Computational Optimization and Applications 8, 173–196 (1997)

A Hypergraph Framework for Optimal Model-Based Decomposition of Design Problems

NESTOR F. MICHELENA AND PANOS Y. PAPALAMBROS
Department of Mechanical Engineering and Applied Mechanics, The University of Michigan, 2250 G.G. Brown Bldg., Ann Arbor, MI 48109-2125

Received April 21, 1995; Revised February 2, 1996; Accepted March 21, 1996

Abstract. Decomposition of large engineering system models is desirable since increased model size reduces reliability and speed of numerical solution algorithms. The article presents a methodology for optimal model-based decomposition (OMBD) of design problems, whether or not initially cast as optimization problems. The overall model is represented by a hypergraph and is optimally partitioned into weakly connected subgraphs that satisfy decomposition constraints. Spectral graph-partitioning methods together with iterative improvement techniques are proposed for hypergraph partitioning. A known spectral K-partitioning formulation, which accounts for partition sizes and edge weights, is extended to graphs with also vertex weights. The OMBD formulation is robust enough to account for computational demands and resources and strength of interdependencies between the computational modules contained in the model.

Keywords: model decomposition, multidisciplinary design, hypergraph partitioning, large-scale design, decomposition

Nomenclature

$\mathbf{A}$	= adjacency matrix
$C(P^K)$	= total weight of hyperedges cut by partition P^K
$\mathbf{D}$	= diagonal degree matrix
$deg(i)$	= degree of vertex v_i
$diag(\ )$	= diagonal matrix of vector components
$E_H^c(P^K)$	= set of hyperedges cut by partition P^K
E_G	= set of edges in graph G
E_H	= set of hyperedges in hypergraph H
e_i	= hyperedge or edge
ew	= entrywise
f	= objective function
FDT	= functional dependence table
FM	= Fiduccia-Mattheyses partitioning algorithm
G	= graph
$\mathbf{g}$	= vector of inequality constraints
GDP	= general design problem
H	= hypergraph

$\mathbf{h}$ = vector of equality constraints
$\mathbf{I}_n$ = $n \times n$ identity matrix
K = number of partitions
KL = Kernighan-Lin partitioning algorithm
$\mathbf{L}$ = Laplacian matrix
$\lambda_i(\,)$ = eigenvalue of matrix, such that $\lambda_i(\,) \leq \lambda_{i+1}(\,)$
$\mathbf{M}$ = *diag*($\mathbf{m}$)
M = number of hyperedges
$\mathbf{m}$ = partition load (or size) vector
MDO = multidisciplinary optimization
N = number of vertices
ODP = optimal design problem
OMBD = optimal model-based decomposition
P = number of terminals in hypergraph, or set of partition representatives
$\mathbf{P}$ = partition matrix
p = cardinality of a hyperedge
P^K = partition of a set into K disjoint subsets $V_1, \ldots, V_K$
$\mathbf{R}_K$ = orthonormal $K \times (K-1)$ matrix spanning $\{M^{1/2}\mathbf{u}_K\}^{\perp}$
$\mathbf{S}_N$ = orthonormal $N \times (N-1)$ matrix spanning $\{\mathbf{w}\}^{\perp}$
tr( ) = trace of matrix
$\mathbf{u}_K$ = $(1, \ldots, 1)^t \in \Re^K$
V = set of vertices
v_i = vertex
$\mathbf{W}$ = *diag*($\mathbf{w}$)
$\mathbf{w}$ = $(\omega_v(v_1), \ldots, \omega_v(v_N))^t$
ω = edge weight for hyperedge model
$\omega_e(e_j)$ = weight of hyperedge e_j
$\omega_v(v_i)$ = weight of vertex v_i
$\mathbf{X}$ = assignment matrix of a partition
X = set constraint
$\mathbf{x}$ = vector of design & state/behavior variables x_i
$\mathbf{X}_f$ = feasible assignment matrix of a partition
$\mathbf{x}_k$ = vector of variables local to partition k
$\mathbf{y}$ = vector of linking variables
$\|\Lambda\|_F$ = Frobenius norm of matrix Λ
$\mathfrak{J}$ = set of feasible assignment matrices $\mathbf{X}$
$\Re^M$ = M-dimensional Euclidean space

1. Introduction

The following classical forms for the *general design problem* (GDP) and *optimal design problem* (ODP) are assumed in this article:

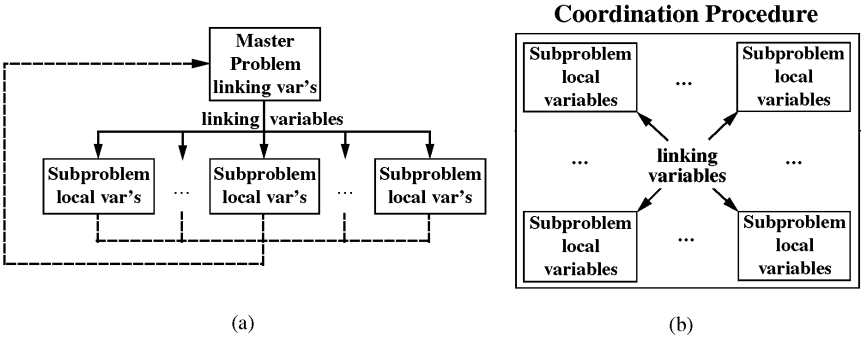

Figure 1. Generic coordination strategy for (a) hierarchically and (b) non-hierarchically partitioned problems.

General Design Problem	**Optimal Design Problem**
find $\mathbf{x} \in X \subseteq \Re^M$	find $\mathbf{x} \in X \subseteq \Re^M$
such that	such that
$\mathbf{h}(\mathbf{x}) = \mathbf{0}$	$\mathbf{h}(\mathbf{x}) = \mathbf{0}$
$\mathbf{g}(\mathbf{x}) \leq \mathbf{0}$	$\mathbf{g}(\mathbf{x}) \leq \mathbf{0}$
	and $f(\mathbf{x})$ is minimized

where f, $\mathbf{h}$, and $\mathbf{g}$ are the design objective, and vector equality and inequality design constraints, respectively, expressed as mathematical functions of the vector of design and state/behavior variables $\mathbf{x}$, and X is the set constraint. Typically, these functions can be evaluated through mathematical models based on physical principles and empirical data and expressed as computational simulations.

Optimization methods have been successfully applied to design system components using well-developed and calibrated simulations with about one hundred variables [61]. Difficulties arise when we start considering design of the overall system. The size of the problem becomes too large to expect reliable results from numerical optimization algorithms, given known model nonlinearities. Even when numerical results are obtained, one may not adequately interpret the engineering trade-offs implied. Decomposition of the optimization model into smaller submodels becomes necessary, and coordination strategies must be employed (see figure 1 and further discussion below). Problem decomposition may result in (or may allow) a conceptual simplification of the system, reduced subproblem dimensionality, parallel/distributed computation, reduced programming/debugging effort, different solution techniques for individual subproblems, modularity in parametric studies, and multicriteria analysis with single/multiple decision makers.

Three types of decomposition are commonly found in the design and optimization literature: object, aspect, and sequential decomposition. *Object decomposition* divides a system into physical components. *Aspect decomposition* divides a system according to the different specialties involved in its modeling, and it is the basis for multidisciplinary optimization (MDO). *Sequential decomposition* is applied to problems involving flow of elements or information.

Object and aspect decomposition assume a "natural" decomposition of the problem. A drawback of object decomposition is that in large, highly integrated systems drawing

"boundaries" around physical components and subassemblies is very subjective. Aspect decomposition, often defined by management considerations, may fail to account for disciplinary coupling. Sequential decomposition presumes unidirectionality of design information flows that contradicts the cooperative behavior desirable in concurrent engineering. A drawback common to the three approaches is that available computational resources may not match the naturally generated system decomposition.

This article presents a formal method for *optimal model-based decomposition* (OMBD) of design optimization problems that aims at advancing the use of nonlinear optimization techniques in the solution of large-scale design problems. Model-based decomposition allows identification of weakly connected model substructures that satisfy demands of parallel computation and availability of computational resources. Moreover, as shown in [30] for overlapping coordination, there is evidence that weakly connected submodels improve the convergence properties of coordination strategies used to solve the partitioned ODP/GDP.

Hypergraphs are used to represent design models, and the OMBD problem is formulated as a hypergraph partitioning problem. The representation and formulation are robust enough to account for computational demands of the modules in the model and the strength of their interdependencies. Hyperedge models allow mapping hypergraph into graph representations for spectral graph partitioning. A spectral graph partitioning technique is extended to included weighted vertices. The approach makes use of recent advances common to such diverse areas as graph theory, VLSI design, computational mechanics, and parallel computing. An application of the methodology to powertrain design is presented in Section 10.

The proposed OMBD method consists of the following steps:

- Represent design model with a hypergraph, and formulate the OMBD problem as a hypergraph partitioning problem (Sections 3 and 4).
- Substitute a graph for the hypergraph representation, using a hyperedge model, and reformulate the OMBD problem as a graph partitioning problem (Sections 5 and 6).
- Find the OMBD by solving the graph partitioning problem as following:
  - — Formulate graph partitioning as a 0-1 quadratic program **P1** (Eq. (5), Section 8).
  - — Replace **P1** by continuos quadratic programs **P2** and **P3** (Eqs. (6) and (7), Section 8).
  - — Obtain the graph geometric representation by solving a relaxed version of **P3** (Eqs. (8)–(10), Section 8).
  - — Generate the (global) partition from the graph geometric representation (Section 9).
  - — Refine the (global) partition using an iterative improvement algorithm (Section 7.1).

2. Related work on decomposition in design

The operations research community has extensively studied structured, partitioned problems to improve computational efficiency and robustness; however, identification of the partitioned problem model has remained a largely *ad hoc* task. Ordering heuristic algorithms have been used to improve or to identify sparsity patterns corresponding to a finite-element or finite-difference approximation over a region [17]. In engineering design, problem decomposition has received considerable attention to reduce multidisciplinary design cycle

time [6, 13, 54–56] and to streamline the design process by adequate arrangement of the tasks [19, 20, 37, 38, 50, 57, 58].

Steward [57, 58], Rogers [50], Eppinger [19, 20], and Kusiak and Wang [37, 38] applied sequential decomposition to the design sequence. Design structure and incidence matrices are used to represent precedence relationships between the tasks. An (i, j)-entry in a *design structure matrix* indicates that task j contributes information to task i. Therefore, for tasks ordered according to the structure matrix's row/column ordering, marks below the diagonal represent information transferred to later tasks; conversely, marks above the diagonal represent information fed back to earlier tasks. An (i, j)-entry in a *design incidence matrix* indicates that information j is needed to perform task i. Groups of tasks are ordered in a feed-forward sequence by detecting "circuits" among task interdependencies.

