

POLYHEDRAL APPROACHES TO HYPERGRAPH
PARTITIONING AND CELL FORMATION

A THESIS

SUBMITTED TO THE DEPARTMENT OF INDUSTRIAL ENGINEERING
AND THE INSTITUTE OF ENGINEERING AND SCIENCE
OF BILKENT UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

By
Levent Kandiller
December 1994

714515
QA
402.5
.K36
1994

POLYHEDRAL APPROACHES TO HYPERGRAPH
PARTITIONING AND CELL FORMATION

A THESIS

SUBMITTED TO THE DEPARTMENT OF INDUSTRIAL ENGINEERING
AND THE INSTITUTE OF ENGINEERING AND SCIENCE
OF BILKENT UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

By

Levent Kandiller

December 1994

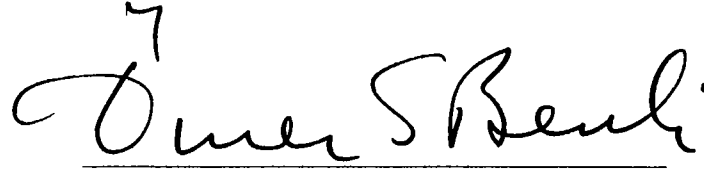
QH
L02.5
.K36
1934

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of Doctor of Philosophy.



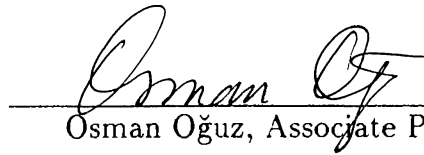
Mustafa Akgül, Associate Professor (Supervisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of Doctor of Philosophy.



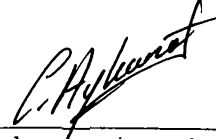
Ömer Benli, Associate Professor

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of Doctor of Philosophy.



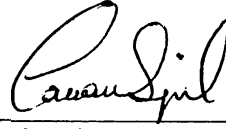
Osman Oğuz, Associate Professor

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of Doctor of Philosophy.



Cevdet Aykanat, Associate Professor

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of Doctor of Philosophy.



Canan Sepil, Associate Professor

Approved for the Institute of Engineering and Science:



Prof. Dr. Mehmet Baray,
Director of Institute of Engineering and Science

Acknowledgement

I am indebted to Assoc. Prof. Mustafa Akgül for his supervision. development of this thesis. I am grateful to Assoc. Prof. Ömer Benli, Asst. Prof. Osman Oğuz, Assoc. Prof. Cevdet Aykanat, Assoc. Prof. Canan Sepil and Assoc. Prof. Barbaros Tansel for their valuable comments.

I would like to extend my deepest gratitude and thanks to my parents and my brother for their morale support. It is to them that this study is affectionally dedicated, without whom it would not be possible.

I would like to express my appreciation to Assoc. Prof. Ülkü Gürler for her suggestions and encouragement throughout the thesis especially at times of despair and hardship. I would like to offer my sincere thanks to my friends Nureddin Kırkavak, Nezh Agan and Oğuzhan Aygün who provided morale support and encouragement.

*Hak'tan gelen şerbeti
İçtik elhamdülillâh
Şol kudret denizini
Geçtik elhamdülillâh*

*Şol karşıki dağları
Meşeleri bağları
Sağlık safâlık ile
Aştık elhamdülillâh*

*Kuru idik yaş olduk
Kanatlandık kuş olduk
Birbirimize eş olduk
Uçtuk elhamdülillâh*

*Vardığımız illere
Şol safâ gönüllere
Halka Tapduk mânâsın
Saçtık elhamdülillâh*

*Beri gel barışalım
Yâd isen bilişelim
Atımız eğerlendi
Eştik elhamdülillâh*

*İndik Rûm'u kısladık
Çok hayr ü şer işledik
Uş bahar geldi geri
Göçtük elhamdülillâh*

*Derildik pınar olduk
İrkildik ırmak olduk
Aktık denize dolduk
Taştık elhamdülillâh*

