetric matrix $S$ is orthonormally similar to a diagonal matrix $\Lambda_s$ such that $Q^{-1}SQ = Q^TSQ = \Lambda_s = \operatorname{diag}(\alpha_1, \cdots, \alpha_n)$, where $\alpha_i$, $i \in \{1, \cdots, n\}$, and the column vectors of $Q = [q_1, \cdots, q_n]$ are the eigenvalues and eigenvectors of $S$. Thus, we have $f_{S,D}(M) = \operatorname{tr}\left(SMBM^T\right) = \operatorname{tr}\left(\Lambda_s H \Lambda_b H^T\right)$, where $H = [h_{ij}] = Q^TMP$ and $h_{ij} = q_i^TMp_j$. Thus,

$$\begin{aligned} f_{S,D}(M) &= \operatorname{tr}\left(\Lambda_s H \Lambda_b H^T\right) \\ &= \operatorname{tr}\left(\begin{bmatrix} \alpha_1 \sum_{k=1}^{n} \beta_k h_{1k}^2 & * & \cdots & * \\ * & \alpha_2 \sum_{k=1}^{n} \beta_k h_{2k}^2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & * \\ * & \cdots & * & \alpha_n \sum_{k=1}^{n} \beta_k h_{nk}^2 \end{bmatrix}\right) \\ &= \sum_{l=1}^{n}\sum_{k=1}^{n} \alpha_l \beta_k h_{lk}^2. \end{aligned} \tag{21}$$

Let $c_{ij} = h_{ij}^2 = \left(q_i^T M p_j\right)^2$. Then we conclude that

$$f_{S,D}(M) = [\alpha_1, \alpha_2, \cdots, \alpha_n] \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1n} \\ c_{21} & c_{22} & \ddots & \vdots \\ \vdots & \ddots & \ddots & c_{n-1,n} \\ c_{n1} & \cdots & c_{n,n-1} & c_{nn} \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{bmatrix}.$$

This produces the quadratic form of the hop-bytes metric through the eigenspaces of $S$ and $B$. □

*Remark.* We call this theorem the Eigen Representation Theorem (ER Theorem).

## 4. Eigenspectra of networks

The following eigenspectra properties of the supply matrix are essential for simplifying the hop-bytes metric in the ER Theorem:

**Property 1.** (General networks): Let $\alpha_1, \cdots, \alpha_n$, with $|\alpha_1| \geq |\alpha_2| \geq \cdots \geq |\alpha_n|$, be the $n$ eigenvalues of a supply matrix $S$. Then we have:

(1) A single positive dominant eigenvalue $\alpha_1$: $\alpha_1 > |\alpha_i|$ $\forall i \in \{2, 3, \cdots, n\}$;
(2) A dominant eigenvector $q_1$ with all positive entries: $q_1[i] > 0$ $\forall i \in \{1, 2, \cdots, n\}$;
(3) The eigenvalue $\alpha_1$ bounded by

$$\min_i \sum_j s_{ij} \leq \alpha_1 \leq \max_i \sum_j s_{ij}. \tag{22}$$

*Proof.* $S$ is a nonnegative and irreducible matrix so the Perron-Frobenius Theorem [15] guarantees Property 1. □

To understand the eigenspectra of popular cellular networks more intuitively, we illustrate the discrete eigenvalues of torus, hypercube, fully-connected, and mesh networks in Figures 1 and 2. In each plot, the horizontal axis displays the total number of nodes, i.e., the network size $N$, and the vertical axis displays the eigenvalues $E$. Figure 1 shows the eigenvalues of supply matrices of the $n$-ary $k$-cube where $k \in [1, 4]$, as well as of two other special networks: hypercube and fully-connected. Figure 2 shows the results for the 2D/3D/4D mesh networks.

In Figures 3 and 4, in addition to the eigenvalues themselves, we also show their multiplicities, for 2D and 3D networks, respectively. Each plot shows the distributions of the nonpositive eigenvalues of torus and mesh networks of the same size. The vertical axes display the eigenvalues. A horizontal bar on the left or right hand side indicates an eigenvalue level of a torus or mesh network, respectively. The eigenvalue of greatest absolute value for a mesh network is always larger than that of a torus of the same size. The digit nearest to a horizontal bar or a set of tightly bunched horizontal bars indicates the multiplicity of a negative eigenvalue or some eigenvalues that are very close to one another.

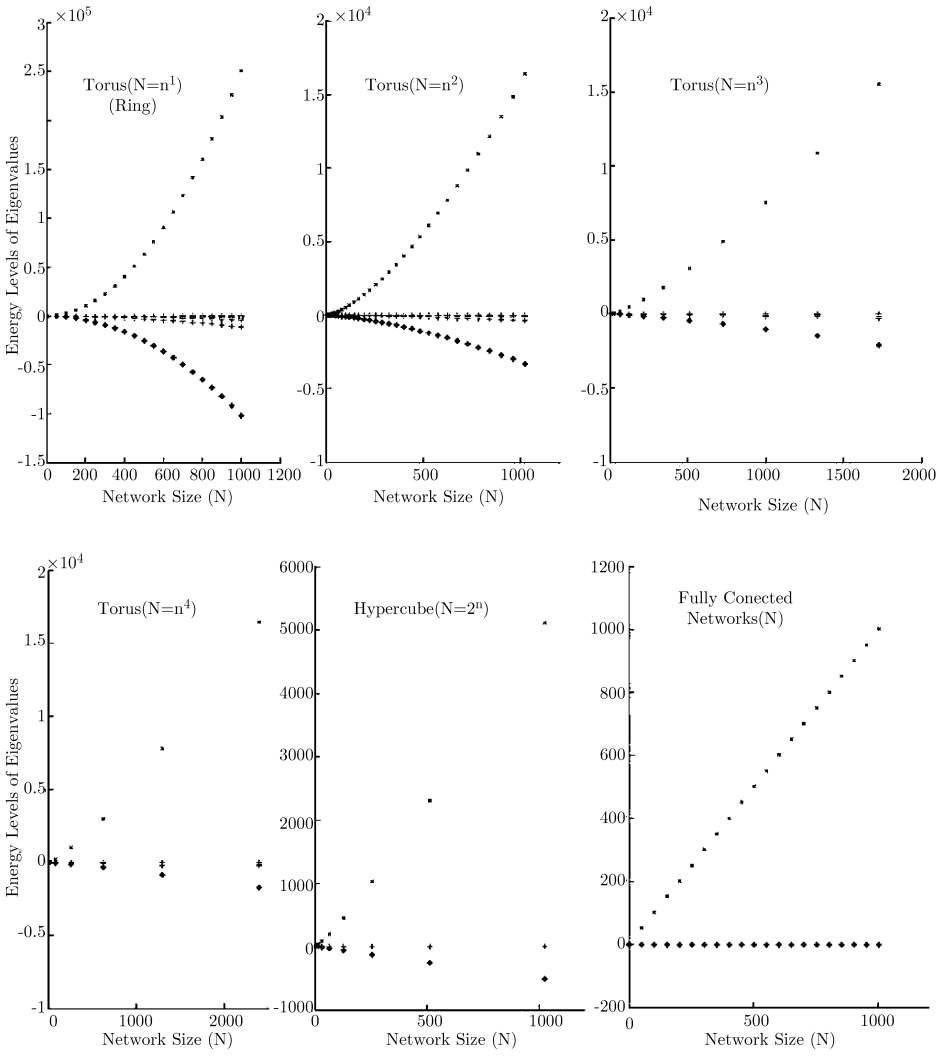


FIGURE 1. Eigenvalues of supply matrices for torus networks

We obtain the following elegant Properties 2 to 5 based on Figures 1 to 4:

**Property 2.** (Torus networks): Let the eigenvalues of Torus $\left(n^d\right)$, for $n \geq 4$ even, be $\alpha_1, \cdots, \alpha_n$, with $|\alpha_1| \geq \cdots \geq |\alpha_N|$. Let $N = n^d$, $\rho$ be the diameter of the network, and $\mu$ the average distance between nodes. Let $\alpha_i$ and $q_i$ be eigenpairs of the supply matrix $S$ of Torus $\left(n^d\right)$. Then we have:

(1) Single dominant eigenvalue $\alpha_1 = dn^{d+1}/4 = N \cdot \mu = N \cdot \rho/2$ and associated eigenvector $q_1 = N^{-1/2} \cdot \mathbf{1}_N$;

(2) $\mu = \frac{1}{N^2} \sum_{i,j,k}^{n} a_i q_i [j] q_i [k]$;

(3) $\alpha_2 = \cdots = \alpha_{2d+1} < 0$ and $|\alpha_2| \in \left(0, \frac{1}{8} n^{d+1}\right]$, more precisely, $|\alpha_2| = \frac{1}{8} n^{d+1}$ if and only if $n = 4$;