Steward used matrix transformations to minimize design iterations. Rogers used a rule-based system to generate a triangular form of the design structure matrix. Eppinger's work is based on Steward's matrix reordering; however, it includes subjective quantifiers for task dependencies. Kusiak and Wang proposed triangularization and diagonalization algorithms for the design structure and incidence matrices, respectively. They also proposed a branch and bound algorithm to identify overlapping design tasks or variables whose removal makes a design incidence matrix decomposable. The need to define the input-output relation for each task may impair the use of most of these techniques in situations where causality between tasks is non-existent or ill-defined. Heuristics or personnel interview data are used to identify "tears" of dependence relations between tasks if the problem structure is not sequentially decomposable.

Wagner and Papalambros [65, 66] used an undirected graph representation of the optimal design problem. Mathematical relations and design and state/behavior variables are depicted by the vertices and edges of the graph, respectively. Identification of "linking" or "coordinating" variables $\mathbf{y}$ leads to independent design subproblems that correspond to connected components in the graph when linking variables are deleted. The remaining variables in each subgraph are "local" variables $\mathbf{x}_k$ of the corresponding subproblem k. Heuristic acceptability criteria were used to select appropriate linking variables. A mathematical programming coordination strategy is then used to solve the original problem as a set of smaller subproblems solved independently but coordinated by a master problem.

Figure 1(a) shows a generic coordination strategy for hierarchically partitioned problems. A hierarchical decomposition usually leads to separate optimizations, supported by their own sensitivity analyses, for each subproblem and master problem. The master problem is solved for the linking variables $\mathbf{y}^*$ which are then input as parameters to the subproblems (solid arrows). Information on the dependence of the local variables with respect to the linking variables (i.e., $\mathbf{x}_k(y^*)$) is fed back to the master problem (dashed arrows). Subproblems may be recursively partitioned to generate a multilevel hierarchy.

In a non-hierarchical decomposition, figure 1(b), a subspace optimization takes place in each subproblem. Bidirectional intervention between subproblems may exist and, in general, global sensitivities and model approximations provide the means to quantify influences of one subproblem on another. We refer the reader to [64, 65] for a complete review of hierarchical and non-hierarchical coordination schemes.

Michelena and Papalambros [41] have also modeled the decomposition problem as a network optimization problem. Mathematical relations are modeled as processing units of

a communication network and design and state/behavior variables as the communication links between these units. The optimal decomposition problem is then formulated as one of finding the communication links whose failure lessens the most the network reliability, which is a measure of network connectivity.

3. Hypergraph representation of a design problem

Function dependence on variables may be represented by a Boolean matrix termed the *functional dependence table* (FDT), rows labeled with design relation/function names and columns labeled with design and state/behavior variable names. The entry in the *i*th row and *j*th column is "True" if the *i*th function depends on the *j*th variable; otherwise, it is "False". Consider the following optimization problem, a modification of No. 55 from Hock and Schittkowski [33]:

$$
\begin{array}{lll}
\min\limits_{\mathbf{x}\in[0,1]^6} & f = f_1 + f_2 & \\
\text{subject to} & f_1 = x_1 + \exp(x_1 x_4) & h_3 = x_4 + x_6 - 2 = 0 \\
& f_2 = 2x_2 + 4x_5 & h_4 = x_1 + x_4 - 1 = 0 \\
& h_1 = x_1 + 2x_2 + 5x_5 - 6 = 0 & h_5 = x_2 + x_5 - 2 = 0 \\
& h_2 = x_1 + x_2 + x_3 - 3 = 0 & h_6 = x_3 + x_6 - 2 = 0
\end{array}
\tag{1}
$$

Figure 2(a) shows the corresponding FDT. A shaded box indicates a "True" Boolean value. Figure 2(b) shows the FDT for the same problem after x_1 and x_3 have been selected as linking variables and rows and columns have been reordered to reveal two partitions of the problem: subproblem 1 with functions $\{f_1, h_3, h_4, h_6\}$ and local variables $\{x_4, x_6\}$,

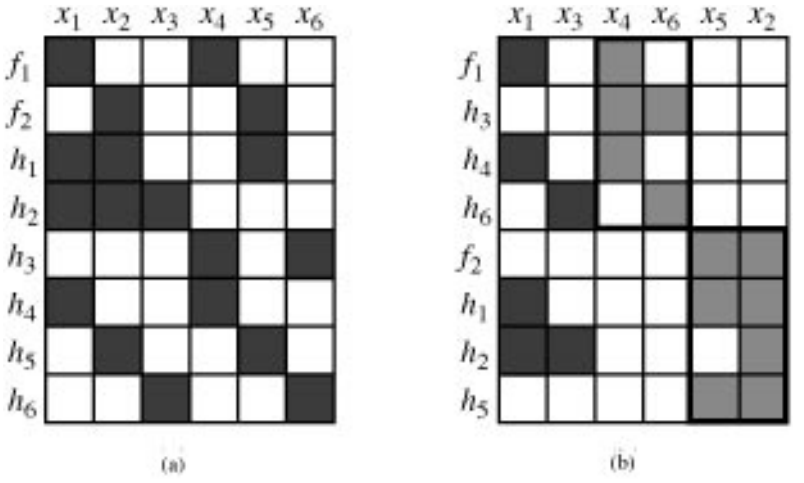

Figure 2. Functional dependence tables for example problem. (a) Original form and (b) after reordering rows and columns to identify two subproblems.

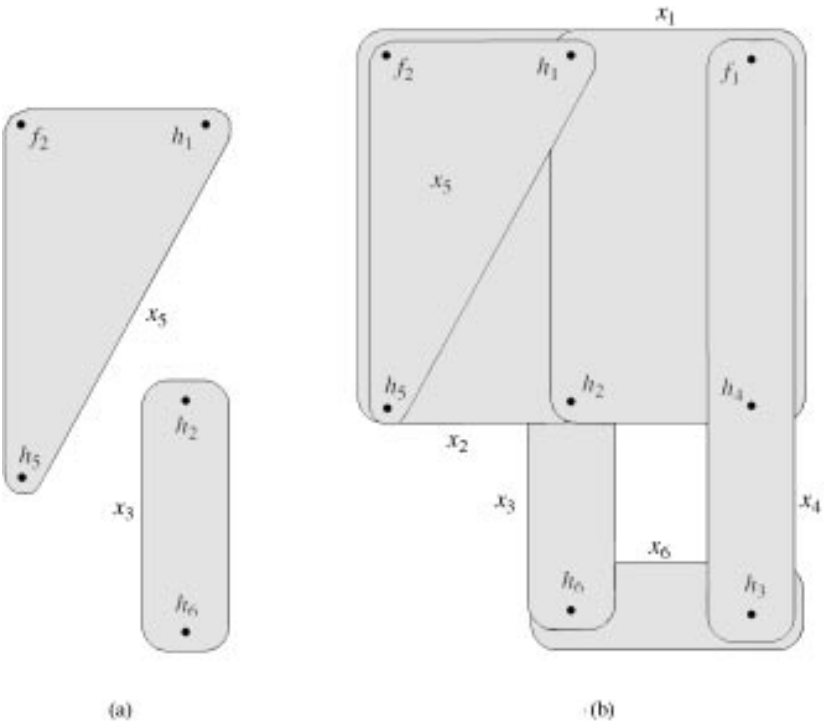

Figure 3. (a) Hyperedge representation of function dependence on variables x_3 and x_5, and (b) hypergraph representation of example problem.

and subproblem 2 with functions $\{f_2, h_1, h_2, h_5\}$ and local variables $\{x_2, x_5\}$. In the case of hierarchical coordination, the linking variables x_1 and x_3 are held constant and the subproblems are solved independently with respect to the local variables. The solution to each subproblem is traced out as a function of x_1 and x_3, and a two-variable master problem is solved to update their values, which are then fed back to the subproblems.

A design problem may also be represented by a *hypergraph* $H = (V, E_H)$ in which hyperedges in E_H are subsets of V. Vertices in V represent design functions (i.e., objective and constraints) or simulation modules, and hyperedges represent design and state/behavior variables. A hyperedge $e_i \in E_H$ represents a variable x_i if and only if for every vertex $v_i \in e_i$, the function associated with v_i depends on x_i. The edge-valency or cardinality of a hyperedge is equal to the number of vertices contained in the hyperedge (which is also equal to the number of nonzero entries in the corresponding column of the FDT). A hyperedge of cardinality p will be termed a p-hyperedge. The vertex-valency or degree of a vertex is equal to the number of hyperedges containing the vertex (which is also equal to the number of nonzero entries in the corresponding row of the FDT). Figure 3(a) shows the 2- and 3-hyperedges associated with variables x_3 and x_5 in the FDT of figure 2, respectively. Figure 3(b) shows the hypergraph representation of the problem of Eq. (1). For our purposes of problem decomposition, 1-hyperedges will be excluded from the representation since they do not contribute to the connectivity of the problem model.

4. Optimal model-based decomposition as a hypergraph partitioning problem

Optimal decomposition of a design problem calls for (i) minimizing the interconnection between subproblems and (ii) balancing the size of the subproblems. The former is aimed at reducing the size of the master problem and the effort to coordinate individual subproblems, and the latter is aimed at matching available computational resources. Hence, the OMBD problem is formulated as the following hypergraph partitioning problem in which vertices represent design functions or simulation modules, and hyperedges depict design and state/behavior variables.

Hypergraph K-partitioning problem. Given a hypergraph $H = (V, E_H)$ containing N vertices $V = \{v_1, v_2, \ldots, v_N\}$ with positive weights $\omega_v(v_i)$, and M hyperedges $E_H = \{e_1, e_2, \ldots, e_M\}$ with positive weights $\omega_e(e_j)$, a constant $2 \leq K \leq N$, and a partition load (or size) vector $\mathbf{m} = (m_1, \ldots, m_K)$ such that $m_k \geq m_{k+1}$ and $\sum_{k=1}^{K} m_k = \sum_{i=1}^{N} \omega_v(v_i)$, find a partition of V into K disjoint subsets $P^K = \{V_1, V_2, \ldots, V_K\}$ that minimizes (i) the total weight of the hyperedges cut by P^K, $C(P^K)$, and (ii) $|\sum_{v_i \in V_k} \omega_v(v_i) - m_k|$ for every k in $\{1, 2, \ldots, K\}$. The hyperedges cut by P^K are $E_H^c(P^K) = \{e_j \in E_H$: there exist v_{i_1}, v_{i_2} in e_j, $v_{i_1} \in V_{j_1} \in P^K$, $v_{i_2} \in V_{j_2} \in P^K$, and $j_1 \neq j_2\}$. Thus, the total weight of the hyperedges cut by P^K is $C(P^K) = \sum_{e_j \in E_H^c(P^K)} \omega_e(e_j)$.