*Tapduk'un tapusunda
Kul olduk kapusunda
Yûnus miskin çiğ idik
Piştik elhamdülillâh*

Abstract

POLYHEDRAL APPROACHES TO HYPERGRAPH PARTITIONING AND CELL FORMATION

Levent Kandiller

Ph.D. in Industrial Engineering

Supervisor: Mustafa Akgül, Associate Professor

December 1994

Hypergraphs are generalizations of graphs in the sense that each hyperedge can connect more than two vertices. Hypergraphs are used to describe manufacturing environments and electrical circuits. Hypergraph partitioning in manufacturing models cell formation in Cellular Manufacturing systems. Moreover, hypergraph partitioning in VLSI design case is necessary to simplify the layout problem. There are various heuristic techniques for obtaining non-optimal hypergraph partitionings reported in the literature. In this dissertation research, optimal seeking hypergraph partitioning approaches are attacked from polyhedral combinatorics viewpoint.

There are two polytopes defined on r -uniform hypergraphs in which every hyperedge has exactly r end points, in order to analyze partitioning related problems. Their dimensions, valid inequality families, facet defining inequalities are investigated, and experimented via random test problems.

Cell formation is the first stage in designing Cellular Manufacturing systems. There are two new cell formation techniques based on combinatorial optimization principles. One uses graph approximation, creation of a flow equivalent tree by successively solving maximum flow problems and a search routine. The other uses the polynomially solvable special case of the one of the previously discussed polytopes. These new techniques are compared to six well-known cell formation algorithms in terms of different efficiency measures according to randomly generated problems. The results are analyzed statistically.

Keywords: Combinatorial Optimization, Polyhedral Combinatorics, Hypergraph Partitioning, Cellular Manufacturing Systems.

Özet

HİPERÇİZGE PARÇALAMA PROBLEMİNE POLYHEDRAL YAKLAŞIMLAR VE HÜCRE BELİRLENMESİ

Levent Kandiller

Endüstri Mühendisliği Doktora

Tez Yöneticisi: Doç. Dr. Mustafa Akgül

Aralık 1994

Hiperçizgeler, çizgelerin ayrıtların birleştirdiği düğümlerin sayılarının ikiden fazla olabildiği daha genel durumlarıdır. Hiperçizgeler imalat sistemlerinin ve elektrik devrelerinin ifade edilmesinde kullanılırlar. Hücre Tipi İmalat sistemlerinde hiperçizge parçalama hücre belirleme problemine dönüşür. Hiperçizge parçalama entegre devre tasarımında yerleşim problemi kolaylaştırmak için gereklidir. Literatürde çeşitli optimal olmayan çözümler veren sezgisel yöntemler vardır. Bu doktora çalışmasında hiperçizge parçalama problemi için tasarlanmış optimal arayan polihedral kombinatoriks temelli yaklaşımlar tanıtılmıştır.

Hiperçizgeleri ikiye ayırma problemi incelemek için r -düzenli hiperçizgeler üzerinde iki politop tanımlanmıştır. R -düzenli hiperçizgelerde her ayrıt r düğümü bağlar. Bu politopların boyutları, geçerli eşitsizlik aileleri ve yüzey tanımlayan eşitsizlikleri araştırılmış ve bu eşitsizliklerin etkinlikleri rastsal problemler yardımıyla denenmişlerdir.

Hücre belirleme aşaması Hücre Tipi İmalat sistemlerinin tasarımındaki ilk aşamadır. Yeni iki kombinatoriyal optimizasyon temelli hücre belirleme tekniği geliştirilmiştir. Birinci teknik bir çizge ile hiperçizgeye yakınlaşmayı, maximum akış problemlerini arka arkaya çözme yoluyla elde edilen akış eşdeğer ağacı yaratmayı ve bir tarama yordamını kullanmaktadır. İkinci teknik ise daha önce bahsedilen politopun polinom zamanda çözülebilen özel halini kullanmaktadır. Bu iki yeni teknik tanıyan altı hücre belirleme algoritması ile değişik ölçüler bazında rastsal problemlerde karşılaştırılmıştır. Bulgular istatistiksel analizlerle yorumlanmıştır.