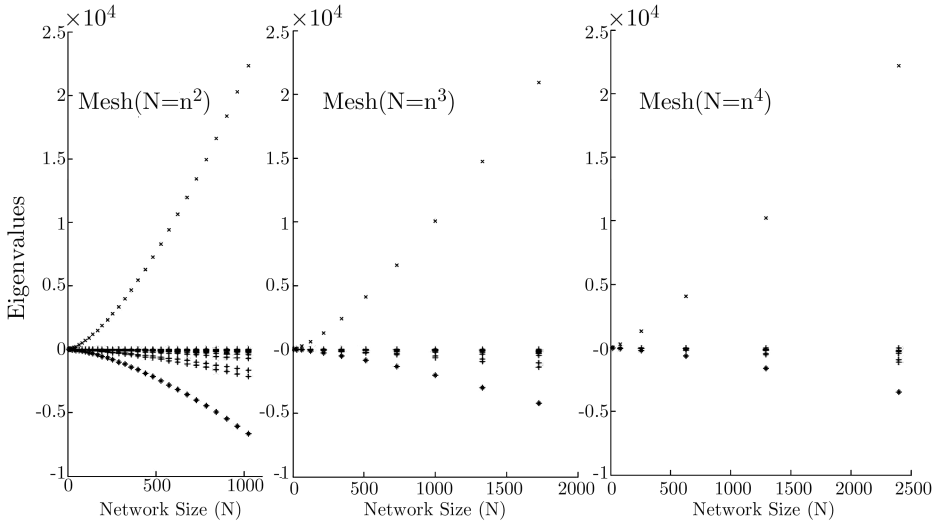


FIGURE 2. Eigenvalues of supply matrices for mesh networks

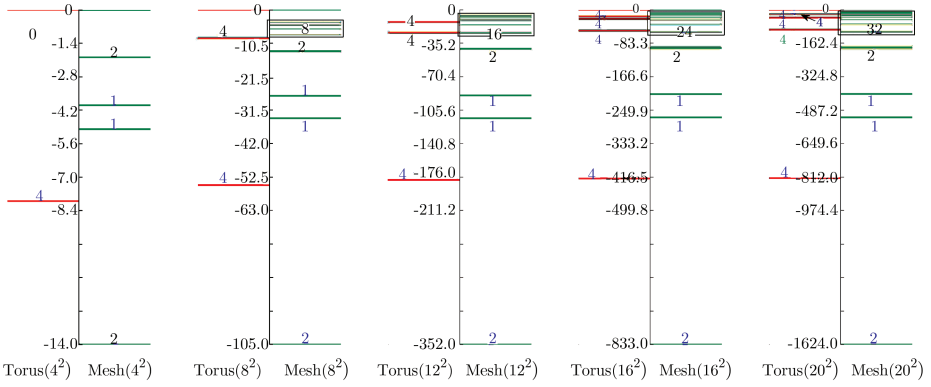


FIGURE 3. Nonpositive eigenvalue distribution of 2D torus and mesh networks

(4) $|\alpha_{2d+1}| > |\alpha_{2d+2}|$;
(5) $|\alpha_j| < |\alpha_2| \cdot 10\%$, $j \in [4d+2, \rho+1]$ and $n \geq 10$;
(6) $\alpha_i \begin{cases} > 0 & \text{for } i = 1, \\ < 0 & \text{for } 2 \leq i \leq \rho+1, \\ = 0 & \forall \text{ other } i. \end{cases}$

*Proof.* Applying Property 1 to torus networks, we achieve the following network-specific properties: a single positive dominant eigenvalue $\alpha_1 = dn^{d+1}/4$ and associated eigenvector $q_1 = N^{-1/2} \cdot \mathbf{1}_N$. Since the average node-to-node distance of a torus is half its diameter; that is, $\mu = \rho/2 = dn/4$, we have $\alpha_1 = N \cdot \mu = N \cdot \rho/2$. This proves Property 2(1).

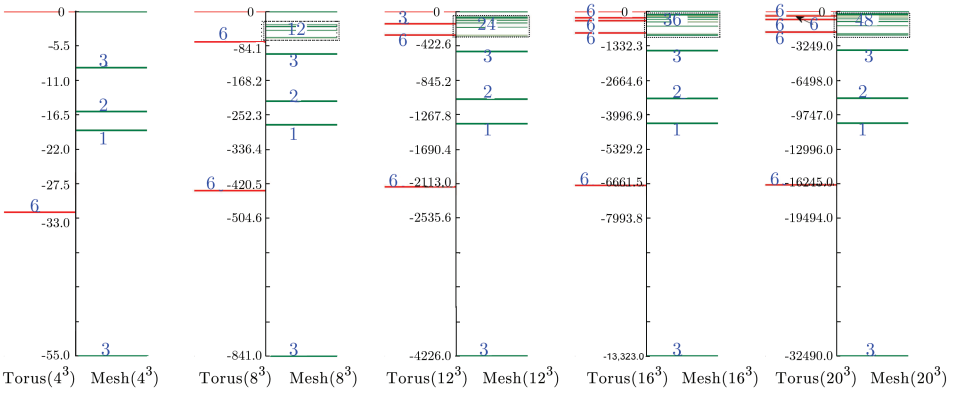


FIGURE 4. Nonpositive eigenvalue distribution of 3D torus and mesh networks

The average node-to-node distance is $\mu = f_{S,1_{N\times N}}(I_{N\times N})/N^2$ where the numerator is the total distance and the denominator is the total number of node pairs $N = n^d$. Thus, Theorem 2 yields

$$f_{S,1_{N\times N}}(I_N) = \sum_{i=1}^{n} \alpha_i q_i^T 1_{N\times N} q_i = \sum_{i,j,k}^{n} a_i q_i[j]\, q_i[k]\,. \tag{23}$$

This proves Property 2(2). □

Figure 1 shows the following elementary observations: (o.1) the second absolute-largest eigenvalue of a torus network is always negative; (o.2) specifically, the trumpet-like curves demonstrate that as the network size increases, such a second absolute-largest eigenvalue of a torus network quickly distances itself from the other absolute-smaller eigenvalues.

Additionally, Figures 3 and 4 show the following observation: (o.3) 2-D and 3-D torus networks always have four and six second absolute-largest negative eigenvalues, respectively.

Thus, these observations (o.1), (o.2), and (o.3) imply Properties 2(3) and 2(4). Through comparing the data in Figure 1 while realizing observation (o.2), we derive Property 2(5). Property 2(6) is a straightforward summary based on Figure 1.

*Remark.* In Property 2(1), $\alpha_1 = N \cdot \mu$ implies that the sum over all nodes of the hop distances from a node to all other nodes is equal to the dominant eigenvalue of the supply matrix of Torus $\left(n^d\right)$.

*Remark.* According to Property 2(1), Theorem 2 for Torus $\left(n^d\right)$ is

$$f_{S,D}(M) = \frac{dn^d}{4}\left(\sum_{i,j} d_{ij}\right) + \sum_{i=2}^{n} \alpha_i \cdot [q_i]_M^T D[q_i]_M. \tag{24}$$

**Property 3.** (Hypercube networks): Let the eigenvalues of Hypercube $\left(2^d\right)$ be $\alpha_1, \cdots, \alpha_n$, with $|\alpha_1| \geq \cdots \geq |\alpha_N|$. Let $N = 2^d$, $\rho$ be the diameter, and $\mu$ be the average distance. Then we have:

(1) Dominant eigenvalue $\alpha_1 = d2^{d-1} = N \cdot \mu = \frac{1}{2}N \cdot \rho$ with associated eigenvector $q_1 = \frac{1}{\sqrt{N}} \cdot \mathbf{1}_N$ and $|\alpha_1| > |\alpha_2|$;
(2) $\alpha_2 = \cdots = \alpha_{d+1} = -2^{d-1} < 0$;
(3) $\alpha_j = 0$, $j \in [d+2, N]$.

*Proof.* Property 3(1) is a special case of Property 2(1).

Property 3(2) can be proved by induction. When $d = 1$, the supply matrix of Hypercube $\left(2^1\right)$ is $\left[\begin{smallmatrix} 0 & 1 \\ 1 & 0 \end{smallmatrix}\right]$ so $-1$ is an eigenvalue of multiplicity 1 with associated eigenvector $[1, -1]^T$. Assume when $d = m$, the supply matrix $A$ of Hypercube $(2^m)$ has an eigenvalue $-2^{m-1}$ of multiplicity $m$ with associated eigenvectors $\vec{v}_1, \cdots, \vec{v}_m$. Then for $d = m + 1$, the supply matrix of Hypercube $\left(2^{m+1}\right)$ is

$$B = \begin{bmatrix} A & A + 1_{2^m \times 2^m} \\ A + 1_{2^m \times 2^m} & A \end{bmatrix}. \tag{25}$$

Using the induction hypothesis, we know $A\vec{v}_l = -2^{m-1} \cdot \vec{v}_l \ \forall\ l \in \{1, \cdots, m\}$. Thus, we have

$$B \begin{bmatrix} \vec{v}_l \\ \vec{v}_l \end{bmatrix} = \begin{bmatrix} A & A + 1_{2^m \times 2^m} \\ A + 1_{2^m \times 2^m} & A \end{bmatrix} \begin{bmatrix} \vec{v}_l \\ \vec{v}_l \end{bmatrix} = -2^m \cdot \begin{bmatrix} \vec{v}_l \\ \vec{v}_l \end{bmatrix}. \tag{26}$$

In addition, we have

$$\begin{aligned} B \begin{bmatrix} \mathbf{1}_{2^m} \\ -\mathbf{1}_{2^m} \end{bmatrix} &= \begin{bmatrix} A & A + 1_{2^m \times 2^m} \\ A + 1_{2^m \times 2^m} & A \end{bmatrix} \begin{bmatrix} \mathbf{1}_{2^m} \\ -\mathbf{1}_{2^m} \end{bmatrix} \\ &= -2^m \cdot \begin{bmatrix} \mathbf{1}_{2^m} \\ -\mathbf{1}_{2^m} \end{bmatrix}. \end{aligned} \tag{27}$$

Therefore, $B$ has eigenvalue $-2^m$ of multiplicity $m+1$ with associated eigenvectors $\left[\vec{v}_l^T, \vec{v}_l^T\right]^T$ $(l \in \{1, \cdots, m\})$ and $\left[1_{2^m}^T, -1_{2^m}^T\right]^T$.

Property 3(3) can also be proven by induction. When $d = 1$, similarly, the supply matrix of Hypercube $\left(2^1\right)$ is $\left[\begin{smallmatrix} 0 & 1 \\ 1 & 0 \end{smallmatrix}\right]$ so the rank of the supply matrix of Hypercube $\left(2^1\right)$ is 2. Assume when $d = m$, the supply matrix $A$ of Hypercube $(2^m)$ has $\operatorname{rank}(A) = m + 1$. Then, for $d = m + 1$, the supply matrix $B$ of Hypercube $\left(2^{m+1}\right)$ has $\operatorname{rank}(B) = \operatorname{rank}(A) + 1$. Using the induction hypothesis, we get $\operatorname{rank}(B) = m + 2$. Therefore, the nullity of the supply matrix of Hypercube $\left(2^d\right)$ is $2^d - d - 1$. Thus, 0 is an eigenvalue with multiplicity $2^d - d - 1$. □

*Remark.* Property 3 is shown in Figure 1.