When this formulation is applied to the OMBD problem, *vertex weights* represent time to evaluate a function or execute a simulation, *edge weights* depict strength of function-variable dependence or amount of transferred data between simulation modules, and *partition loads* represent processing capabilities in a distributed computational environment. In this article, the terms hypergraph and graph K-partitioning imply some sort of constraint on the partition sizes (loads), as oppose to the unrestricted common meaning of the terms.

5. Hyperedge model

Some partitioning methods found in the literature are applicable only to graphs. (A [linear] graph is a hypergraph in which the cardinality of every hyperedge is equal to two.) Specifically, spectrum-based methods have only been developed for graphs, their extension to hypergraphs being a major challenge. Thus, partitioning of a design problem and, thereby, its hypergraph representation may require approximating the hypergraph by a graph.

Hypergraphs are also used to model circuit netlists in VLSI design, where the vertices of the hypergraph represent modules or cells, and the hyperedges represent signal nets. A method used by VLSI designers to approximate a hypergraph H by a graph G consists in defining the vertex set of G the same as the vertex set of H. The edge set of G is obtained by replacing each hyperedge of H by the edge set of a clique containing the vertices of the hyperedge. That is, a clique of C_2^p weighted edges, which will be called a p-clique, replaces every p-hyperedge, as shown in figure 4 for $p = 4$. Weighted edges are needed to estimate the number of hyperedges of H cut by a vertex set partition from the weights of the edges of G cut by the same partition. Resulting parallel edges in G are replaced by a single edge whose weight is determined by adding the weights of the parallel edges.

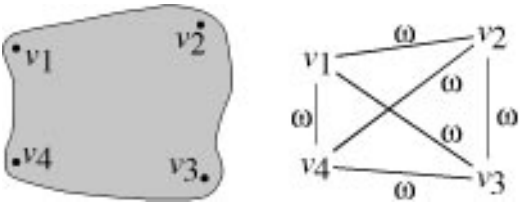

Figure 4. Hyperedge of cardinality $p = 4$ and its clique model.

Four hyperedge models can be found in the circuit and graph partitioning literature, each assigning different edge weights to the associated cliques.

(a) The "standard" clique model [39] replaces every p-hyperedge by a clique with edge weights $\omega = \frac{1}{(p-1)}$. Since the minimum number of edges needed to be cut to partition the vertex set of a p-hyperedge is $(p-1)$, the total weight of cut edges will be at least one. Thus, the standard hyperedge model overestimates the number of hyperedges cut by a partition. For example, in figure 4 with $\omega = \frac{1}{3}$, the estimated number of hyperedges cut by the partition $P^2 = \{\{v_1, v_4\}, \{v_2, v_3\}\}$ is $\frac{4}{3}$ instead of 1.

(b) Hadley et al. [28] proposed a hyperedge model in which the total weight of the edges of graph G that are cut by any vertex partition is not greater than the number of hyperedges of the associated hypergraph H cut by the same partition. (So this hyperedge model underestimates the number of hyperedges cut by a partition.) Each p-clique edge is assigned the weight $\omega = \frac{1}{\phi(p,k)}$, where $\phi(p, k)$ is the maximum number of edges cut by a k-partition of a p-clique. $\phi(p, k)$ is given by

$$\phi(p,k) = \binom{p}{2} - \binom{\lfloor \frac{p}{k} \rfloor}{2}\left(k\left(1 + \left\lfloor \frac{p}{k} \right\rfloor\right) - p\right) - \binom{\lfloor \frac{p}{k} \rfloor}{2}\left(p - k\left\lfloor \frac{p}{k} \right\rfloor\right) \qquad (2)$$

When p is exactly divisible by k, $\omega = \frac{2k}{(k-1)p^2}$. For the 4-hyperedge of figure 4, $\omega = \frac{1}{4}$ when $k = 2$, and $\omega = \frac{1}{6}$ when $k = 4$.

(c) Alpert and Kahng [1] proposed a "middle of the road" approach whereby any cut hyperedge will make an expected contribution of one to the weighted cut cost function. By enumerating all possible bipartitions of a p-hyperedge and assigning a uniform distribution to each bipartition, they showed that each edge of the p-clique should receive a weight $\omega = \frac{4}{p(p-1)} \cdot \frac{2p-2}{2^p}$. For the hyperedge of figure 4, $\omega = \frac{7}{24}$.

(d) Bolla [7] modeled the hypergraph partitioning problem by constructing a K-dimensional geometric representative of the hypergraph, whereby vertices and hyperedges are mapped onto the K-dimensional Euclidean space. The geometric representative of a hyperedge is then replaced by the center of gravity of the geometric representatives of the vertices contained in the hyperedge. The mathematical formulation results in edge weights $\omega = \frac{1}{p}$ for a p-clique substituting for a p-hyperedge. For the hyperedge of figure 4, $\omega = \frac{1}{4}$.

In this article, we propose a hyperedge model that generalizes Alpert and Kahng model (see (c) above) to consider up to a K-partition of a hyperedge, instead of only a bipartition. The proposed clique edge weights are given in the following Lemma.

Lemma 1 (Proposed hyperedge model). *Assume that the set of p vertices of a p-hyperedge can be partitioned in up to k subsets such that each one of the* $(k^p - k)$ *possible assignments of vertices to the subsets that generates a cut hyperedge is equally likely. Then, a cut p-hyperedge will make an expected contribution of one to the weighted cut cost function if and only if the edges of the associated p-clique have weights* $\omega = \frac{2k}{p(p-1)(k-1)} \cdot \frac{k^p - k}{k^p}$.

Proof: Consider the partition of a p-hyperedge vertex set into k subsets of sizes $m_1, m_2, m_3, \ldots, m_k$. The expected size of the cut set ϕ is

$$\begin{aligned}
\phi &= \frac{1}{k^p - k} \sum_{\substack{m_i \in \mathbb{N} \\ \sum_{i=1}^{k} m_i = p}} \left(\sum_{i<j} m_i m_j \right) \left(\frac{p!}{m_1!\, m_2!\, m_3! \cdots m_k!} \right) \\
&= \frac{p(p-1)}{k^p - k} \sum_{\substack{m_i \in \mathbb{N} \\ \sum_{i=1}^{k} m_i = p}} \left(\sum_{i<j} \frac{(p-2)!}{m_1! \cdots (m_i - 1)! \cdots (m_j - 1)! \cdots m_k!} \right) \\
&= \frac{p(p-1)}{k^p - k} \sum_{i<j} k^{p-2} = \frac{p(p-1)(k-1)}{2k} \frac{k^p}{k^p - k}
\end{aligned}$$

Clique edge weights ω are defined as $\frac{1}{\phi}$. □

A comparison of the five hyperedge models is shown in figure 5 for $k \geq 5$. As mentioned above, model (a) overestimates the size of cut set, whereas model (b) underestimates it. The proposed model and model (c) give intermediate values for the size of the cut set; however, model (c) shall overestimate the size of the cut set every time that the vertices in a hyperedge are divided among more than two vertex partitions.

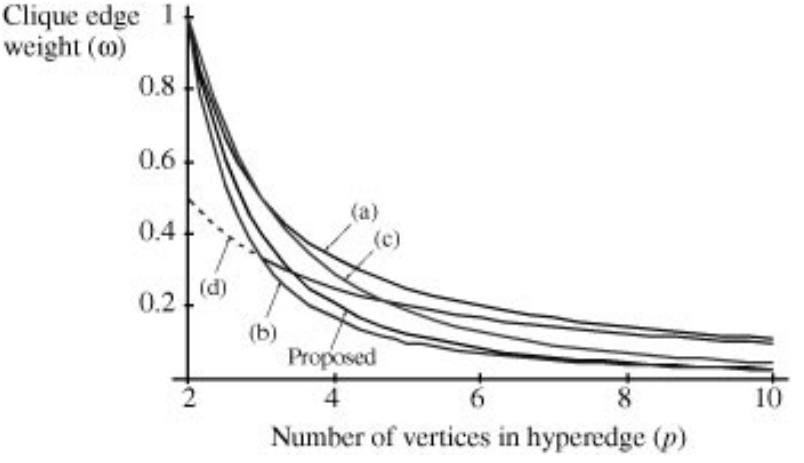

Figure 5. Comparison of four existing and proposed hyperedge models.

6. Optimal model-based decomposition as a graph partitioning problem

Given a hypergraph representation of a design problem, a graph representation may be constructed using one of the hyperedge models explained above. The OMBD problem can then be formulated as the following graph partitioning problem with vertices V representing design functions and (cliques of) edges E_G representing design and state/behavior variables.

Graph K-partitioning problem. Given a graph $G = (V, E_G)$ containing N vertices $V = \{v_1, v_2, \ldots, v_N\}$ with positive weights $\omega_v(v_i)$, and M edges $E_G = \{e_1, e_2, \ldots, e_M\}$ with positive weights $\omega_e(e_j)$, a constant $2 \leq K \leq N$, and a partition load (or size) vector $\mathbf{m} = (m_1, \ldots, m_K)$ such that $m_k \geq m_{k+1}$ and $\sum_{k=1}^{K} m_k = \sum_{i=1}^{N} \omega_v(v_i)$, find a partition of V into K disjoint subsets $P^K = \{V_1, V_2, \ldots, V_K\}$ that minimizes (1) the total weight of the edges cut by P^K, $C(P^K)$, and (2) $|\sum_{v_i \in V_k} \omega_v(v_i) - m_k|$ for every k in $\{1, 2, \ldots, K\}$. The edges cut by P^K are $E_G^c(P^K) = \{\{v_{i_1}, v_{i_2}\} \in E_G : v_{i_1} \in V_{j_1} \in P^K, v_{i_2} \in V_{j_2} \in P^K$, and $j_1 \neq j_2\}$. Thus, the total weight of the edges cut by P^K is $C(P^K) = \sum_{e_j \in E_G^c(P^K)} \omega_e(e_j)$.

In the above graph partitioning formulation, an edge weight is computed by multiplying the weight defined by the hyperedge model and the weight of the associated hyperedge (which depicts strength of function-variable dependence or amount of transferred data between simulation modules). As mentioned above, minimizing the total weight of cut hyperedges is equivalent to minimizing the total weight assigned to linking variables. However, minimizing the weight of cut edges in a graph representation is equivalent to minimizing not only the weight of linking variables but also the number of functions on which these variables depend.