Anahtar kelimeler: Kombinatoriyal Optimizasyon, Polihedral Kombinatoriks, Hiperçizge Parçalama, Hücre Tipi İmalat Sistemleri.

Contents

1	INTRODUCTION	1
1.1	General Background and Summary	1
1.2	Notation and Preliminary Results	5
2	PARTITIONING: DEFINITIONS, COMPLEXITIES and TECHNIQUES	10
2.1	Preliminaries	10
2.2	Partitioning Techniques	13
2.2.1	Local Search Heuristics	13
2.2.2	Network Flow Based Approaches	14
2.2.3	Nonlinear Optimization Based Methods	16
2.2.4	Clustering Methods	17
2.2.5	Polyhedral Approaches	18
3	POLYHEDRAL APPROACHES TO PARTITIONING	21
3.1	Partitioning Related Polyhedra	22
3.2	Cut Polytope	23
4	BOOLEAN R-ATIC POLYTOPE	31
4.1	Preliminaries	31
4.2	Definition and Dimension	34
4.3	Facet Defining Inequalities	36
4.4	Clique and Cut Inequalities	40
4.5	Computational Results	48

4.6	A Special Case	52
5	R-UNIFORM HYPERGRAPH CUT POLYTOPE	55
5.1	Dimension	55
5.2	Complete Subhypergraph Inequalities	64
5.3	Trivial Inequalities	65
5.4	Generalized Triangle Inequalities	66
5.5	Congestion Inequalities	71
5.6	Computational Results	74
6	CELL FORMATION PROBLEM	78
6.1	Hypergraph Representation	79
6.2	A Generic Cell Formation Algorithm	81
6.2.1	Hypergraph Approximation	81
6.2.2	Gomory - Hu Cut Tree	84
6.2.3	Generic Algorithm	88
6.3	Sequential Identifying the Best Manufacturing Cell	91
6.4	Efficiency Measures	95
6.5	Other Cell Formation Techniques Evaluated	106
6.6	Problem Generator	117
6.7	Experimental Design	122
6.8	Results and Discussion	125
7	CONCLUSIONS	144
7.1	Contributions	144
7.2	Future Research	147
A		149
A.1	Graph Theory	149
A.2	Linear and Integer Programming	150
Vita		162

List of Figures

1.1	Example complete r -uniform hypergraphs of order 5.	6
1.2	A cut $\delta(S)$ in a hypergraph.	7
1.3	Union operation in H_5^3	9
3.1	A bicycle 5-wheel and its inequality.	24
3.2	A CW_7^1 and its inequality.	29
4.1	An instance of the problem on an example hypergraph representation.	32
4.2	The same instance of the problem on the transformed 5-uniform hypergraph.	34
5.1	A cut in region I.	57
5.2	A cut in region III.	58
5.3	Situations defining homogeneous triangular inequalities.	67
5.4	Congestion inequalities, $r = 4$	72
6.1	The associated hypergraph of the example cell formation situation.	81
6.2	The two representations.	82
6.3	Donath - Hadley approximation of the hyperedges in various sizes.	83
6.4	The clique approximation of the example hypergraph.	84
6.5	The star approximation of the example hypergraph.	85
6.6	First iteration of Gomory-Hu algorithm.	85
6.7	First link of tree diagram.	86
6.8	Second iteration of Gomory-Hu algorithm.	86