**Property 4.** (Mesh networks): Let the eigenvalues of Mesh $\left(n^d\right)$, for $n \geq 4$ even, be $\alpha_1, \cdots, \alpha_n$, with$|\alpha_1| \geq \cdots \geq |\alpha_N|$. Let $N = 2^d$ and $\rho$ be the diameter. Then we observe the following:

(1) $\alpha_1 > 0$ and $|\alpha_1| > |\alpha_2|$;
(2) $\alpha_2 = \cdots = \alpha_{d+1} < 0$;
(3) $|\alpha_{d+1}| > |\alpha_{d+2}|$;
(4) $|\alpha_j| < |\alpha_2| \cdot 10\%$, $j \in [3d + 2, \rho + 1]$ and $n \geq 6$;
(5) $\alpha_i \begin{cases} > 0 & \text{for } i = 1, \\ < 0 & \text{for } 2 \leq i \leq \rho + 1, \\ = 0 & \forall \text{ other } i. \end{cases}$

*Proof.* Property 4(1) is guaranteed by Property 1(1). □

Similarly to Figure 1, Figure 2 also shows the following: (o.4) a mesh network has a single positive dominant eigenvalue, thus implying Property 4(1); (o.5) the trumpet-like curves demonstrate that the second-absolute-largest energy levels are always negative and become larger and larger in comparison to absolute-smaller energy levels, thus implying Properties 4(2) and 4(3). Figures 3 and 4 also confirm Property 4(2) by showing that 2D and 3D mesh networks have two and three second-absolute-largest eigenvalues. Through comparing the data in Figure 2 while realizing observation (o.5), we derive Property 4(4). Property 4(5) is a straightforward summary based on Figure 2.

**Property 5.** (Fully-connected networks): Let the eigenvalues of a fully-connected network $(N)$ be $\alpha_1, \cdots, \alpha_n$, with $|\alpha_1| \geq \cdots \geq |\alpha_N|$. Then we have:

(1) $\alpha_1 = N - 1$ with associated eigenvector $q_1 = \frac{1}{\sqrt{N}} \cdot \mathbf{1}_N$;

(2) $\alpha_j = -1$, $j \in [2, N]$.

*Proof.* Let $S$ be a supply matrix of a fully-connected network. Then $S + I_{N\times N} = 1_{N\times N}$. Clearly, $1_{N\times N}$ has two eigenvalues, $N$ and 0, with multiplicities 1 and $N-1$, respectively. Second, if $\alpha_k$ is an eigenvalue of $1_{N\times N}$, $\alpha_k - 1$ is an eigenvalue of $S$. This proves Property 5. □

*Remark.* The same statement is in [7]. Property 5 is shown in Figure 1.

## 5. Analyses and the numerical experiments

The value of our new formulation is demonstrated analytically and validated numerically through mapping applications on torus networks.

5.1. **Analytical solution of a sample case.** On a small mapping problem, we apply the ER Theorem to find the analytic solution by exploring symmetries of the network's and the application's eigenpairs. The problem involves mapping a 4-task looped communication pattern depicted in Figure 5 to a 2D Torus ($2 \times 2$). In this application, each task exchanges data of equal size with its two nearest neighbors in the ring. Figures 6 and 7 are the supply and demand matrices for the network and the application.

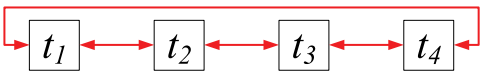


Figure 5. A communication pattern of 4-node ring

| source \ destination | | | | |
|---|---|---|---|---|
| | 0 | 1 | 1 | 2 |
| | 1 | 0 | 2 | 1 |
| | 1 | 2 | 0 | 1 |
| | 2 | 1 | 1 | 0 |

Figure 6. Supply matrix of 2D torus($2 \times 2$)

| senders \ receivers | | | | |
|---|---|---|---|---|
| | 0 | 2 | 0 | 2 |
| | 2 | 0 | 2 | 0 |
| | 0 | 2 | 0 | 2 |
| | 2 | 0 | 2 | 0 |

Figure 7. Demand matrix of a 4-task ring

We notice that

$$\alpha \equiv \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \end{bmatrix} = \begin{bmatrix} 4 \\ -2 \\ -2 \\ 0 \end{bmatrix}, \quad \beta \equiv \begin{bmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \\ \beta_4 \end{bmatrix} = \begin{bmatrix} 4 \\ -4 \\ 0 \\ 0 \end{bmatrix}, \tag{28}$$

$$q_1 = p_1 = \frac{1}{2}\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \Rightarrow q_1^T M = q_1^T \quad \text{and} \quad Mp_1 = p_1 \forall M \in R^{n\times n}.$$

Thus, the hop-bytes metric in the ER Theorem is

$$\begin{aligned} f_{S,D}(M) &= \alpha^T C \beta = [4,-2,-2] \begin{bmatrix} \left(q_1^T p_1\right)^2 & \left(q_1^T p_2\right)^2 \\ \left(q_2^T p_1\right)^2 & \left(q_2^T M p_2\right)^2 \\ \left(q_3^T p_1\right)^2 & \left(q_3^T M p_2\right)^2 \end{bmatrix} \begin{bmatrix} 4 \\ -4 \end{bmatrix} \\ &= [4,-2,-2] \begin{bmatrix} 1 & 0 \\ 0 & \left(q_2^T M p_2\right)^2 \\ 0 & \left(q_3^T M p_2\right)^2 \end{bmatrix} \begin{bmatrix} 4 \\ -4 \end{bmatrix}, \quad \text{where} \end{aligned} \tag{29}$$

$$q_2 = \frac{\sqrt{2}}{2}\begin{bmatrix} 1 \\ 0 \\ 0 \\ -1 \end{bmatrix}, q_3 = \frac{\sqrt{2}}{2}\begin{bmatrix} 0 \\ 1 \\ -1 \\ 0 \end{bmatrix}, p_2 = \frac{1}{2}\begin{bmatrix} 1 \\ -1 \\ 1 \\ -1 \end{bmatrix}.$$

Substituting $q_2' = \sqrt{2}q_2$, $q_3' = \sqrt{2}q_3$, $p_2' = 2p_2$ into this equation yields

$$\begin{aligned} & f_{S,D}(M) = 8\left[\left(q_2^T M p_2\right)^2 + \left(q_3^T M p_2\right)^2\right] + 16 \\ \Rightarrow\ & f_{S,D}(M) = \left(\left(q_2'\right)^T M p_2'\right)^2 + \left(\left(q_3'\right)^T M p_2'\right)^2 + 16 \\ \Rightarrow\ & f_{S,D}(M) = \tfrac{1}{2}\left\{ \left(Mp_2'\right)^T \begin{bmatrix} 1 & 0 & 0 & -1 \\ 0 & 1 & -1 & 0 \\ 0 & -1 & 1 & 0 \\ -1 & 0 & 0 & 1 \end{bmatrix} \left(Mp_2'\right) \right\} + 16 \\ \Rightarrow\ & f_{S,D}(M) = 18 - \tfrac{1}{2}\left\{ \left(Mp_2'\right)^T \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \left(Mp_2'\right) \right\} \\ \Rightarrow\ & f_{S,D}(M) = 18 - (v_1 v_4 + v_2 v_3), \end{aligned} \tag{30}$$

$$\text{where } v \equiv [v_1, v_2, v_3, v_4]^T = Mp_2'.$$

Therefore, this mapping problem can be restated as: Find a mapping matrix $M$ to

$$\text{maximize}\left\{(v_1 v_4 + v_2 v_3)\right\}, \tag{31}$$

$$\text{where } [v_1, v_2, v_3, v_4]^T = M[1,-1,1,-1]^T.$$

Clearly, the optimal mapping condition is $v_1 = v_4 = 1$ and $v_2 = v_3 = -1$, i.e.,

$$M = [e_1, e_2, e_4, e_3] = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}. \tag{32}$$

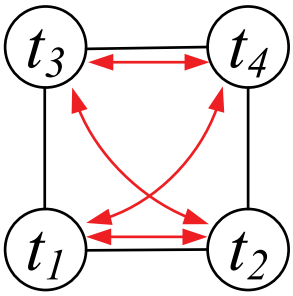


FIGURE 8. Rank-order mapping

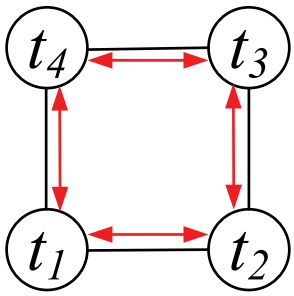


FIGURE 9. Optimal mapping

The optimal permutation and its optimal hop-bytes metric becomes:

$$\sigma_{optimal} = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 1 & 2 & 4 & 3 \end{pmatrix} \quad \text{and} \quad f_{S,D}\left(\sigma_{optimal}\right) = 16. \tag{33}$$

This mapping solution is depicted in Figure 9 and is compared with a more natural rank-order mapping as shown in Figure 8.

The symmetries of the demand matrices in equation (31) reflected by the eigenvector $p_2'$ guarantee the invariance of $f_{S,D}(M)$ when swapping the mapping for the task pair $t_1$ and $t_3$ and for the task pair $t_2$ and $t_4$. The ER Theorem also eliminates the need to consider mapping options that would help find the optimal mapping. These mapping options, enormous in number, are a result of the symmetries of the network and tasks. In our example of mapping 4 tasks to 4 nodes, the number of mapping options is $P_4^4 = 4! = 24$. With our new formulation, we only need to consider 6 mapping options.

5.2. **Objective function truncation.** Task mapping is one of the classic NP-complete problems whose solutions require heuristic techniques such as simulated annealing (SA) to obtain [1, 9, 12]. We have conducted various experiments to examine the extent of solution time reduction and quality through SA that we adopted from SA Tool version 1.03 for Matlab in [12]. In our experiments, we use an exponential annealing schedule: $T_i = T_0 \cdot \eta^i$ where $\eta \approx 1$ [9, 12]. We perform sixteen steps to "melt" the system from an initial "frozen" rank-order mapping. We launch two random walkers in this SA tool and consider 100 pair exchanges in each temperature stage per walker. The SA process is terminated once a predetermined number of steps have been taken or the energies have no significant change over 2,000 steps:

$$|E_t - E_c| / |E_c| < 10^{-3}, \tag{34}$$

where $E_c$ is the latest energy and $E_t$ is any of the accepted energies over the past 2,000 steps.