7. Review of graph and hypergraph partitioning techniques

Hypergraph and graph K-partitioning problems have been studied for applications where a large network or system must be partitioned into subsystems such that elements in the same subsystem are strongly interconnected, whereas elements in different subsystems are weakly interconnected. Such applications include computer logic and page partitioning [15, 23], VLSI layout and packaging of circuits [1, 18, 49, 67], machine layout in manufacturing systems [62, 63], assignment of computations to multiple processors [31], and domain decomposition of finite- element or finite-volume grids for parallel computation [3, 22, 53].

Both hypergraph and graph K-partitioning problems are NP-hard, even if edge and vertex weights are one, and the number of partitions is two. If the number of partitions K is fixed and there is *no* restriction on the size of the partitions, then the problem is solvable in polynomial time $O(N^{K^2})$, where N is the number of vertices in the graph [27]. However, this case has no practical application since the resulting partitions could be very unbalanced.

Partitioning methods include iterative improvement and global techniques. Iterative improvement algorithms, also know as local search algorithms, make local changes to an initial partition to minimize the total weight of the edges cut while keeping the parts balanced. These algorithms are quite robust because they can deal with graphs and hypergraphs, and arbitrary vertex and edge weights and balance criteria. For global methods, the partitioning problem is formulated as an optimization problem and solved using approximation techniques.

7.1. Iterative improvement partitioning methods

Most iterative improvement algorithms are based on a heuristic procedure devised by Kernighan and Lin for graph bisection [35]. Variants to this algorithm have extended it to K-partition of hypergraphs containing weighted vertices and edges with improved running times.

Kernighan-Lin algorithm. Kernighan and Lin [35] first proposed a local search method to solve the graph bisection problem. Their algorithm (KL) starts with an initial (balanced) partition and exchanges pairs of vertices across the cut of the bisection. To reduce the risk of being trapped in a local minimum, the KL procedure determines the vertex pair whose exchange results in the largest decrease of the cutsize or in the smallest increase, if no decrease is possible. The original KL algorithm can handle edge weights.

Dunlop and Kernighan [18] extended the KL algorithm to hypergraph bisection. Overall time complexity of this partitioning algorithm is $O(N^2 \log N)$ (per pass), where N is the number of vertices in the hypergraph.

Fiduccia-Mattheyses algorithm. Fiduccia and Mattheyses [24] proposed a bipartition algorithm based on the KL algorithm that could handle vertex and edge weights, unequal partitions, and hypergraphs. In the Fiduccia-Mattheyses (FM) algorithm a single vertex is moved across the cut in a single move. Thus, the algorithm can deal with partitions of different sizes and nonuniform vertex weights. Partition balance is enforced by a bound on the partition sizes. Bucket sort is used for selecting the vertices to be moved between partition and results in the principal characteristic of the FM algorithm: an average running time of $O(P)$ (per pass), where P is the total number of terminals in the hypergraph. The total number of terminals in a hypergraph $H = (V, E_H)$ is $P = \sum_{e \in E_H} |e|$.

Other iterative improvement algorithms. Variants to the KL and FM algorithms continue to appear in the graph and hypergraph partitioning literature. Krishnamurthy [36] introduced the concept of *level gains* into the FM heuristics, using gains in later moves to distinguish between equal gain vertices. Krishnamurthy's algorithms runs in time $O(LP)$ (per pass), where L is the number of levels used. Suaris and Kedem [59] extended the FM algorithm to hypergraph quadrisection. Barnes et al. [5] determined sets of vertices to interchange between partitions from solving a transportation (linear) problem. Sanchis [51, 52] adapted Krishnamurthy's bipartition algorithm to the hypergraph K-partitioning problem. The time complexity of Sanchis' algorithm is $O(LPK(\log K + LD_{\max}))$ (per pass), where $D_{\max}$ is the maximum vertex degree in the hypergraph. Hendrickson and Leland [32] have implemented FM-HL, an iterative improvement graph partitioning algorithm patterned after the FM algorithm but generalized in several ways: First, K-partitioning is possible. Second, the algorithms can handle an arbitrary interset cost metric. Third, robustness is improved by including an element of randomness. Fourth, the algorithm implementation reuses computations in a manner that reduces the overall running time to $O((K-1)M)$, where M is the number of edges.

7.2. Global partitioning methods

Iterative improvement algorithms as those described above are quite good at finding locally optimal answers, but unless they are initialized with a "good" partition, the local optimum may be far from the global. Global partitioning methods start with an encoding of a problem instance, such as a graph adjacency matrix or list, or a hypergraph incidence matrix, and compute an approximation to the optimal partition that can be used as input to an iterative improvement algorithm.

Network-reliability-based method. Michelena and Papalambros [41] have modeled the hypergraph partitioning problem as a network optimization problem. Hypergraph vertices are modeled as the processing units of a communication network. Hyperedges are communication links between these units. The optimal partitioning problem is then formulated as one of finding the communication links that have the most effect on the overall network reliability, i.e., links whose failure lessens the most the network reliability—which is taken as a measure of the hypergraph connectivity. A pair-connected reliability measure was chosen to generate balanced partitions.

Spectral methods. Spectral partitioning methods identify a good approximation to a graph K-partition from global information about the structure of the graph extracted from a matrix spectrum. Specifically, a K-dimensional geometric representation of the graph is constructed from the K eigenvectors that correspond to the smallest eigenvalues of the graph Laplacian matrix. A drawback of these methods is that they cannot be directly applied to hypergraphs, so a hyperedge model is needed to approximate the hypergraph by a graph. The relation between the spectrum of a graph and other graph properties has been an area of active research [4, 8, 16, 25, 26, 43], but only recently spectrum-based methods have been successfully applied to graph partitioning [1, 2, 21, 31, 46, 47, 53]. We present below a spectral graph K-partitioning formulation that extends Rend and Wolkowicz's to graphs containing weighted vertices.

Other global methods. Simulated annealing (SA) has been used for graph partitioning by Johnson et al. [34] and Bui et al. [10] with mixed results. They showed that SA usually needs much more time than iterative improvement methods, specifically when used on graphs generated with a built-in geometric structure. Bui and Moon [11] presented a hybrid genetic algorithm that combines a variation of the Fiduccia-Mattheyses algorithm with genetic space exploration to give competitive performance. Bui et al. [9] applied network-flow techniques to graph bisection with good results for small-degree graphs.

8. Spectral graph partitioning formulation

As described in Section 6, the K-partitioning problem entails finding a partition of V into K disjoint subsets $P^K = \{V_1, V_2, \ldots, V_K\}$ such that the vector of weighted sizes of the partitions is close to $\mathbf{m}$ componentwise and the total weight $C(P^K)$ of edges cut by the partition is minimized.

The following spectral formulation of the graph partitioning problem is similar to that given by Rendl and Wolkowicz [47] and Falkner et al. [21]; however, it also accounts for weighted vertices. The $N \times N$ adjacency matrix $\mathbf{A}$ of a graph G is defined as $\mathbf{A} = (a_{ij})$, where $a_{ij} = \omega_e(\{v_i, v_j\})$ if $\{v_i, v_j\} \in E_G$, and $a_{ij} = 0$ if no such edge exist. Let $deg(i) = \sum_{j=1}^{N} a_{ij}$ be the degree of v_i. The $N \times N$ diagonal degree matrix $\mathbf{D}$ is given by $d_{ii} = deg(i)$ and $d_{ij} = 0$ if $i \neq j$. The $N \times N$ Laplacian matrix of G is defined as $\mathbf{L} = \mathbf{D} - \mathbf{A}$. Properties of a graph Laplacian matrix include:

- $\mathbf{L}$ is positive semidefinite.
- $\mathbf{L}$'s rows and columns add up to zero.
- $\mathbf{L}$'s smallest eigenvalue $\lambda_1 = 0$ and has a corresponding eigenvector $(1/\sqrt{N}, \ldots, 1\sqrt{N})^t$.
- If the graph is connected, then $\mathbf{L}$'s second smallest eigenvalue $\lambda_2 > 0$. The multiplicity of zero as an eigenvalue of L equals the number of connected components of the graph.

Let $\mathbf{X} \in \Re^{N \times K}$ be the assignment matrix for P^K, i.e., $x_{ik} = 1$ if vertex v_i is assigned to partition V_k, and $x_{ik} = 0$ otherwise. The kth column of $\mathbf{X}$ is denoted by $\mathbf{x}_k$. The weighted edge cut is

$$
\begin{aligned}
C(P^K) &= \frac{1}{2}\sum_{k=1}^{K}\sum_{v_i \in V_k}\sum_{v_j \notin V_k} a_{ij} = \frac{1}{2}\sum_{k=1}^{K}\sum_{v_i \in V_k}\left[\sum_{j=1}^{N} a_{ij} - \sum_{v_j \in V_k} a_{ij}\right] \\
&= \frac{1}{2}\sum_{k=1}^{K}\sum_{i=1}^{N} x_{ik}\left[deg(i) - \sum_{j=1}^{N} x_{jk} a_{ij}\right] \\
&= \frac{1}{2}\sum_{k=1}^{K}\sum_{i=1}^{N} x_{ik}x_{ik}deg(i) - \frac{1}{2}\sum_{k=1}^{K}\sum_{i=1}^{N}\sum_{j=1}^{N} x_{ik}x_{jk}a_{ij} \\
&= \frac{1}{2}\sum_{k=1}^{K}\sum_{i=1}^{N}\sum_{j=1}^{N} x_{ik}x_{jk}\delta_{ij}deg(i) - \frac{1}{2}\sum_{k=1}^{K}\sum_{i=1}^{N}\sum_{j=1}^{N} x_{ik}x_{jk}a_{ij} \\
&= \frac{1}{2}\sum_{k=1}^{K}\sum_{i=1}^{N}\sum_{j=1}^{N} x_{ik}x_{jk}[\delta_{ij}deg(i) - a_{ij}] \\
&= \frac{1}{2}\sum_{k=1}^{K} \mathbf{x}_k^t \mathbf{L}\mathbf{x}_k = \frac{1}{2}\mathrm{tr}(\mathbf{X}^t\mathbf{L}\mathbf{X})
\end{aligned}
\tag{3}
$$

where $\delta_{ij} = 1$ if $i = j$, and $\delta_{ij} = 0$ otherwise. Moreover, the elements of $\mathbf{X}$ have to satisfy the following constraints to ensure both a balanced partitioning and assignment of each vertex to a single partition:

$$
\begin{aligned}
\sum_{i=1}^{N} x_{ik}\omega_v(v_i) &= m_k, \quad k = 1, \ldots, K \\
\sum_{k=1}^{K} x_{ik} &= 1, \quad i = 1, \ldots, N
\end{aligned}
\tag{4}
$$

The graph K-partitioning problem then becomes

$$(\mathbf{P1}) \quad \begin{cases} \text{Minimize } \frac{1}{2}\text{tr}(\mathbf{X}^t\mathbf{L}\mathbf{X}) \text{ such that} \\ \mathbf{X}^t\mathbf{w} = \mathbf{m} \\ \mathbf{X}\mathbf{u}_K = \mathbf{u}_N \\ \mathbf{X} \text{ is a 0-1 } N \times K \text{ matrix} \end{cases} \tag{5}$$

where $\mathbf{w} = (\omega_v(v_1), \omega_v(v_2), \ldots, \omega_v(v_N))^t$ and $\mathbf{u}_l = (1, \ldots, 1)^t \in \Re^l$. Note that any two column vectors of $\mathbf{X}$ are orthogonal.