6.9	Second link of tree diagram.	87
6.10	A Gomory-Hu cut tree of the clique approximation of the example problem.	87
6.11	A Gomory-Hu cut tree of the star approximation of the example problem.	88
6.12	HAP-CUT{Example,clique,maximize number of partitions,80%}.	90
6.13	HAP-CUT{Example,star,any-criterion,60%}.	91
6.14	First iteration of SIBC{Example,netflo,2/3}.	95
6.15	Second iteration of SIBC{Example,netflo,2/3}.	96
6.16	Some examples of efficiency values.	103
6.17	Effect of part and machine type weights on grouping efficiency values.	104
6.18	Effect of work-loads on balance and under-utilization measures.	105
6.19	Effect of clumpiness on generated incidence matrices.	120
6.20	Effects clumpiness, density, number of parts and algorithms on grouping efficiencies.	136
6.21	Effects clumpiness, density, number of parts and algorithms on work-load balances.	137
6.22	Effects clumpiness, density, number of parts and algorithms on under-utilizations.	138

List of Tables

2.1	A 2×2 contingency table.	17
2.2	Similarity-density functions in clustering binary data.	18
4.1	The stages of computation.	49
4.2	The results of the first set of the computational study on $RP(H_n^r)$ for $n=10,15$ and $r=3,4$	50
4.3	The results of the second set of the computational study on $RP(H_n^r)$ for $n=10,15$ and $r=3,4$	51
5.1	The stages of computation.	74
5.2	The results of the computational study on $P_C(H_n^r)$ for $n=10,15$ and $r=3,4$	76
6.1	Part information of the example problem.	80
6.2	Machine type information of the example problem.	80
6.3	Time comparison of the four alternative methods in Cplex3.0.	94
6.4	Effect of shape parameters on inner-cell and off-diagonal densities.	119
6.5	Factors of the experiment.	124
6.6	The best fine tuning values of cell formation techniques.	124
6.7	Analysis of Variance table for inter-cell flow.	126
6.8	Analysis of Variance table for inner-cell density.	127
6.9	Analysis of Variance table for work-load balance.	128
6.10	Analysis of Variance table for under-utilizations.	129
6.11	Estimated mean differences for the factor levels.	135

6.12 Best cell formation techniques in terms of grouping efficiencies. . .	139
6.13 Best cell formation techniques in terms of work-load balance and under-utilizations.	140

Chapter 1

INTRODUCTION

1.1 General Background and Summary

A *hypergraph* [11] on a set $V = \{1, \dots, n\}$ is a family $H = \{e_1, \dots, e_m\}$ of nonempty subsets e_j of V . The elements of V are *vertices* of H and e_j 's are *hyperedges* of H . The elements of e_j are *end points* of j^{th} hyperedge. Hypergraphs are generalizations of graphs in which each hyperedge may be incident to more than two vertices (nodes).

Hypergraphs are very useful structures in representing incidence relationships of two sets, V and E . The set V is taken as the vertex set and for each element j of the other set E a hyperedge e_j is constructed by connecting vertices which have an incidence relation to the selected element j . That is, $e_j = \{i \in V : i \text{ is related to } j\}$.

Hypergraphs can be used to represent manufacturing environments. The machine set may constitute the vertex set of this representation and the parts may form the hyperedges. In this case, each hyperedge e_j connects the vertices which represent the machines through which the part j is routed. Hypergraphs are also powerful data structures for abstract representation of physical connections. For instance, wires in an electrical circuit usually connect more than two components. Each hyperedge in the hypergraph then corresponds to one electrical connection whereas circuit components are represented by nodes.

Hypergraph partitioning can be used both in grouping the elements of two sets V and E into distinct clusters, and in breaking some physical connections to have disconnected parts. An example of the former case arises in Cellular Manufacturing design, whereas VLSI layout is an instance of the latter case.

Cellular Manufacturing is an application of Group Technology philosophy to the manufacturing environment [4, 97]. In Cellular Manufacturing, parts are identified as families such that design and manufacturing functions can take the advantage of the similarities in a family [96]. Cell formation is the first phase of the design process of Cellular Manufacturing systems. This initial decision on the top of design hierarchy influences all other decisions in Cellular Manufacturing. During this stage, machine groups of functionally dissimilar types are placed together and dedicated to the fabrication of part families.