Generating a trial state and evaluating the energy difference $\delta f = E_{new} - E_{old}$ is the most time-consuming portion of the SA process, and also the most memory-intensive because of the need to store the supply and demand matrices. In a standard SA experiment, the hop-bytes objective function $f$ in Definition 4 is used as the energy function to minimize. Our ER Theorem ($f = \alpha^T C \beta$) plays an essential role in truncating, yet preserving the principle of, the energy function; it is not possible to do this with the graph theory-based energy function.

Considering the eigenspectral properties of the cellular networks in Section 4, we truncate this energy function $f = \alpha^T C \beta$ to a new form by selecting a subset of eigenvalues from $\alpha$ and $\beta$:

$$\hat{f} = \hat{\alpha}^T \hat{C} \hat{\beta}, \tag{35}$$

where $\hat{\alpha} = [\alpha_{i1}, \cdots, \alpha_{il}]^T \in R^l$, $\hat{\beta} = [\beta_{j1}, \cdots, \beta_{jk}]^T \in R^k$, and

$$\hat{C} = \begin{bmatrix} \left(q_{i1}^T M p_{j1}\right)^2 & \cdots & \left(q_{i1}^T M p_{jk}\right)^2 \\ \vdots & \ddots & \vdots \\ \left(q_{il}^T M p_{j1}\right)^2 & \cdots & \left(q_{il}^T M p_{jk}\right)^2 \end{bmatrix} \in R^{l \times k}.$$

Thus, $|\hat{\alpha}| = l < n$, $|\hat{\beta}| = k < n$ and $\hat{C}$ is a submatrix of $C$.

This new truncated energy function $\hat{f}$ helps eliminate unnecessary calculations. For example, the supply matrices for torus and mesh networks contain many zero eigenvalues; terms involving these values can be ignored in forming $\hat{\alpha}$. It also helps approximate the principle changes of the energy function $f$.

For example, in mapping a 2D $8 \times 8$ computing mesh for a wave equation onto a 3D Torus $\left(4^3\right)$ network, we order the eigenvalues of $S$ and $B$ respectively: $|\alpha_1| \geq |\alpha_2| \geq \cdots \geq |\alpha_{64}|$ and $\beta_1 \geq \cdots \geq \beta_{64}$. Then the new energy function $\hat{f}$ can be written as

$$\hat{f} = \hat{\alpha}^T \hat{C} \hat{\beta} = -32 \cdot \mathbf{1}_6^T \cdot \begin{bmatrix} \left(q_2^T M p_2\right)^2 & \cdots & \left(q_2^T M p_5\right)^2 \\ \vdots & \ddots & \vdots \\ \left(q_6^T M p_2\right)^2 & \cdots & \left(q_6^T M p_5\right)^2 \end{bmatrix}_{6 \times 4} \cdot \begin{bmatrix} 5.41 \\ 5.41 \\ 4.83 \\ 4.83 \end{bmatrix}. \tag{36}$$

Thus, $|\hat{\alpha}| = 6$ and $|\hat{\beta}| = 4$ .

In this case, the dominant eigenvalues in both $S$ and $B$ (i.e., $\alpha_1 = 192$ and $\beta_1 = 6$) have the same eigenvectors $q_1 = p_1 = \frac{1}{8} \cdot 1_{64}^T$. Thus, these two eigenvalues' contribution to the energy function is a constant $c$ regardless of mapping:

$$c = \alpha_1 \left(q_1^T P \Lambda_b P^T q_1\right) + \beta_1 \left(p_1^T Q \Lambda_s Q^T p_1\right) - \alpha_1 \beta_1 \left(q_1^T p_1\right)^2 \approx 1152. \tag{37}$$

The relation between the energy function $f$ and its truncated form $\hat{f}$ is $f = \hat{f} + c + s = \hat{f} + 1152 + s$ where $s$ serves as a relaxation.

Replacing the energy function $f$ with this new truncated form $\hat{f}$ produces a new task mapping solver: eigenbased SA (SA-E). We refer to the implementation with the original energy function $f$ as regular SA (SA-B).

### 5.3. **Analysis of heuristic solution effectiveness and efficiency.**

5.3.1. *SA-B vs. SA-E for general cases.* Before presenting numerical solutions, we analyze the efficiency of the SA applied to the two formulations in terms of CPU time, measured in number of floating-point operations for SA steps, and memory space.

In finding a new mapping, we swap a task pair $\{i, j\}$ and SA-B uses

$$E_{new} = E_{old} - 2(s_i - s_j)^T (b_i - b_j) + 4 (s_{ij} \times b_{ij}), \tag{38}$$

where $s_i$ and $b_i$ are the $i$th column vectors of $S$ and $B$, and the same for $j$. This requires $N+3$ multiplications and $3N+1$ additions, totaling $4(N+1)$ operations. Thus, its time complexity is $O(4N)$. This process requires use of the entire supply and demand matrices in the memory so its space complexity is $O\left(N^2\right)$.

However, SA-E requires the following two steps:

Step 1: Calculate $\hat{C}_{new} = [d_{lm}^{new}]_{|\hat{\alpha}|\times|\hat{\beta}|}$ through $\hat{C}_{old} = \left[d_{lm}^{old}\right]_{|\hat{\alpha}|\times|\hat{\beta}|}$ as:

$$d_{lm}^{new} = d_{lm}^{old} - (q_l(i) - q_l(j))(p_m(i) - p_m(j))$$

where

$$l \in \{1, \cdots, |\hat{\alpha}|\}, m \in \left\{1, \cdots, |\hat{\beta}|\right\}.$$

Step 2: Calculate a new truncated energy:

$$E_{new} = \sum_{m=1}^{|\hat{\beta}|} \left[ \beta_m \left( \sum_{l=1}^{|\hat{\alpha}|} \alpha_l (d_{lm}^{new})^2 \right) \right], \tag{39}$$

where $\hat{\alpha} = \left[\alpha_1, \cdots, \alpha_{|\hat{\alpha}|}\right]^T \in R^{|\hat{\alpha}|}$, $\hat{\beta} = \left[\beta_1, \cdots, \beta_{\left|\hat{\beta}\right|}\right]^T \in R^{|\hat{\beta}|}$.

This requires $|\hat{\beta}|(3|\hat{\alpha}|+1)$ multiplications and $4|\hat{\alpha}||\hat{\beta}|-1$ additions, totaling $|\hat{\beta}|(7|\hat{\alpha}|+1)-1$ operations. Thus, its time complexity is $O\left(7|\hat{\alpha}||\hat{\beta}|\right)$. This process requires the use of a matrix $\hat{C}_{old} \in R^{|\hat{\alpha}|\times|\hat{\beta}|}$ and eigenvectors $\{q_l\}_{l=1}^{|\hat{\alpha}|} \in R^N$ and $\{p_m\}_{m=1}^{|\hat{\beta}|} \in R^N$ so its space complexity is $O\left(|\hat{\alpha}||\hat{\beta}| + \left(|\hat{\alpha}| + |\hat{\beta}|\right)N\right)$.

5.3.2. *SA-B vs. SA-E for tori.* In the torus networks, Property 2 in Section 4 implies:

(1) We can ignore $\alpha_1$ in forming $\hat{\alpha}$ since $q_1 = N^{-1/2} \cdot \mathbf{1}_N$ causes $\alpha_1$'s contribution to the energy function to be a constant under any map, by Property 2(1);
(2) $\alpha_2 = \cdots = \alpha_{2d+1} < 0$ are the $2d$ smallest eigenvalues, by Property 2(3);
(3) $|\alpha_2|$ becomes more significant than other negative eigenvalues as $N$ grows larger, by Property 2(5).

Therefore, in mapping applications onto Torus $\left(n^d\right)$, if we only select the $2d$ smallest eigenvalues for forming $\hat{\alpha}$, i.e., $\hat{\alpha} = [\alpha_2, \cdots, \alpha_{2d+1}]^T \in R^{2d}$, then the truncated energy in Step 2 becomes:

$$E_{new} = \alpha_2 \cdot \sum_{m=1}^{|\hat{\beta}|} \left[ \beta_m \left( \sum_{l=1}^{|\hat{\alpha}|} (d_{lm}^{new})^2 \right) \right]. \tag{40}$$

Thus, SA-E for torus requires $\left|\hat{\beta}\right|(2|\hat{\alpha}|+1)$ multiplications and $4|\hat{\alpha}|\left|\hat{\beta}\right| - 1$ additions, totaling $\left|\hat{\beta}\right|(6|\hat{\alpha}|+1) - 1$ operations. Its time complexity is $O\left(12d \cdot \left|\hat{\beta}\right|\right)$.

For example, we consider mapping applications on 3D Torus $\left(n^3\right)$ networks. Table 2 compares the complexities between the SA-B and SA-E approaches. In SA-E, we use two selection methods for $\hat{\alpha}$: all of the negative eigenvalues to get $\hat{\alpha} = [\alpha_2, \cdots, \alpha_{1.5\times n+1}] \in R^{1.5\times n}$ (SA-E-1) and only the $2d$ (six in this case) smallest eigenvalues to get $\hat{\alpha} = [\alpha_2, \cdots, \alpha_7] \in R^6$ (SA-E-2).

5.3.3. *SA-B vs. SA-E for hypercubes.* In the hypercube networks, Property 3 in Section 4 implies:

(1) We can ignore $\alpha_1$ in forming $\hat{\alpha}$ since $q_1 = N^{-1/2} \cdot \mathbf{1}_N$ causes $\alpha_1$'s contribution to the energy function to be a constant under any map, by Property 3(1); (This is the same as for tori.)

TABLE 2. Complexity comparisons between the SA-B and SA-E approaches for torus networks

| Methods | Time Complexity | Space Complexity | Note |
|---|---|---|---|
| SA-B | $4N$ | $N^2$ | |
| SA-E-1 | $10.5\left\|\hat{\beta}\right\| \cdot N^{1/3}$ | $\left(6+\left\|\hat{\beta}\right\|\right) N$ | $\|\hat{\alpha}\| = 1.5 \times n$ |
| SA-E-2 | $36\left\|\hat{\beta}\right\|$ | $\left(6+\left\|\hat{\beta}\right\|\right) N$ | $\|\hat{\alpha}\| = 6$ |

(2) $\alpha_2 = \cdots = \alpha_{d+1} = -2^{d-1} < 0$ are the $d$ second-absolute-largest eigenvalues, by Property 3(2);
(3) We can ignore $\alpha_j$ for $j \in [d+2, N]$ in forming $\hat{\alpha}$ since $|\alpha_j| \equiv 0$ for $j \in [d+2, N]$, by Property 3(3).