Lemma 2. *The feasible set of problem* (**P1**) $\mathfrak{J} = \{\mathbf{X} \in \Re^{N\times K} : x_{ij}$ *is* 0 *or* 1, $\mathbf{X}^t\mathbf{w} = \mathbf{m}$, *and* $\mathbf{X}\mathbf{u}_K = \mathbf{u}_N\}$ *is equal to* $\mathfrak{J}' = \{\mathbf{X} \in \Re^{N\times K} : x_{ij} \geq 0,\ \mathbf{X}^t\mathbf{w} = \mathbf{m}, \mathbf{X}\mathbf{u}_K = \mathbf{u}_N$, *and* $\mathbf{X}^t\mathbf{W}\mathbf{X} = \mathbf{M}\}$, *where* $\mathbf{W}$ *is the* $N \times N$ *diagonal matrix diag*($\mathbf{w}$), *and* $\mathbf{M}$ *is the* $K \times K$ *diagonal matrix diag*($\mathbf{m}$). (*By problem assumption*, *tr*($\mathbf{W}$) = *tr*($\mathbf{M}$).)

Proof: That $\mathfrak{J} \subseteq \mathfrak{J}'$ is clear. Let $\mathbf{X} \in \mathfrak{J}'$. Since $x_{ij} \geq 0$ and $\mathbf{X}\mathbf{u}_K = \mathbf{u}_N$, then $0 \leq x_{ij} \leq 1$ and $x_{ij}^2 \leq x_{ij}$. $\text{tr}(\mathbf{X}^t\mathbf{W}\mathbf{X}) = \sum_{i,j} \omega_v(v_i)x_{ij}^2 = \text{tr}(\mathbf{M})$, and $\mathbf{u}_N^t\mathbf{W}\mathbf{X}\mathbf{u}_K = \sum_{i,j} \omega_v(v_i)x_{ij} = \mathbf{u}_N^t\mathbf{W}\mathbf{u}_N = \text{tr}(\mathbf{W}) = \text{tr}(\mathbf{M})$. Hence, $\sum_{i,j} \omega_v(v_i)x_{ij}^2 = \sum_{i,j} \omega_v(v_i)x_{ij}$, so x_{ij} is 0 or 1, and $\mathbf{X} \in \mathfrak{J}$. □

Therefore, an equivalent formulation of (**P1**) is

$$(\mathbf{P2}) \quad \begin{cases} \text{Minimize } \frac{1}{2}\text{tr}(\mathbf{X}^t\mathbf{L}\mathbf{X}) \text{ such that} \\ \mathbf{X}^t\mathbf{w} = \mathbf{m} \\ \mathbf{X}\mathbf{u}_K = \mathbf{u}_N \\ \mathbf{X}^t\mathbf{W}\mathbf{X} = \mathbf{M} \\ \mathbf{X} \text{ is a nonnegative element-wise } N \times K \text{ matrix} \end{cases} \tag{6}$$

Let $\mathbf{X} = \frac{\mathbf{u}_N\mathbf{u}_K^t\mathbf{M}}{\text{tr}(\mathbf{M})} + \mathbf{W}^{-1/2}\mathbf{S}_N\mathbf{Y}\mathbf{R}_K^t\mathbf{M}^{1/2}$, where $\mathbf{Y}$ is a $(N-1) \times (K-1)$ matrix, $\mathbf{S}_N$ is a orthonormal $N \times (N-1)$ matrix spanning $\{\mathbf{W}^{1/2}\mathbf{u}_N\}^\perp$, so $\mathbf{S}_N^t\mathbf{W}^{1/2}\mathbf{u}_N = 0$ and $\mathbf{S}_N^t\mathbf{S}_N = \mathbf{I}_{N-1}$, and $\mathbf{R}_K$ is a orthonormal $K \times (K-1)$ matrix spanning $\{\mathbf{M}^{1/2}\mathbf{u}_K\}^\perp$, so $\mathbf{R}_K^t\mathbf{M}^{1/2}\mathbf{u}_K = 0$ and $\mathbf{R}_K^t\mathbf{R}_K = \mathbf{I}_{K-1}$. As suggested in [47], $\mathbf{S}_N$ and $\mathbf{R}_K$ may be obtained by applying the Gram-Schmidt orthogonalization process to matrices $[\mathbf{W}^{1/2}\mathbf{u}_N : (\mathbf{0} : \mathbf{I}_{N-1})t]$ and $[(\mathbf{M}^{1/2}\mathbf{u}_K) : (\mathbf{0} : \mathbf{I}_{K-1})^t]$, respectively. Any matrix $\mathbf{X}$ defined as above satisfies the first two constraints in (**P2**). The third constraint and nonnegativity condition in (**P2**) are reduced to $\mathbf{Y}^t\mathbf{Y} = \mathbf{I}_{K-1}$ and $\mathbf{W}^{-1/2}\mathbf{S}_N\mathbf{Y}\mathbf{R}_K^t\mathbf{M}^{1/2} \geq_{\text{ew}} -\frac{\mathbf{u}_N\mathbf{u}_K^t\mathbf{M}}{\text{tr}(\mathbf{M})}$, respectively. The objective function is reduced to $\frac{1}{2}\text{tr}(\mathbf{R}_K^t\mathbf{M}\mathbf{R}_K\mathbf{Y}^t\mathbf{S}_N^t\mathbf{W}^{-1/2}\mathbf{L}\mathbf{W}^{-1/2}\mathbf{S}_N\mathbf{Y})$. Therefore, (**P1**) and (**P2**) are equivalent to

$$(\mathbf{P3}) \quad \begin{cases} \text{Minimize } \frac{1}{2}\text{tr}(\mathbf{R}_K^t\mathbf{M}\mathbf{R}_K\mathbf{Y}^t\mathbf{S}_N^t\mathbf{W}^{-1/2}\mathbf{L}\mathbf{W}^{-1/2}\mathbf{S}_N\mathbf{Y}) \text{ such that} \\ \mathbf{Y}^t\mathbf{Y} = \mathbf{I}_{K-1} \\ \mathbf{W}^{-1/2}\mathbf{S}_N\mathbf{Y}\mathbf{R}_K^t\mathbf{M}^{1/2} \geq_{\text{ew}} -\frac{\mathbf{u}_N\mathbf{u}_K^t\mathbf{M}}{\text{tr}(\mathbf{M})} \end{cases} \tag{7}$$

An approximate solution to (**P3**) and a lower bound for $C(P^K)$ may be obtained by relaxing the constraints to $\mathbf{Y}^t\mathbf{Y} = \mathbf{I}_{K-1}$ and using the following *Representation Theorem* to compute $\mathbf{Y}$, with $\mathbf{A} = \mathbf{R}_K^t\mathbf{M}\mathbf{R}_K$ and $\mathbf{B} = \mathbf{S}_N^t\mathbf{W}^{-1/2}\mathbf{L}\mathbf{W}^{-1/2}\mathbf{S}_N$. This approximate solution provides a graph geometric representation $V \to \Re^K$ given by the rows of $\mathbf{X}$.

Representation Theorem

Let $\mathbf{A}$ and $\mathbf{B}$ be symmetric matrices. The $P \times Q$ ($P \geq Q$) matrix that minimizes $\mathrm{tr}(\mathbf{A}\mathbf{Y}^t\,\mathbf{B}\mathbf{Y})$ subject to the constraint $\mathbf{Y}^t\mathbf{Y} = \mathbf{I}_Q$ is the matrix $\mathbf{Y} = \mathbf{P}\mathbf{Q}^t$, where $\mathbf{Q}$ is the $Q \times Q$ orthogonal matrix whose columns contain the eigenvectors of $\mathbf{A}$, and $\mathbf{P}$ is the $P \times Q$ orthonormal matrix whose columns contain the eigenvectors of $\mathbf{B}$ corresponding to the Q smallest eigenvalues of $\mathbf{B}$. The order of the columns of $\mathbf{Q}$ and $\mathbf{P}$ is such that the corresponding eigenvalues of $\mathbf{A}$ and $\mathbf{B}$ are in nonincreasing and nondecreasing order, respectively. Also, the minimum of $\mathrm{tr}(\mathbf{A}\mathbf{Y}^t\,\mathbf{B}\mathbf{Y})$ such that $\mathbf{Y}^t\mathbf{Y} = \mathbf{I}_Q$ is $\sum_{i=1}^{Q} \lambda_i(\mathbf{B})\lambda_{Q-i+1}(\mathbf{A})$, where $\lambda_i(\cdot) \leq \lambda_{i+1}(\cdot)$. (This theorem is an extension of Theorem 3.1 by Rendl and Wolkowicz [48].)