The cell formation problem can be defined via hypergraphs. The machine types are represented by vertices of its associated hypergraph, and hyperedges represent parts, or vice versa. Various weights can be tagged to the hypergraph representation of the situation. Machine types differ from each other by their operating cost values. Parts are different in their unit profits. The cell formation problem can be defined as the minimum cost hypergraph partitioning when the nodes of the associated hypergraph are machine types, it is the minimum cost (vertex) separation problem when the parts are represented by vertices. In the former case, the associated hypergraph is partitioned into pieces by deleting hyperedges. In a sense, the original manufacturing shop is divided into machine groups by subcontracting removed parts. Here the objective is to subcontract least valuable parts. In the latter case, the separation set made up of nodes divides the hypergraph into parts that are connected only to that separation set. In other words, the conflicting machinery is duplicated so that no part is subcontracted. The objective in this case is to duplicate machinery in the cheapest way.

Another application of hypergraph representation arises from VLSI circuits. In VLSI circuit design, it is assumed that the description of the circuit to be laid out is given in any convenient form. The circuit components are represented by

vertices of the associated hypergraph. The wires are represented by hyperedges. Moreover, one can associate weights with the hypergraph representation of a VLSI circuit. For instance, vertex weights can be the estimated area requirement of the corresponding circuit element. Also, the hyperedges may have wiring costs as weights.

The VLSI layout problem formulations given in the literature [53, 68, 74] are both theoretically and practically intractable. One and the most common heuristic method is to break up the VLSI layout problem into sequential subproblems in such a way that the solution of a subproblem is fed as an input to the next one. Unfortunately, these subproblems are still \mathcal{NP} -Hard. The most common [53, 74, 91] way of breaking up the VLSI layout problem is the following: partitioning \mapsto placement \mapsto global routing and topological compaction \mapsto detailed routing and geometric compaction. In the partitioning phase, the huge hypergraph representing a VLSI circuit is divided into a number of subhypergraphs by cracking the hyperedges having minimum total weights. Since partitioning phase stays on the top of the hierarchy that produces a solution, the final layout is influenced most by the quality of partitioning.

In this dissertation, partitioning problems are defined via a generic mathematical programming formulation. An immediate analysis shows that all hypergraph partitioning problems belong to \mathcal{NP} class. The main motivation in the thesis is to design methods for obtaining optimal solutions to the hypergraph partitioning problems using the principles of polyhedral combinatorics which are proved to be an effective tool for graph partitioning. A detailed analysis of polyhedral techniques for graph partitioning points out the ways to generalize graph partitioning polytopes to hypergraph partitioning polytopes.

Graphs are generalized by r -uniform hypergraphs in which every hyperedge has exactly r end-points. There are two polytopes related with hypergraph partitioning, which are defined on r -uniform hypergraphs. The first polytope, the Boolean R -atic polytope, is investigated for obtaining the best portion of a given hypergraph. A simple transformation from a general hypergraph to an r -uniform hypergraph is detected for which respective solutions are invariant. Therefore

the Boolean R-atic Polytope can be used to solve the problem of picking the best portion for all hypergraphs. This polytope is properly defined, its dimension and facet defining inequalities are investigated. The valid inequalities are tested by means of random problems using both simplex and interior point codes. The results state that more than 90 percent of the cases an integer solution is found before branch and bound. A polynomially solvable special case of this problem is identified, which yields a new problem in Cellular Manufacturing systems: identifying the best manufacturing cell. This leads to a new heuristic for cell formation problem.

The second polytope, the R-uniform Hypergraph Cut polytope, is investigated for solving maximum cut and bipartitioning problems on r-uniform hypergraphs. This polytope is proved to be full dimensional. Families of valid inequalities, some of which define facets are found. The effectiveness of these inequalities are tested for both of the above problems by means of interior point and simplex based procedures. The integer optimal solutions obtained by using these inequalities only are more than 91 percent for the maximum cut problem and more than 76 percent for bipartitioning.