Therefore, in mapping applications onto Hypercube $\left(2^d\right)$, we need only select the $d$ negative eigenvalues for forming $\hat{\alpha}$, i.e., $\hat{\alpha} = [\alpha_2, \cdots, \alpha_{d+1}]^T \in R^d$. Then the truncated energy in Step 2 becomes:

$$E_{new} = -2^{d-1} \cdot \sum_{m=1}^{\left|\hat{\beta}\right|} \left[ \beta_m \left( \sum_{l=1}^{|\hat{\alpha}|} (d_{lm}^{new})^2 \right) \right]. \tag{41}$$

Similarly to the torus case, its time complexity is $O\left(6d \cdot \left|\hat{\beta}\right|\right)$.

For example, we consider mapping applications on $d$-D Hypercube $\left(2^d\right)$ networks. Table 3 compares the complexities between the SA-B and SA-E approaches. We can see that the time and space complexities of the SA-E approaches for 6-D hypercube networks are the same for 3-D torus networks.

TABLE 3. Complexity comparisons between the SA-B and SA-E approaches for hypercube networks

| Methods | Time Complexity | Space Complexity | Note |
|---|---|---|---|
| SA-B | $4N$ | $N^2$ | |
| SA-E | $6d \cdot \left\|\hat{\beta}\right\|$ | $\left(d+\left\|\hat{\beta}\right\|\right) N$ | $\|\hat{\alpha}\| = d$ |

5.3.4. *Error estimates.* Sections 5.3.2 and 5.3.3 present the truncated objective functions and analyze and compare their time and space efficiencies. Before showing numerical results, we analyze the estimates of the error between the original objective function as in equation (20) and our truncated models in equations (40) and (41).

In a step of a simulated annealing experiment for the task mapping problem, we randomly select a task pair $(t_1, t_2)$ and exchange them in the current mapping $M_1$ to generate a new mapping $M_2$. As stated in Section 5.3.1, the acceptance criteria for this "move" require evaluation of the difference $\delta E \equiv E(M_1) - E(M_2)$ of the objective functions $E(M_1)$ and $E(M_2)$ under two mappings $M_1$ and $M_2$.

ER Theorem proved that the objective function $E_o$ in SA-B is:

$$E_o(M) \equiv \sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\beta_j\left(q_i^T M p_j\right)^2. \tag{42}$$

Thus, we have $\delta E$ in SA-B as $\delta E_o = E_o(M_1) - E_o(M_2)$.

Similarly, equation (35) presents the truncated objective function $E_t$ in SA-E as

$$E_t(M) = \sum_{\forall i\in|\hat{\alpha}|}\sum_{\forall j\in\left|\hat{\beta}\right|}\alpha_i\beta_j\left(q_i^T M p_j\right)^2. \tag{43}$$

Thus, we have $\delta E$ in SA-B as $\delta E_t = E_t(M_1) - E_t(M_2)$.

In the hypercube $H\left(2^d\right)$ case, we have: $\hat{\alpha} = [\alpha_2, \cdots, \alpha_{d+1}]$ and $\hat{\beta} = [\beta_1, \cdots, \beta_k]$ so $|\hat{\alpha}| \equiv d$ and $\left|\hat{\beta}\right| = k$. According to Property 3, under any mapping $M$, we have:

$$\begin{aligned} E_o(M) &= c_0 + E_t(M) - 2^{d-1}\sum_{i=2}^{d+1}\sum_{j=k+1}^{N}\beta_j\left(q_i^T M p_j\right)^2 \\ &\text{in which} \quad c_0 = \frac{d}{2}\sum_{j=1}^{N}\beta_j\left|1_N^T\cdot p_j\right|^2. \end{aligned} \tag{44}$$

Here, $c_0$ is the contribution from $\alpha_0$ invariant under any mapping and $N = 2^d$. Then, we have:

$$\begin{aligned} \delta E_o - \delta E_t &= \left(E_o(M_1) - E_t(M_1)\right) - \left(E_o(M_2) - E_t(M_2)\right) \\ &= -2^{d-1}\sum_{i=2}^{d+1}\sum_{j=k+1}^{N}\beta_j\left(q_i^T(M_1+M_2)p_j\right)\left(q_i^T(M_1-M_2)p_j\right). \end{aligned} \tag{45}$$

On the other hand, since $||q_i|| = ||p_j|| = 1$, we have:

$$\left|q_i^T(M_1 - M_2)p_j\right| = \left|\left(p_i[t_1] - p_i[t_2]\right)\left(q_j[t_1] - q_j[t_2]\right)\right| \le 1, \tag{46}$$

$$\left|q_i^T(M_1 + M_2)p_j\right| \le \left|q_i^T M_1 p_j\right| + \left|q_i^T M_2 p_j\right| \le 2. \tag{47}$$

From equations (45), (46) and (47), we obtain:

$$|\delta E_o - \delta E_t| \le d2^d\sum_{j=k+1}^{N}|\beta_j| \tag{48}$$

Therefore, in the hypercube case, we prove that the difference $|\delta E_o - \delta E_t|$ between $\delta E_o$ obtained by considering the original objective function as in equation (20) and $\delta E_t$ obtained by considering only a subset of eigenvalues for an application demand matrix and all eigenvalues of a given hypercube $\left(2^d\right)$ as in equation (40) is bounded by the set of eigenvalues that are not considered in the truncated objective function and the network size.

In a torus $T\left(n^d\right)$ case, we have: $\hat{\alpha} = \left[\alpha_2, \cdots, \alpha_{|\hat{\alpha}|+1}\right]$ and $\hat{\beta} = [\beta_1, \cdots, \beta_k]$ so $|\hat{\beta}| = k$. According to Property 2, under any mapping $M$, we have:

$$E_o(M) = c_0 + E_t(M) + \sum_{i=2}^{|\hat{\alpha}|+1} \sum_{j=k+1}^{N} \alpha_i \beta_j \left(q_i^T M p_j\right)^2$$

$$\text{in which} \quad c_0 = \frac{nd}{4} \sum_{j=1}^{N} \beta_j \left|\mathbf{1}_N^T \cdot p_j\right|^2. \tag{49}$$

Here, similarly to equation (44), $c_0$ is the contribution from $\alpha_0$, invariant under any mapping, and $N = n^d$. Then similarly to equation (45), we have:

$$\begin{aligned} \delta E_o - \delta E_t &= \left(E_o(M_1) - E_t(M_1)\right) - \left(E_o(M_2) - E_t(M_2)\right) \\ &= \sum_{i=2}^{|\hat{\alpha}|} \sum_{j=k+1}^{N} \alpha_i \beta_j \left(q_i^T (M_1 + M_2) p_j\right) \left(q_i^T (M_1 - M_2) p_j\right). \end{aligned} \tag{50}$$

According to equations (46) and (47), we know:

$$\left|\left(q_i^T (M_1 + M_2) p_j\right) \left(q_i^T (M_1 - M_2) p_j\right)\right| \leq 2. \tag{51}$$

Then, according to Property 2, we have:

$$|\delta E_o - \delta E_t| \leq 2 |\hat{\alpha}| |\alpha_2| \sum_{j=k+1}^{N} |\beta_j| \leq \frac{n^{d+1}}{4} |\hat{\alpha}| \sum_{j=k+1}^{N} |\beta_j| \leq \frac{3}{8} n^{d+2} \sum_{j=k+1}^{N} |\beta_j|. \tag{52}$$

Therefore, in the torus case, we also prove that $|\delta E_o - \delta E_t|$ is bounded by the set of eigenvalues that are not considered in the truncated objective function and the network size.

Equations (48) and (52) imply that the larger $\beta_j$ contribute more heavily to the objective function, and its move changes, in the simulated annealing algorithm. In other words, contributions from small $\beta_j$ may be ignored; we can approximate a near-optimal task mapping on supercomputers with reasonable cost in terms of time and space complexities rather than demanding a perfectly optimal solution regardless of expenses.

5.3.5. *Numerical results.* Figure 10 shows the complexity of space and time of mapping applications onto 3D Torus $\left(n^3\right)$ networks. It is obvious that proper choice of $\hat{\beta}$ in SA-E greatly reduces the time and memory demands.

In Figures 11 to 19 and Table 2, we test mapping the tasks of a 2D wave equation onto a 3D Torus network with the regular (SA-B) and the new eigen method (SA-E). In Figures 20 to 23 and Table 3, we compare results for mapping a 2D wave equation onto a 6D Hypercube network with the regular (SA-B) and the new eigen method (SA-E). The inter-task communication pattern of solving the wave equation on a mesh $16 \times 4$ is shown in Figure 24.

In Figures 11 to 17, the two digits in parentheses following SA-E indicate the dimensions of $\hat{\alpha}$ and $\hat{\beta}$. For example, SA-E (6, 4) in Figure 11 means $|\hat{\alpha}| = 6$ and $|\hat{\beta}| = 4$. Since all of the tested $B$'s have a largest eigenvalue $\beta_1$ with corresponding eigenvector $p_1 = N^{-1/2} \cdot \mathbf{1}_N$ ($N$ is the problem size), we always ignore this first eigenvalue in selecting the eigenvalues in forming $\hat{\beta}$. For example, in Figure 11, SA-E (6, 4) implies that $\hat{\beta} = [\beta_2, \cdots, \beta_5]^T \in R^4$ and $|\hat{\beta}| = 4$. Second, we do not select any eigenvalue with multiplicity $\geq 4$ in some cases and we identify them as SA-E ($|\hat{\alpha}|$, selected $|\hat{\beta}|$). For example, in Figure 12, SA-E (6, selected 11) implies

that $\hat{\beta}$ contains the first eleven eigenvalues with multiplicities less than 4 but in the same figure, SA-E (6, 14) implies that $\hat{\beta} = [\beta_2, \cdots, \beta_{15}]^T$ without considering the multiplicity.