Corollary 1. *For the K-partitioning of a graph with Laplacian matrix* $\mathbf{L}$, *vertex weight vector* $\mathbf{w}$ ($\mathbf{W} = diag(\mathbf{w})$), *and partition load vector* $\mathbf{m}$ ($\mathbf{M} = diag(\mathbf{m})$),

$$C(P^K) \geq \frac{1}{2}\sum_{k=1}^{K-1} \lambda_k\left(\mathbf{S}_N^t\mathbf{W}^{-1/2}\mathbf{L}\mathbf{W}^{-1/2}\mathbf{S}_N\right)\lambda_{K-k}\left(\mathbf{R}_K^t\mathbf{M}\mathbf{R}_K\right) \tag{8}$$

where $\mathbf{R}_K$ *is the orthonormal* $K \times (K-1)$ *matrix spanning* $\{\mathbf{M}^{1/2}\mathbf{u}_K\}^{\perp}$, *and* S_N *is the orthonormal* $N \times (N-1)$ *matrix spanning* $\{\mathbf{W}^{1/2}\mathbf{u}_N\}^{\perp}$. *The lower bound is attained for* $\mathbf{X} = \frac{\mathbf{u}_N\mathbf{u}_K^t\mathbf{M}}{\mathrm{tr}(\mathbf{M})} + \mathbf{W}^{-1/2}\mathbf{S}_N\mathbf{P}\mathbf{Q}^t\mathbf{R}_K^t\mathbf{M}^{1/2}$ (*that defines a K-dimensional representation of the graph*). $\mathbf{Q}$ *is the* $(K-1) \times (K-1)$ *orthogonal matrix whose columns contain the eigenvectors of* $\mathbf{R}_K^t\mathbf{M}\mathbf{R}_K$, *and* $\mathbf{P}$ *is the* $(N-1) \times (K-1)$ *orthonormal matrix whose columns contain the eigenvectors of* $\mathbf{S}_N^t\mathbf{W}^{-1/2}\mathbf{L}\mathbf{W}^{-1/2}\mathbf{S}_N$ *corresponding to the* $(K-1)$ *smallest eigenvalues of* $\mathbf{S}_N^t\mathbf{W}^{-1/2}\mathbf{L}\mathbf{W}^{-1/2}\mathbf{S}_N$. *The order of the columns of* $\mathbf{P}$ *and* $\mathbf{Q}$ *is defined as for the Representation Theorem above.*

The following Corollary 2 applies to equal load partitions, i.e., $\mathbf{m} = \frac{\mathrm{tr}(diag(\mathbf{w}))}{K}\mathbf{u}_K$—the case when processing resources assigned to each subproblem are planned to be identical. Corollary 3 applies when, in addition, vertex weights are equal to one, i.e., $\mathbf{w} = \mathbf{u}_N$ and $\mathbf{m} = \frac{N}{K}\mathbf{u}_K$—the case when function evaluation and simulation running times are assumed to be the same.

Corollary 2. *For the K-partitioning of a graph with Laplacian matrix* $\mathbf{L}$, *vertex weight vector* $\mathbf{w}$ ($\mathbf{W} = diag(\mathbf{w})$), *and partition load vector* $\mathbf{m} = \frac{tr(diag(\mathbf{w}))}{K}\mathbf{u}_K$,

$$C(P^K) \geq \frac{tr(diag(\mathbf{w}))}{2K}\sum_{k=1}^{K-1} \lambda_k\left(\mathbf{S}_N^t\mathbf{W}^{-1/2}\mathbf{L}\mathbf{W}^{-1/2}\mathbf{S}_N\right) \tag{9}$$

where $\mathbf{S}_N$ *is the orthonormal* $N \times (N-1)$ *matrix spanning* $\{W^{1/2}\mathbf{u}_N\}^{\perp}$. *The lower bound is attained for* $\mathbf{X} = \frac{1}{K}\mathbf{u}_N\mathbf{u}_K^t + \sqrt{\frac{tr(diag(\mathbf{w}))}{K}}\ \mathbf{W}^{-1/2}\mathbf{S}_N\mathbf{Y}\mathbf{R}_K^t$, *where* $\mathbf{R}_K$ *is the orthonormal* $K \times (K-1)$ *matrix spanning* $\{\mathbf{u}_K\}^{\perp}$. $\mathbf{Y}$ *is the* $(N-1) \times (K-1)$ *orthonormal matrix whose columns consist of the eigenvectors of* $\mathbf{S}_N^t\mathbf{W}^{-1/2}\mathbf{L}\mathbf{W}^{-1/2}\mathbf{S}_N$ *corresponding to the* $(K-1)$ *smallest eigenvalues in nondecreasing order.*

Corollary 3. *For the* K*-partitioning of a graph with Laplacian matrix* $\mathbf{L}$, *vertex weight vector* $\mathbf{w} = \mathbf{u}_N$, *and partition load vector* $\mathbf{m} = \frac{N}{K}\mathbf{u}_K$,

$$C(P^K) \geq \frac{N}{2K}\sum_{k=2}^{K}\lambda_k(\mathbf{L}) \tag{10}$$

The lower bound is attained for $\mathbf{X} = \frac{1}{K}\mathbf{u}_N\mathbf{u}_K^t + \sqrt{\frac{N}{K}}\mathbf{S}_N\mathbf{Y}\mathbf{R}_K^t$, where $\mathbf{R}_K$ is the orthonormal $K \times (K-1)$ matrix spanning $\{\mathbf{u}_K\}^{\perp}$, and $\mathbf{S}_N$ is the orthonormal $N \times (N-1)$ matrix spanning $\{\mathbf{u}_N\}^{\perp}$. $\mathbf{Y}$ is the $(N-1)\times(K-1)$ orthonormal matrix whose columns consist of the eigenvectors of $\mathbf{S}_N^t\mathbf{L}\mathbf{S}_N$ corresponding to the $(K-1)$ smallest eigenvalues in nondecreasing order.

Under the assumptions of Corollary 3, let $\mathbf{Z}$ be the $N \times (K-1)$ matrix whose columns consist of the eigenvectors of $\mathbf{L}$ corresponding to $\mathbf{L}$'s $(K-1)$ smallest positive eigenvalues in nondecreasing order (so $\mathbf{S}_N\mathbf{Y} = \mathbf{Z}$), then the lower bound is attained for $\mathbf{X} = \frac{1}{K}\mathbf{u}_N\mathbf{u}_K^t + \sqrt{\frac{N}{K}}\mathbf{Z}\mathbf{R}_K^t$. Solutions that relax all but the third constraint in (**P2**), e.g., due to Barnes [4], Bolla [7], and Chan et al. [12], result in the geometric representation $\mathbf{X} = [\frac{\mathbf{u}_N}{\sqrt{N}} : \mathbf{Z}]$ instead.

The most efficient algorithm for computing the eigenvalues and eigenvectors of a large, sparse, and symmetric $N \times N$ matrix is the Lanczos algorithm with $O(N^{1.4})$ runtime [14, 45]. Thus, $O(N^{1.4})$ is the running time for solutions that relax all but the third constraint in (**P2**). Corollary 1 requires computation of $\mathbf{R}_K$ ($O(K^3)$ runtime) and $\mathbf{S}_N$ ($O(N^3)$ runtime), and matrix multiplications with $O(N^2K)$ runtime. Hence, the running time for obtaining the geometric representation $\mathbf{X}$ using Corollary 1 is $O(N^3)$. The running time for Corollary 2 is also $O(N^3)$ since $\mathbf{S}_N$ has yet to be computed. The running time for Corollary 3 is $O(NK^3 + N^2)$ since $\mathbf{R}_K$ and $\mathbf{S}_N$ can be given in closed form [29].

9. Generation of partition from the geometric representation

Corollaries 1, 2 or 3 allows mapping the N vertices of a graph to N points (or geometric representatives) in the K-dimensional Euclidean space, which are given by the N rows of the assignment matrix $\mathbf{X}$. However, entries in matrix $\mathbf{X}$ may not be 0 or 1 since the second constraint in (**P3**) was ignored. It is evident that vertices with geometric representatives close in Euclidean metric tend to have several incident edges in common and should, therefore, belong to the same partition. Moreover, these geometric clusters already account for size constraints on partitions. The following approaches have been proposed to construct a (feasible) partition $\mathbf{X}_f$ of a vertex set from the geometric representation $\mathbf{X}$ obtained by spectral methods.

Closest in Frobenius norm. Rendl and Wolkowicz [47] suggested looking for a feasible matrix $\mathbf{X}_f$ that is as close as possible to $\mathbf{X}$ in Frobenius norm. Because we are considering weighted vertices, we find $\mathbf{X}_f$ by minimizing $\|\mathbf{W}^{1/2}(\mathbf{X}-\mathbf{X}_f)\|_F$. Since

$$\begin{aligned}\|\mathbf{W}^{1/2}(\mathbf{X}-\mathbf{X}_f)\|_F^2 &= \mathrm{tr}(\mathbf{X}^t\mathbf{W}\mathbf{X}) + \mathrm{tr}\big(\mathbf{X}_f^t\mathbf{W}\mathbf{X}_f\big) - 2\mathrm{tr}(\mathbf{X}^t\mathbf{W}\mathbf{X}_f)\\ &= 2\mathrm{tr}(\mathbf{W}) - 2\mathrm{tr}(\mathbf{X}^t\mathbf{W}\mathbf{X}_f)\end{aligned} \tag{11}$$

the linear problem (**P4**): $\mathrm{Max}_{\mathbf{X}_f}\{\mathrm{tr}(\mathbf{X}^t\mathbf{W}\mathbf{X}_f) : \mathbf{X}_f \in \mathfrak{J}\}$ produces a feasible partition $\mathbf{X}_f$ that is close to the geometric representation $\mathbf{X}$. ($\mathfrak{J}$ is the feasible set defined in Lemma 2.)

Linear approximation. Alternatively, since

$$\mathrm{tr}\big(\mathbf{X}_f^t\mathbf{L}\mathbf{X}_f\big) = \mathrm{tr}(\mathbf{X}^t\mathbf{L}\mathbf{X}) + 2\mathrm{tr}(\mathbf{X}^t\mathbf{L}(\mathbf{X}_f-\mathbf{X})) + \mathrm{tr}((\mathbf{X}_f-\mathbf{X})^t\mathbf{L}(\mathbf{X}_f-\mathbf{X})) \tag{12}$$

a feasible partition $\mathbf{X}_f$ can be obtained by neglecting the quadratic term in Eq. (12) and solving the linear problem (**P5**): $\mathrm{Min}_{\mathbf{X}_f}\{\mathrm{tr}(\mathbf{X}^t\mathbf{L}\mathbf{X}_f) : \mathbf{X}_f \in \mathfrak{J}\}$. Note that if $\mathbf{w} = \mathbf{u}_N$ and N is divisible by K, then (**P4**) and (**P5**) become *transportation problems* and, therefore, nonnegativity constraints $x_{ij} \geq 0$ can substitute for the Boolean constraints on the feasible set $\mathfrak{J}$. This approach, also suggested by Rendl and Wolkowicz [47], corresponds to Barnes' method [4].

Minimum cost assignment. Hendrickson and Leland [31] identify a feasible $\mathbf{X}_f$ by solving a mapping problem that we modify as follows. Let V be the set of N weighted vertices in the original graph and P be a set of K vertices representing partitions. A weighted edge connects each vertex in $v_i \in V$ to each vertex $p_k \in P$, with weight equal to the square of the Euclidean distance between the ith row of $\mathbf{X}$ and $(0,\ldots,0,1_k,0\ldots,0)^t$. The optimal mapping is given by the minimum cost of assignment from V to P with the constraint that the sum of the vertex weights of the elements of V mapped to p_k is m_k, for each k in $\{1,\ldots,K\}$. Algorithms for solving this assignment problem terminate in $O(K^3N + K^2N\log N)$ time [60].