Two new cell formation techniques are developed. One is based on the special case mentioned above on the Boolean R-atic polytope which is proved to be polynomially solvable. The manufacturing cells are sequentially created by solving the problem of identifying the best cell one after another. The second technique involves a graph approximation of the hypergraph that represents the manufacturing environment. A special tree which is flow equivalent to the graph approximation is constructed next. Cells are formed by sequentially deleting the edges in this tree. These new techniques are analyzed and compared to six well-known cell formation techniques. Four different efficiency measures are developed. All the techniques are compared in terms of efficiency values obtained from solving random problems. Algorithms, part demand and operating cost variations, density and manufacturing environment, and number of parts in the system are the considered factors in the experimental design. The statistical analyses state that our measures are effective in evaluating cell formation solutions and each

technique has a region of superiority indicated by the sensitive factors.

The manuscript is organized as follows: In the rest of this chapter hypergraphs are introduced and the basic properties are developed. In Chapter 2, the partitioning problems are defined on a mathematical programming formulation and a literature review of partitioning techniques is presented. Recent polyhedral approaches to graph partitioning problems are analyzed in Chapter 3, with a special treatment of the cut polytope of graphs. Chapters 4 and 5 are devoted to the study of the two polytopes defined for hypergraph partitioning. The two new cell formation algorithms are illustrated and compared with the other cell formation techniques in Chapter 6, which is followed by the conclusions presented in Chapter 7.

1.2 Notation and Preliminary Results

The basic concepts of hypergraph theory and the notation used throughout the thesis are introduced in this section. The conventions used in graph theory, polyhedral theory and linear programming are included in Appendix A.

Let $S \subseteq V$, then $\gamma(S)$ is the set of *internal hyperedges* contained in S . If $\emptyset \neq S, T \subset V$ and $S \cap T = \emptyset$, then $\delta(S; T)$ is the set of hyperedges that have at least one end point from each set. A *cut* $\delta(S)$ in a hypergraph H is the set of hyperedges which have at least one end point in S , i.e., $\delta(S) = \delta(V \setminus S)$. We write $\delta(v)$ instead of $\delta(\{v\})$, call it as *star* of v . The cardinality of the star of a node v is termed as its *degree* and denoted by $deg(v)$.

There are weights associated with nodes and hyperedges. Nodes are differentiated by means of c_i 's, in the node weight associated with $i \in V$. On the other hand, hyperedges are discriminated via d_j 's, in the edge weights associated with e_j . Let $c(S)$ denote the total weight of the nodes in S , and $d(\gamma(S))$ denotes the total weight of the edges inside S .

A hypergraph H is *simple* if no hyperedge contains any other. The *order* of H is $|V| = n$, its *maximal rank* is $r(H) = \max_{i=1, \dots, m} |e_i|$, and its *minimal rank* is $\rho(H) = \min_{i=1, \dots, m} |e_i|$. A hypergraph H is called *r-uniform* if it