In Figures 11 to 14 and 16, we select all of the negative eigenvalues in forming $\hat{\alpha}$. Thus, $|\hat{\alpha}| = 6$ for 3D Torus $\left(3^3\right)$ and Torus $\left(4^3\right)$, and $|\hat{\alpha}| = 9$ for 3D Torus $\left(6^3\right)$. In Figures 15 and 17, we select only the smallest six negative eigenvalues and thus $|\hat{\alpha}| = 6$ for 3D Torus $\left(6^3\right)$. In Figures 20 and 21, we always select six negative eigenvalues for a 6-D Hypercube network, thus $|\hat{\alpha}| = 6$.

These numerical solutions show the following:

(1) SA-E converges faster as the problem size increases. In Figures 18 and 19, most cases require fewer operations to reach lower best-so-far (BSF) energies than SA-B. The same is in Figures 22 and 23.

(2) SA-E is superior in escaping from local minima. In Figures 11 to 13 and 20 to 21, SA-E reached the global minimum; SA-B failed in Figures 12, 13, 20, and 21. In a large problem shown in Figures 14 and 16, SA-E reached BSF energies 22% and 30% lower than those reached by SA-B.

(3) Properly decreasing the eigenvalue count in $\hat{\alpha}$ while increasing the eigenvalue count in $\hat{\beta}$ reduces time complexity but does not decrease accuracy greatly. For example, in mapping 2D wave equation $(18 \times 12)$ onto 3D Torus $\left(6^3\right)$ in Table 4, the case of SA-E (9, 14) requires 32% more operations than that of SA-E (6, 16) per energy update. However, both cases reach the same BSF energies.

(4) Properly increasing the eigenvalue count in $\hat{\beta}$ increases the probability of reaching lower BSF energies.

(5) All SA-E cases require much less memory than their SA-B counterparts, as shown in Tables 2 and 3.

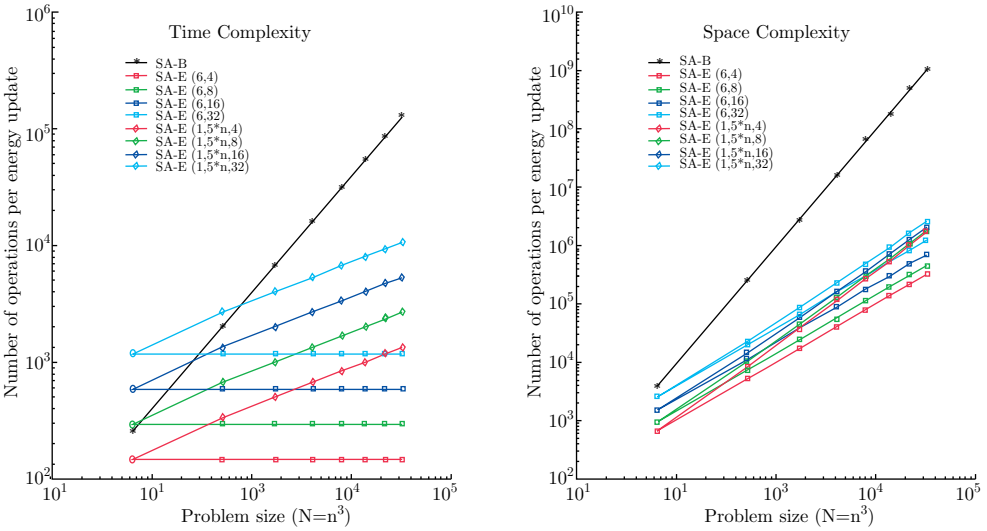


FIGURE 10. Algorithmic complexity of mapping applications on 3D torus $\left(N = n^3\right)$ of equal size (SA-E $(x, y)$ indicates $x = |\hat{\alpha}|$ and $y = |\hat{\beta}|$)

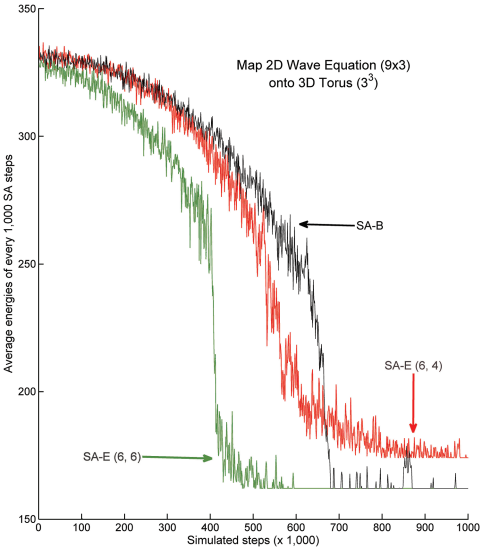


FIGURE 11. Map 2D wave equation $(9 \times 3)$ onto 3D torus $\left(3^3\right)$ $\left(|\hat{\alpha}| = 6, |\hat{\beta}| \in \{4, 6\}\right)$

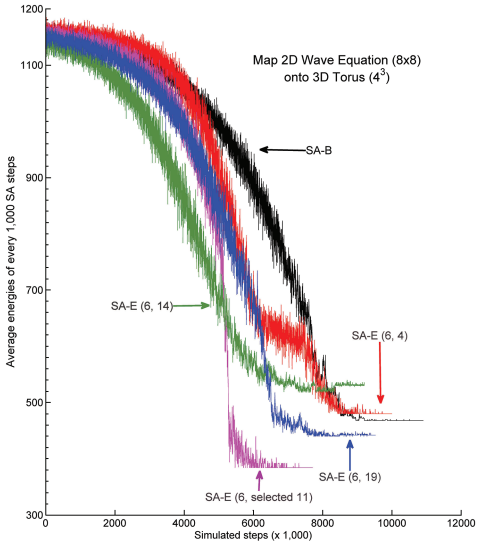


FIGURE 12. Map 2D wave equation $(8 \times 8)$ onto 3D torus $\left(4^3\right)$ $\left(|\hat{\alpha}| = 6, |\hat{\beta}| \in \{4, 11, 14, 19\}\right)$

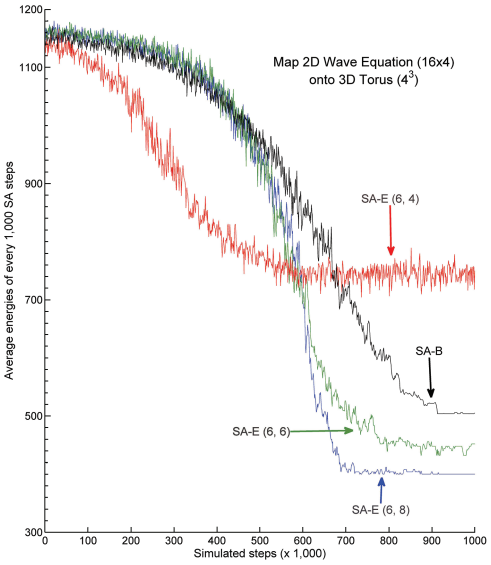


FIGURE 13. Map 2D wave equation $(16 \times 4)$ onto 3D torus $\left(4^3\right)$ $\left(|\hat{\alpha}| = 6, |\hat{\beta}| \in \{4, 6, 8\}\right)$

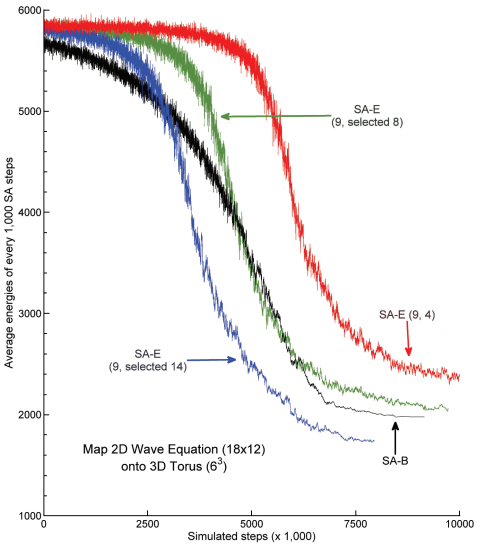


FIGURE 14. Map 2D wave equation $(18 \times 12)$ onto 3D torus $\left(6^3\right)$ $\left(|\hat{\alpha}| = 9, |\hat{\beta}| \in \{4, 8, 14\}\right)$

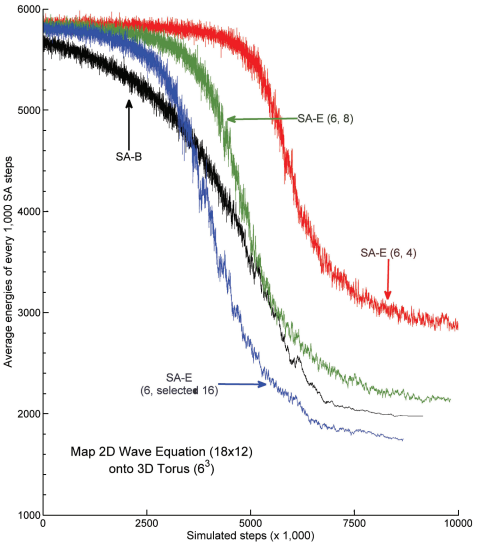


FIGURE 15. Map 2D wave equation $(18 \times 12)$ onto 3D torus $\left(6^3\right)$ $\left(|\hat{\alpha}| = 6, |\hat{\beta}| \in \{4, 8, 16\}\right)$

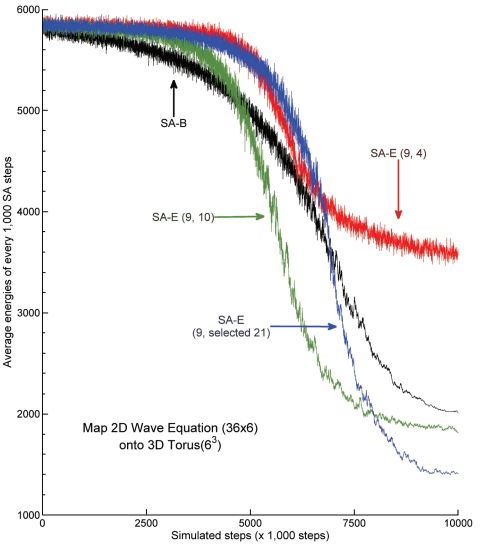


FIGURE 16. Map 2D wave equation $(36 \times 6)$ onto 3D torus $\left(6^3\right)$ $\left(|\hat{\alpha}| = 9, |\hat{\beta}| \in \{4, 10, 21\}\right)$