KC technique. Alpert and Kahng [1] suggested the following KC algorithm:

```
Initialize W, a set of partition centers, to empty
Choose some random v from V and add it to W
While |W| ≤ K, find v ∈ V such that min_{w∈W} d(v, w) is maximized,
   and add it to W
Form partitions V_1, V_2, ... V_K each containing a single point of W;
   add each v ∈ V to the partition of the closest w_i ∈ W
```

The distance between two vertices, $d(v,w)$, is the Euclidean distance between the corresponding K-dimensional vertex representatives (rows of $\mathbf{X}$). The running time of this algorithm is $O(N\log K)$. A variation of this technique is the K-means classification method [40] in which each new vertex is added to the partition with the nearest mean. The mean of a partition is the mean of its vertex geometric representatives.

KP technique. Chan et al. [12] suggested the KP algorithm which is based on information from the $N \times N$ partition matrix $\mathbf{P} = \mathbf{X}\mathbf{X}^t$. p_{ij} is one if v_i and v_j are in the same partition, and zero otherwise. A set of partition centers is obtained as in the KC algorithm; however, placement of a new vertex v_i in a partition with center v_j is based on how close p_{ij} is from one. The running time of this algorithm is $O(NK^2 + NK \log N)$.

10. Application to powertrain system design

The above hypergraph representation and partitioning techniques have been applied to a powertrain system model proposed by Wagner [64]. This model represents one of the most comprehensive powertrain system studies available in the open literature, containing 87 design relations, 57 design variables, and 62 state/behavior variables. Additional results, although limited to unitary vertex weights, have been presented in [42].

Seven design criteria are considered by Wagner: (1) Fuel consumption and (2) emissions directly affect profits since a vehicle that cannot be certified to meet emissions cannot be sold. (3) The distance a vehicle travels from rest in four seconds and (4) the 5–20 mph time correlate with initial acceleration. (5) The 0–60 mph time correlates with average vehicle acceleration over the speed range of the engine. (6) Starting gradeability is important for markets with hilly or mountainous terrain. (7) Cruising velocity at grade is the speed at which a vehicle can climb a six percent grade in fourth gear.

Design variables describe either geometry or a control strategy. A design relation may be an equality or inequality constraint or the objective function in the powertrain optimization model. A design relation may entail direct evaluation of an algebraic function, access to response surface data, or some kind of simulation. Wagner's powertrain model includes computation of 1 simulation, access to 19 response surface data, and evaluation of 67 algebraic functions. To account for different computational times, we assign the following weights to the corresponding vertices: 1 for algebraic functions, 5 for response surfaces, and 15 for simulation.

The software package *Chaco* [32] was used to identify optimal partitions of the powertrain model. *Chaco* implements several methods for finding small edge separators in weighted (nodes and edges) graphs. We use the spectral method (with minimum cost assignment) together with the FM-HL algorithm described in Section 7.1. In this article we present results for model quadrisection.

Previously to running *Chaco*, a hyperedge model was used to generate a graph adjacency list from the FDT of the model. The hypergraph representation of the powertrain problem contains 87 nodes and 119 hyperedges. The graph representation of the powertrain model contains 87 nodes and 560 edges.

Quadrisection of the powertrain model resulted in four subproblems of sizes 45, 46, 41, and 45 (where the size of a subproblem equals the sum of its design relation weights). Partitioning time was 0.65 sec. on a Sun SPARCstation. The reordered functional dependence table depicted in figure 6 shows that 14 linking variables are needed for model quadrisection. Subproblem 1 contains the torque converter model as well as accessories torque, transmission, and powertrain geometry design relations Subproblems 2 and 4 contain the engine model in addition to the fuel consumption and emissions criteria. Subproblem 3 contains the wheel model together with powertrain geometry, transmission, and vehicle geometry

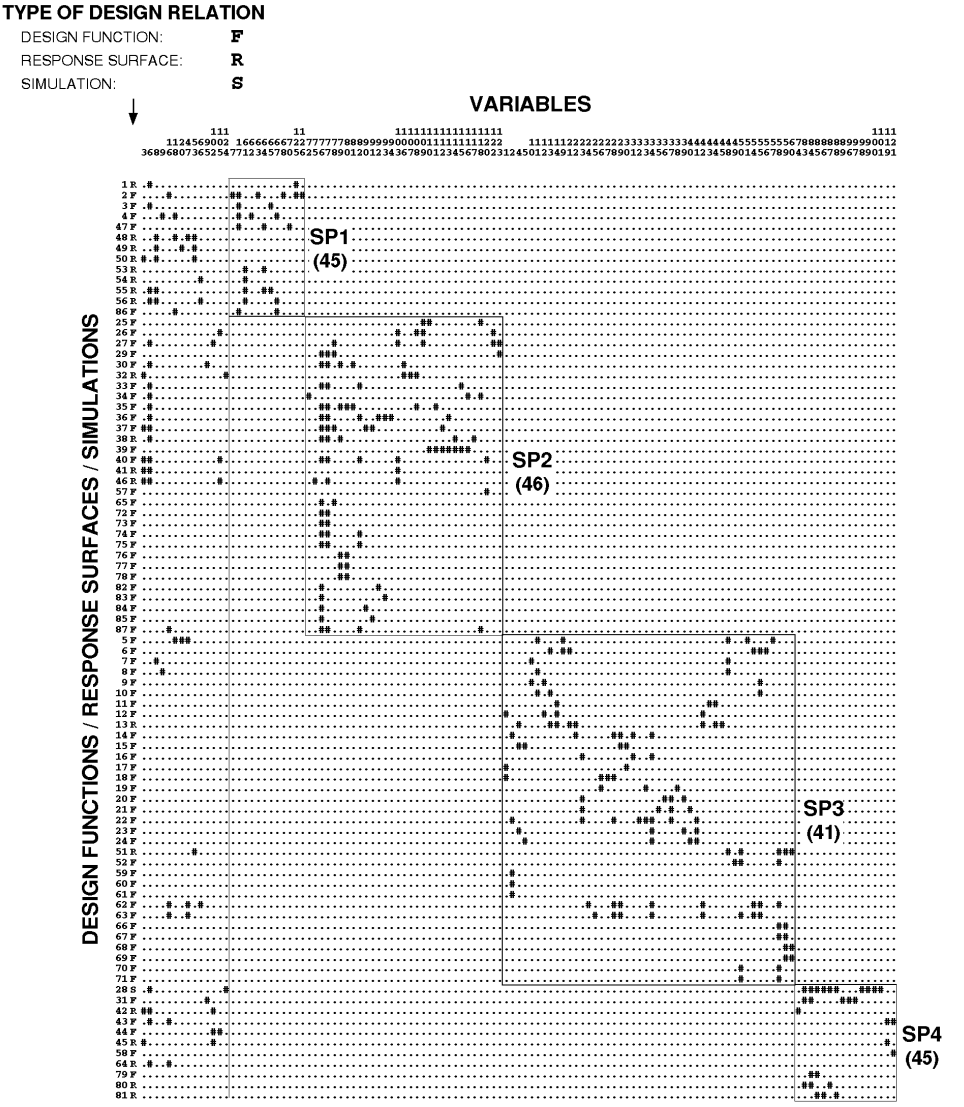

Figure 6. Functional dependence table after quadrisection of powertrain design model.

design relations. Subproblem 3 also includes the acceleration, starting gradeability, and cruising velocity at grade criteria.

11. Conclusions

The article presented a graph-/hypergraph-based methodology for optimal model-based decomposition of design problems. The design problem is represented by a hypergraph

that is then partitioned to identify weakly connected structures implicit in the mathematical design model. Hyperedge models are used to approximate hypergraphs by graphs. Spectral graph-partitioning methods and iterative improvement techniques are proposed for graph/hypergraph partitioning. A known spectral K-partitioning formulation, which accounts for partition sizes and edge weights, is extended to graphs with also vertex weights. The decomposition formulation and solution are robust enough to account for partition loads, function evaluation and simulation times, and the strength of function dependence on variables. Hence, the optimal problem partition may be forced to meet an existing analysis and simulation environment. A vehicle powertrain model was used as example and divided into four parts, two subproblems containing the engine model and the other two being assigned the rest of the model. A typical object or aspect decomposition of this model cannot generate the decomposition obtained by OMBD.

Acknowledgment

This work has been partially supported by research grants from Ford Motor Company and the U.S. Army Automotive Research Center on Modeling and Simulation of Ground Vehicles. This support is gratefully acknowledged.

References

1. C.J. Alpert and A.B. Kahng, "Geometric embeddings for faster and better multi-way netlist partitioning," in Proceedings 30th ACM/IEEE Design Automation Conference, pp. 743–748, 1993.
2. C.J. Alpert and S.-Z. Yao, "Spectral partitioning: The more eigenvectors, the better," in Proceedings 32nd ACM/IEEE Design Automation Conference, June 1995.
3. S.T. Barnard and H.D. Simon, "A fast multilevel implementation of recursive spectral bisection for partitioning unstructured problems," in Proceedings 6th SIAM Conference on Parallel Processing for Scientific Computing, pp. 711–718, 1993.
4. E. Barnes, "An algorithm for partitioning the nodes of a graph," SIAM Journal on Algebraic Discrete Methods, vol. 3, no. 4, pp. 541–550, 1982.
5. E. Barnes, A. Vannelli, and J. Walker, "A new heuristic for partitioning the nodes of a graph," SIAM Journal on Discrete Mathematics, vol. 1, no. 3, pp. 299–305, 1988.
6. C.L. Bloebaum, "Formal and heuristic system decomposition methods in multidisciplinary synthesis," NASA Contractor Report 4413, 1991.
7. M. Bolla, "Spectra, euclidean representations and clusterings of hypergraphs," Discrete Mathematics, vol. 117, pp. 19–39, 1993.
8. R.B. Boppana, "Eigenvalues and graph bisection: An average-case analysis," in Proceedings 28th IEEE Symposium on Foundations of Computer Science, pp. 280–285, 1987.
9. T.N. Bui, S. Chauduri, F.T. Leighton, and M. Sipser, "Graph bisection algorithms with good average case behavior," Combinatorica, vol. 7, no. 2, pp. 171–191, 1987.
10. T.N. Bui, C. Heigham, C. Jones, and T. Leighton, "Improving the performance of the Kernighan-Lin and simulated annealing graph bisection algorithms," in Proceedings 26th ACM/IEEE Design Automation Conference, pp. 775–778, 1989.
11. T.N. Bui and B.R. Moon, "A fast and stable hybrid genetic algorithm for the ratio-cut partitioning problem on hypergraphs," in Proceedings 31th ACM/IEEE Design Automation Conference, pp. 664–669, 1994.
12. P.K. Chan, M.D.F. Schlag, and J.Y. Zien, "Spectral K-way ratio-cut partitioning and clustering," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 13, no. 9, pp. 1088–1096, 1994.
13. R.D. Consoli and J. Sobieszczanski-Sobieski, "Application of advanced multidisciplinary analysis and optimization methods to vehicle design synthesis," Journal of Aircraft, vol. 29, no. 5, pp. 811–818, 1992.

14. J.K. Cullum and R.A. Willoughby, Lanczos Algorithms for Large Symmetric Eigenvalue Computations, Birkhäuser: Boston, vol. 1, 1985.
15. W.E. Donath, "Logic partitioning," Physical Design Automation of VLSI Systems, B.T. Preas and M.J. Lorenzetti (Eds.), Benjamin Cummings: Menlo Park, California, Chap. 3, 1988.
16. W.E. Donath and A.J. Hoffman, "Lower bounds for the partitioning of graphs," IBM Journal of Research and Development, vol. 1, pp. 420–425, 1973.
17. I.S. Duff, A.M. Erisman, and J.K. Reid, Direct Methods for Sparse Matrices, Oxford Science Publications: Oxford, 1989.
18. A. Dunlop and B.W. Kernighan, "A procedure for placement of standard-cell VLSI circuits," IEEE Transactions on Computer-Aided Design, vol. CAD-4, no. 1, pp. 92–98, 1985.
19. S.D. Eppinger, D.E. Whitney, and D.A. Gebala, "Organizing the tasks in complex design projects: Development of tools to represent design procedures," in Proceedings NSF Design and Manufacturing System Conference, Atlanta, Georgia, Jan. 1992.
20. S.D. Eppinger, D.E. Whitney, R.P. Smith, and D.A. Gebala, "A model-based method for organizing tasks in product development," Research in Engineering Design, vol. 6, pp. 1–13, 1994.
21. J. Falkner, F. Rendl, and H. Wolkowicz, "A computational study of graph partitioning," Mathematical Programming, vol. 66, pp. 211–239, 1994.
22. C. Farhat and M. Lesoinne, "Automatic partitioning of unstructured meshes for the parallel solution of problems in computational mechanics," International Journal for Numerical Methods in Engineering, vol. 36, pp. 745–764, 1993.
23. D. Ferrari, "Improving locality by critical working sets," Communications of the ACM, vol. 17, no. 11, pp. 614–620, 1974.
24. C.M. Fiduccia and R.M. Mattheyses, "A linear-time heuristic for improving network partitions," in Proceedings 19th ACM/IEEE Design Automation Conference, pp. 175–181, 1982.
25. M. Fiedler, "Algebraic connectivity of graphs," Czechoslovak Mathematics Journal, vol. 23, pp. 298–305, 1973.
26. M. Fiedler, "Laplacian of graphs and algebraic connectivity," Combinatorics and Graph Theory, Banach Center Publications: Warsaw, vol. 25, pp. 57–70, 1989.
27. O. Goldschmidt and D. Hochbaum, "A polynomial algorithm for the k-Cut problem for fixed k," Mathematics of Operations Research, vol. 19, no. 1, pp. 24–37, 1994.
28. S. Hadley, B. Mark, and A. Vannelli, "An efficient Eigenvector approach for finding netlist partitions," IEEE Transactions on Computer-Aided Design, vol. 11, no. 7, pp. 885–892, 1992.
29. S. Hadley, F. Rendl, and H. Wolkowicz, "A new lower bound via projection for the quadratic assignment problem," Mathematics of Operations Research, vol. 17, no. 3, pp. 727–739, 1992.
30. Y. Haimes, K. Tarvainen, T. Shima, and J. Thadathil, Hierarchical Multiobjective Analysis of Large-Scale Systems, HPC: New York, 1990.
31. B. Hendrickson and R. Leland, "An improved spectral graph partitioning algorithm for mapping parallel computations," SIAM Journal on Scientific Computing, vol. 16, no. 2, pp. 452–469, 1995.
32. B. Hendrickson and R. Leland, "The Chaco user's guide version 2.0," Technical Report SAND94-2692, Sandia National Labs, Albuquerque, New Mexico, 1995.
33. W. Hock and K. Schittkowski, "Test examples for nonlinear programming codes," Lecture Notes in Economics and Mathematical Systems, no. 187, M. Beckman and H.P. Künzi (Eds.), Springer-Verlag: Berlin, 1981.
34. D.S. Johnson, C.R. Aragon, L.A. McGeoch, and C. Schevon, "Optimization by simulated annealing: An experimental evaluation; part I, graph partitioning," Operations Research, vol. 37, no. 6, pp. 865–891, 1989.
35. B.W. Kernighan and S. Lin, "An efficient heuristic procedure for partitioning graphs," The Bell System Technical Journal, vol. 49, pp. 291–307, Feb. 1970.
36. B. Krishnamurthy, "An improved min-cut algorithm for partitioning VLSI networks," IEEE Transactions on Computers, vol. C-33, no. 5, pp. 438–446, 1984.
37. A. Kusiak and J. Wang, "Efficient organizing of design activities," International Journal of Production Research, vol. 31, no. 4, pp. 753–769, 1993.
38. A. Kusiak and J. Wang, "Decomposition of the design process," trans. ASME, Journal of Mechanical Design, vol. 115, pp. 687–695, Dec. 1993.

39. T. Lengauer, Combinatorial Algorithms for Integrated Circuit Layout, John Wiley & Sons: Chichester, 1990.
40. J. MacQueen, "Some methods for classification and analysis of multivariate observations," in Proceedings 5th Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, pp. 281–297, 1967.
41. N. Michelena and P. Papalambros, "A network reliability approach to optimal decomposition of design problems," Trans. ASME, Journal of Mechanical Design, vol. 117, no. 3, pp. 433–440, Sept. 1995.
42. N. Michelena and P. Papalambros, "Optimal model-based partitioning of powertrain system design," Trans. ASME, Journal of Mechanical Design, vol. 117, no. 4, pp. 499–505, Dec. 1995.
43. B. Mohar, "The Laplacian spectrum of graphs," in Proceedings 6th International Conference on Theory and Applications of Graphs, Kalamazoo, Michigan, 1988, vol. 2, pp. 871–898.
44. B. Mohar, "Laplace eigenvalues of graphs—A survey," Discrete Mathematics, vol. 109, pp. 171–183, 1992.
45. B.N. Parlett, The Symmetric Eigenvalue Problem, Prentice-Hall, Inc.: Englewood Cliffs, 1980.
46. A. Pothen, H. Simon, and K. Liou, "Partitioning sparse matrices with eigenvectors of graphs," SIAM Journal on Matrix Analysis and Applications, vol. 11, no. 3, pp. 430–452, 1990.
47. F. Rendl and H. Wolkowicz, "A projection technique for partitioning the nodes of a graph," Research Report CORR 90-20, University of Waterloo, Canada, 1990.
48. F. Rendl and H. Wolkowicz, "Applications of parametric programming and eigenvalue maximization to the quadratic assignment problem," Mathematical Programming, vol. 53, pp. 63–78, 1992.
49. B. Riess, K. Doll, and F. Johannes, "Partitioning very large circuits using analytical placement techniques," in Proceedings 31st ACM/IEEE Design Automation Conference, pp. 646–651, 1994.
50. J.L. Rogers and C.L. Bloebaum, "Ordering design tasks based on coupling strengths," Proceedings 5th AIAA/NASA/USAF/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Panama City, 1994, AIAA paper 94-4326-CP, pp. 708–717.
51. L.A. Sanchis, "Multiple-way network partitioning," IEEE Transactions on Computers, vol. 38, no. 1, pp. 62–81, 1989.
52. L.A. Sanchis, "Multiple-way network partitioning with different cost functions," IEEE Transactions on Computers, vol. 42, no. 12, pp. 1500–1504, 1993.
53. H. Simon, "Partitioning of unstructured problems for parallel processing," Computing Systems in Engineering, vol. 2, no. 2/3, pp. 135–148, 1991.
54. J. Sobieszczanski-Sobieski, "Optimization by decomposition: Step from hierarchic to non-hierarchic systems," NASA TM 101494, Langley Research Center, Hampton, Virginia, 1988.
55. J. Sobieszczanski-Sobieski, "Sensitivity of complex, internally coupled systems," AIAA Journal, vol. 28, pp. 153–160, Jan. 1990.
56. J. Sobieszczanski-Sobieski, "Optimization by decomposition," Structural Optimization: Status and Promise, M.P. Kamat (Ed.), Progress in Astronautics and Aeronautics Series, vol. 150, pp. 487–515, 1993.
57. D.V. Steward, Systems Analysis and Management: Structure, Strategy, and Design, Petrocelli Books, Inc.: New York, 1981.
58. D.V. Steward, "The design structure system: A method for managing the design of complex systems," IEEE Transactions on Engineering Management, vol. EM-28, no. 3, pp. 71–74, 1981.
59. P. Suaris and G. Kedem, "An algorithm for quadrisection and its application to standard cell placement," IEEE Transactions on Circuits and Systems, vol. 35, no. 3, pp. 294–303, 1988.
60. T. Tokuyama and J. Nakano, "Geometric algorithms for a minimum cost assignment problem," in Proceedings 7th ACM Symposium on Computational Geometry, pp. 262–271, 1991.
61. N. Tzannetakis, M. Jensen, and J. Novak, "Development of optimal design methodologies for engine air management," presented at the 15th International Symposium on Mathematical Programming, Ann Arbor, Michigan, 1994.
62. A.J. Vakharia, "Methods of cell formation in group technology: A framework for evaluation," International Journal of Operations Management, vol. 6, no. 3, pp. 257–271, 1986.
63. A. Vannelli and K. Kumar, "A method for finding minimal bottleneck cells for grouping part-machine families," International Journal of Production Research, vol. 24, pp. 387–400, 1986.
64. T.C. Wagner, A General Decomposition Methodology for Optimal System Design, Ph.D. dissertation, Department of Mechanical Engineering and Applied Mechanics, University of Michigan, Ann Arbor, Michigan, 1993.

65. T.C. Wagner and P.Y. Papalambros, "A general framework for decomposition analysis in optimal design," Advances in Design Automation—1993, B.J. Gilmore (Ed.), ASME: New York, vol. 2, pp. 315–325, 1993.
66. T.C. Wagner and P.Y. Papalambros, "Implementation of decomposition analysis in optimal design," Advances in Design Automation–1993, B.J. Gilmore (Ed.), ASME: New York, vol. 2, pp. 327–335, 1993.
67. Y.C. Wei and C.K. Cheng, "Ratio cut partitioning for hierarchical designs," IEEE Transactions on Computer-Aided Design, vol. 10, no. 7, pp. 911–921, 1991.