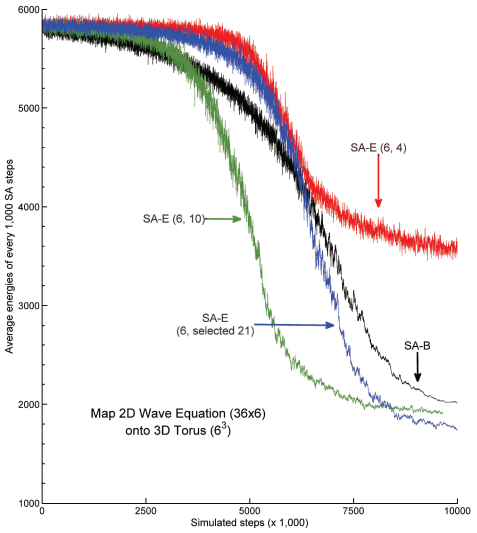


FIGURE 17. Map 2D wave equation $(36 \times 6)$ onto 3D torus $\left(6^3\right)$ $\left(|\hat{\alpha}| = 6, |\hat{\beta}| \in \{4, 10, 21\}\right)$

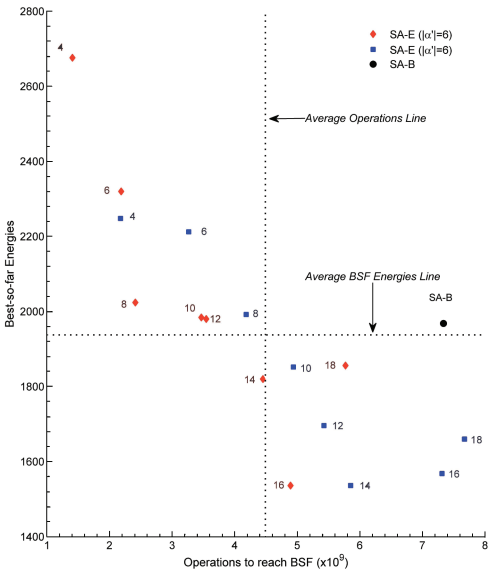


FIGURE 18. Algorithmic efficiency vs. accuracy for mapping 2D wave equation $(18 \times 12)$ onto 3D torus $\left(6^3\right)$ (A red diamond means SA-E with $|\hat{\alpha}| = 6$ near which a red number indicates $|\hat{\beta}|$. A blue square means SA-E with $|\hat{\alpha}| = 9$ near which a blue number indicates $|\hat{\beta}|$.)

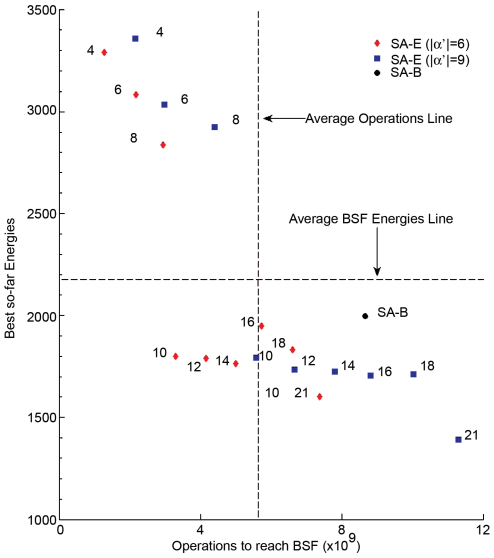


FIGURE 19. Algorithmic efficiency vs. accuracy for mapping 2D wave equation $(36 \times 6)$ onto 3D torus $\left(6^3\right)$ (A red diamond means SA-E with $|\hat{\alpha}| = 6$ near which a red number indicates $|\hat{\beta}|$. A blue square means SA-E with $|\hat{\alpha}| = 9$ near which a blue number indicates $|\hat{\beta}|$.)

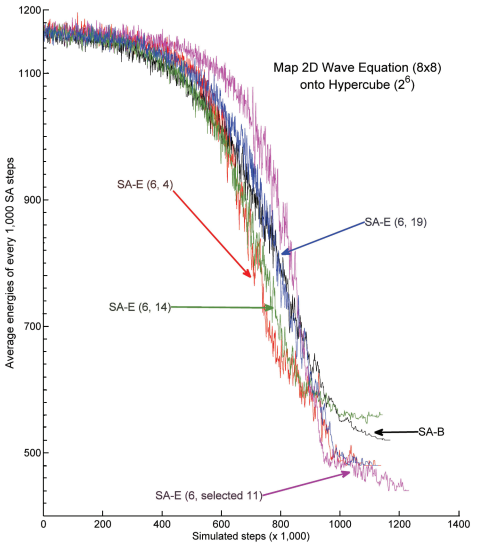


FIGURE 20. Map 2D wave equation $(8 \times 8)$ onto 6D hypercube $\left(2^6\right)$ $\left(|\hat{\alpha}| = 6, |\hat{\beta}| \in \{4, 11, 14, 19\}\right)$

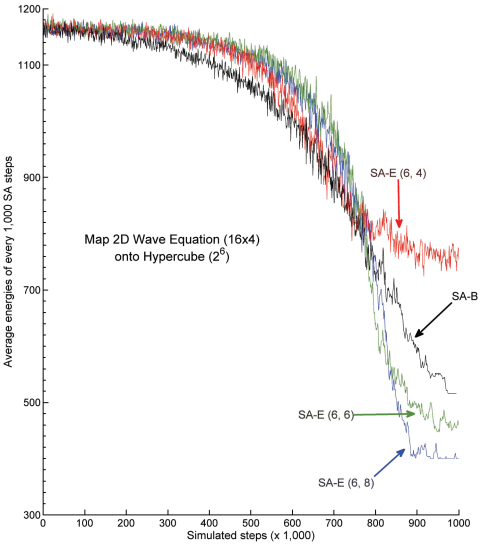


FIGURE 21. Map 2D wave equation $(16 \times 4)$ onto 6D hypercube $\left(2^6\right)$ $\left(|\hat{\alpha}| = 6, |\hat{\beta}| \in \{4, 6, 8\}\right)$

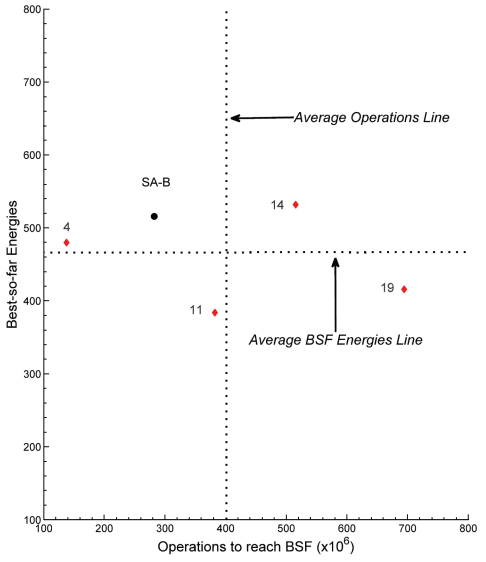


FIGURE 22. Algorithmic efficiency vs. accuracy for mapping 2D wave equation $(8 \times 8)$ onto 6D hypercube $\left(2^6\right)$ (A red diamond means SA-E with $|\hat{\alpha}| = 6$ near which a red number indicates $|\hat{\beta}|$.)

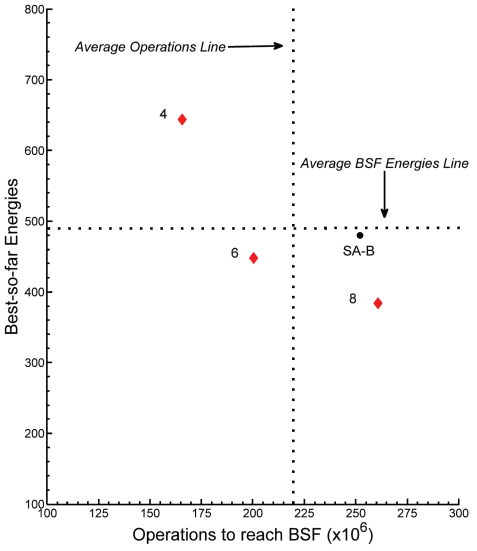


FIGURE 23. Algorithmic efficiency vs. accuracy for mapping 2D wave equation $(16 \times 4)$ onto 3D hypercube $\left(2^6\right)$ (A red diamond means SA-E with $|\hat{\alpha}| = 6$ near which a red number indicates $|\hat{\beta}|$.)

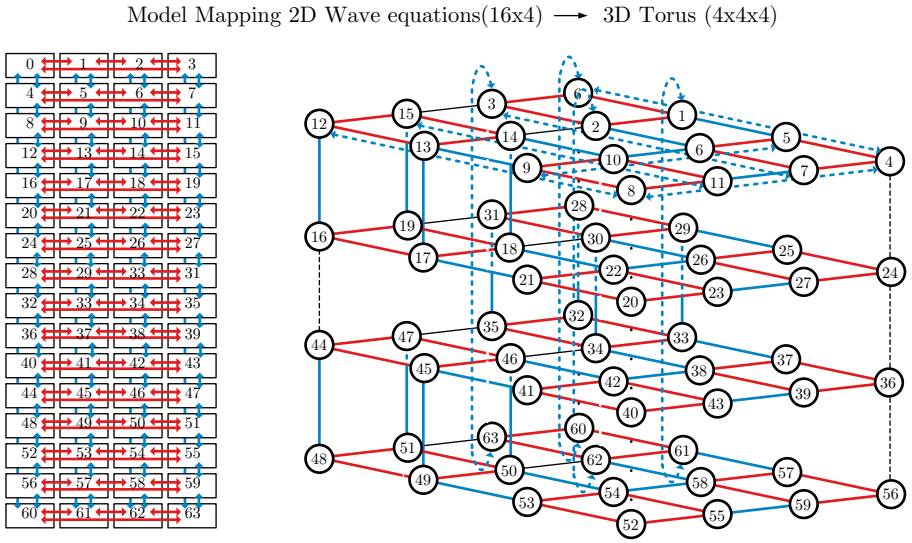


FIGURE 24. Mapping result of 2D wave equation $(16 \times 4)$ onto 3D torus $(4^3)$

TABLE 4. Simulated annealing experiments on the mapping problems

| Mapping Problems | | Inputs | | | Accuracy Analysis | | | | Efficiency Analysis | | | | Figures |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Dimensions of 2D Wave Equation | Networks *T: tours* *H: hypercube* | Eigenvalues $\lvert\tilde{\alpha}\rvert$ | Eigenvalues $\lvert\beta\rvert$ | Size $N$ | BSF energy SA-E | BSF energy SA-B | Steps to reach BSF energy ($\times 10^6$) SA-E | Steps to reach BSF energy ($\times 10^6$) SA-B | Time Complexity SA-E | Time Complexity SA-B | Space Complexity SA-E | Space Complexity SA-B | |
| $9 \times 3$ | $T(3^3)$ | 6 | 4 | 27 | 174 | 162 | 0.565 | 0.666 | 147 | 112 | 294 | 729 | Figure 11 |
| | | | 6 | | 162 | | 0.417 | | 221 | | 360 | | |
| $8 \times 8$ | $T(4^3)$ | 6 | 4 | 64 | 480 | 448 | 7.79 | 10.91 | 147 | 260 | 664 | 4,096 | Figure 12 |
| | | | 11 [A] | | 384 | | 7.72 | | 406 | | 1,154 | | |
| | | | 14 | | 488 | | 8.34 | | 517 | | 1,364 | | |
| | | | 19 | | 384 | | 9.53 | | 702 | | 1,714 | | |
| $16 \times 4$ | | 6 | 4 | | 652 | 488 | 0.652 | 0.619 | 147 | | 664 | | Figure 13 |
| | | | 6 | | 416 | | 0.874 | | 221 | | 804 | | |
| | | | 8 | | 384 | | 0.630 | | 295 | | 944 | | |
| $8 \times 8$ | $H(2^6)$ | 6 | 4 | 64 | 480 | 516 | 0.937 | 1.09 | 147 | 260 | 664 | 4,096 | Figure 20 |
| | | | 11 [A] | | 384 | | 0.941 | | 406 | | 1,154 | | |
| | | | 14 | | 532 | | 1.00 | | 517 | | 1,364 | | |
| | | | 19 | | 416 | | 0.989 | | 702 | | 1,714 | | |
| $16 \times 4$ | | | 4 | | 644 | 480 | 1.13 | 0.969 | 147 | | 664 | | Figure 21 |
| | | | 6 | | 448 | | 0.906 | | 221 | | 804 | | |
| | | | 8 | | 384 | | 0.884 | | 295 | | 944 | | |
| $18 \times 12$ | $T(6^3)$ | 9 | 4 | 216 | 2,248 | 1,968 | 9.76 | 8.45 | 223 | 868 | 2,844 | 46,656 | Figure 14 |
| | | | 8 [A] | | 1,992 | | 9.36 | | 447 | | 3,744 | | |
| | | | 14 [A] | | 1,536 | | 7.48 | | 783 | | 5,094 | | |
| | | 6 [B] | 4 | | 2,676 | | 9.57 | | 147 | | 2,184 | | Figure 15 |
| | | | 8 [A] | | 2,024 | | 8.18 | | 295 | | 3,072 | | |
| | | | 16 [A] | | 1,536 | | 8.28 | | 591 | | 4,848 | | |
| $36 \times 6$ | | 9 | 4 | | 3,360 | 1,996 | 9.69 | 9.99 | 223 | | 2,844 | | Figure 16 |
| | | | 10 | | 1,792 | | 9.98 | | 559 | | 4,194 | | |
| | | | 21 [A] | | 1,392 | | 9.62 | | 1,175 | | 6,669 | | |
| | | 6 [B] | 4 | | 3,292 | | 8.67 | | 147 | | 2,184 | | Figure 17 |
| | | | 10 | | 1,800 | | 8.94 | | 369 | | 3,516 | | |
| | | | 21 [A] | | 1,600 | | 9.50 | | 776 | | 5,958 | | |

NOTE: [A]: Eigenvalues whose multiplicities are no less than 4 are ignored.
[B]: Only the six smallest negative eigenvalues of equal magnitude, i.e., $\alpha_2$ to $\alpha_7$ are taken into account, while ignoring the rest three negative eigenvalues, i.e., $\alpha_8$ to $\alpha_{10}$.

## 6. Conclusions

A new eigenanalysis-based formulation of the well-studied task mapping problem is presented. Theoretical analysis on common sample cases and numerical experiments on more elaborate cases for task mapping demonstrate the key value of our new formulation. The new objective function can be truncated methodically to significantly reduce the required computing resources for achieving mapping solutions of no lesser quality. Additionally, the new formulation allows the optimization process to escape from local minima more effectively.

## References

[1] Michael Affenzeller and Rene Mayrhofer, *Generic heuristics for combinatorial optimization problems*, Proc. of the 9th International Conference on Operational Research 2002, 2002, pp. 83–92.

[2] T. Agarwal, Amit Sharma, A. Laxmikant, and Laxmikant V. Kala, *Topology-aware task mapping for reducing communication contention on large parallel machines*, 20th International Parallel and Distributed Processing Symposium (IPDPS 2006), Proceedings, 25-29 April 2006, Rhodes Island, Greece, IEEE, 2006.

[3] George Almasi, Siddhartha Chatterjee, Alan Gara, John A. Gunnels, Manish Gupta, Amy Henning, Josa E. Moreira, and Robert Walkup, *Unlocking the performance of the bluegene/l supercomputer*, Proceedings of the ACM/IEEE SC2004 Conference on High Performance Networking and Computing, 6-12 November 2004, Pittsburgh, PA, USA, CD-Rom, IEEE Computer Society, 2004, p. 57.

[4] Gyan Bhanot, Alan Gara, Philip Heidelberger, Eoin Lawless, James C. Sexton, and Robert Walkup, *Optimizing task layout on the blue gene/l supercomputer*, IBM Journal of Research and Development **49** (2005), no. 2-3, 489–500.

[5] Shahid H. Bokhari, *On the mapping problem*, IEEE Trans. Comput. **30** (1981), no. 3, 207–214, DOI 10.1109/TC.1981.1675756. MR608522 (82b:68055)

[6] Yongzhi Chen and Yuefan Deng, *Task mapping on supercomputers with cellular networks*, Computer Physics Communications **179** (2008), no. 7, 479–485.

[7] M.A.M. de Aguiar and Y. Bar-Yam, *Spectral analysis and the dynamic response of complex networks*, Physical Review E **71(1)** (2005), 016106.

[8] Gerd Finke, Rainer E. Burkard, and Franz Rendl, *Quadratic assignment problems*, Surveys in combinatorial optimization (Rio de Janeiro, 1985), North-Holland Math. Stud., vol. 132, North-Holland, Amsterdam, 1987, pp. 61–82, DOI 10.1016/S0304-0208(08)73232-8. MR878775 (88e:90057)

[9] S. Kirkpatrick, C. D. Gelatt Jr., and M. P. Vecchi, *Optimization by simulated annealing*, Science **220** (1983), no. 4598, 671–680, DOI 10.1126/science.220.4598.671. MR702485 (85f:90091)

[10] Panos M. Pardalos, Franz Rendl, and Henry Wolkowicz, *The quadratic assignment problem: a survey and recent developments*, Quadratic assignment and related problems (New Brunswick, NJ, 1993), DIMACS Ser. Discrete Math. Theoret. Comput. Sci., vol. 16, Amer. Math. Soc., Providence, RI, 1994, pp. 1–42. MR1290345 (95f:90040)

[11] P. Sadayappan and Fikret Eraal, *Nearest-neighbor mapping of finite element graphs onto processor meshes*, IEEE Transactions on Computers **36** (1987), no. 12, 1408–1424.

[12] Peter Salamon, Paolo Sibani, and Richard Frost, *Facts, conjectures, and improvements for simulated annealing*, SIAM Monographs on Mathematical Modeling and Computation, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2002. MR1945007 (2004k:90006)

[13] Oliver Sinnen, *Task scheduling for parallel systems (Wiley series on parallel and distributed computing)*, Wiley-Interscience, 2007.

[14] Brian E. Smith and Brett Bode, *Performance effects of node mappings on the IBM bluegene/l machine*, Euro-Par 2005, Parallel Processing, 11th International Euro-Par Conference, Lisbon, Portugal, August 30 - September 2, 2005, Proceedings (Josa C. Cunha and Pedro D. Medeiros, eds.), Lecture Notes in Computer Science, vol. 3648, Springer, 2005, pp. 1005–1013.

[15] M. R. F. Smyth, *A spectral theoretic proof of Perron-Frobenius*, Math. Proc. R. Ir. Acad. **102A** (2002), no. 1, 29–35, DOI 10.3318/PRIA.2002.102.1.29. MR1930810 (2003g:15023)

[16] Lee Soo-Young and J.K. Aggarwal, *A mapping strategy for parallel processing*, IEEE Transactions on Computers **36** (1987), 433–442.

[17] Hao Yu, I-Hsin Chung, and Josa E. Moreira, *Blue gene system software - topology mapping for blue gene/l supercomputer*, Proceedings of the ACM/IEEE SC2006 Conference on High Performance Networking and Computing, November 11-17, 2006, Tampa, FL, USA, ACM Press, 2006, p. 116.

Department of Applied Mathematics and Statistics, Stony Brook University, Stony Brook, New York 11794

*E-mail address*: Peng.Zhang@stonybrook.edu

Department of Applied Mathematics and Statistics, Stony Brook University, Stony Brook, New York 11794

*E-mail address*: Yuxiang.Gao@stonybrook.edu

Department of Mathematics and Computer Science, La Salle University, Philadelphia, Pennsylvania 19141

*E-mail address*: fierson@lasalle.edu

Department of Applied Mathematics and Statistics, Stony Brook University, Stony Brook, New York 11794

*E-mail address*: Yuefan.Deng@stonybrook.edu