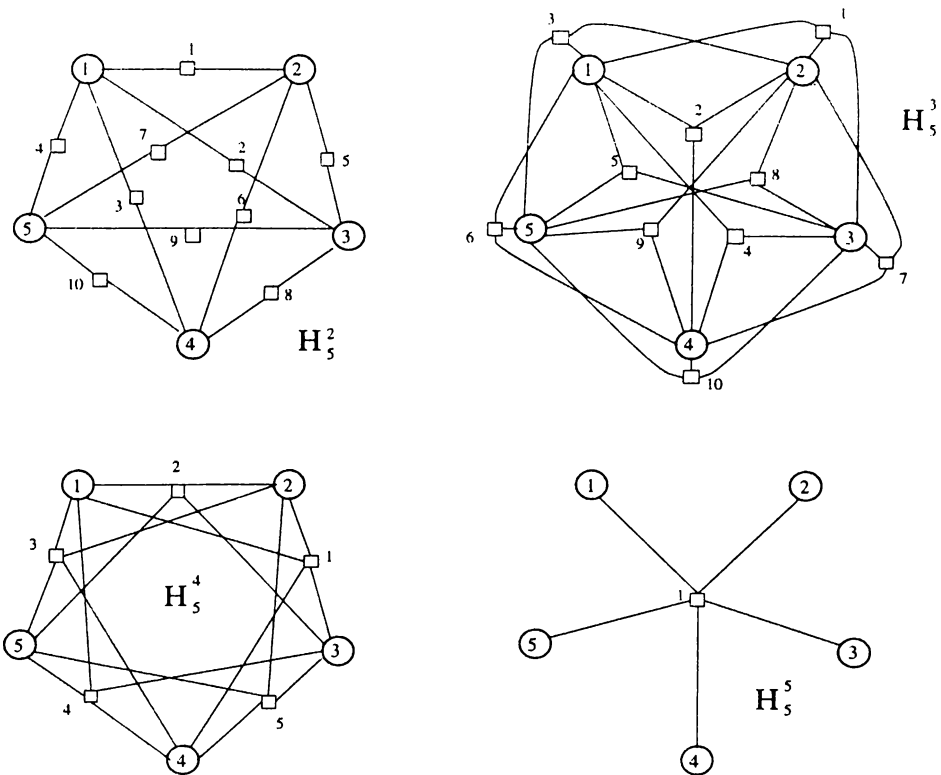


Figure 1.1: Example complete r -uniform hypergraphs of order 5.

is a simple hypergraph such that $|e_i| = r, \forall i = 1, \dots, m$. In an r -uniform hypergraph, a hyperedge can be represented by a sequence of r vertices. An ordinary graph with no isolated vertex is a 2-uniform hypergraph. If there is a hyperedge for every subset of V of size r , then the resultant hypergraph is called *complete r -uniform*, and denoted by H_n^r . There are $m = \binom{n}{r}$ hyperedges in an complete r -uniform hypergraph H_n^r . Without loss of generality we may assume that hyperedges of a complete r -uniform hypergraph H_n^r are lexicographically ordered. So, the first edge of H_n^r is the edge $12 \dots r$ whereas the last edge is the edge $(n - r + 1)(n - r + 2) \dots n$.

Four complete r -uniform hypergraphs of $n = 5$ vertices are illustrated in Figure 1.1. In illustrating hypergraphs, circles represent vertices and squares represent hyperedges. This representation leads to the following observation: each hypergraph is actually a bipartite graph with $n + m$ vertices such that one part is the vertex set of the hypergraph and the other part is the set of hyperedges.

Any cut $\delta(S)$ in H_n^r can be represented by means of five sets as illustrated in Figure 1.2. Two of these sets are node sets: S and $V \setminus S$. The remaining sets

contain hyperedges. A cut divides all hyperedges into three sets: one set for the hyperedges in the cut, and two sets of hyperedges inside of the two parts S and $V \setminus S$.

Let $\mathcal{C}_{H_n^r}^s$ denote the set of all cuts $\delta(S)$ in H_n^r such that $|S| = s$. Clearly, $\mathcal{C}_{H_n^r}^s$ is equivalent to $\mathcal{C}_{H_n^{r-s}}^{n-s}$. Thus, we are interested in $\mathcal{C}_{H_n^r}^s$'s where $1 \leq s \leq \lfloor \frac{n}{2} \rfloor$. For instance, $\mathcal{C}_{H_n^r}^{s=1}$ is the set of stars in the hypergraph. The cardinality of $\mathcal{C}_{H_n^r}^s$ is $\binom{n}{s}$. The number of hyperedges cut by an element of $\mathcal{C}_{H_n^r}^s$ is denoted by $C_{n,r}^s$.

In $\delta(S)$, there are $\min\{r-1, s\}$ subsets $\delta_l(S)$ as illustrated in Figure 1.2. Each $\delta_l(S)$ includes the set of hyperedges in the cut whose exactly l endpoints are in S . Then, the cardinality of $\delta_l(S)$ is $\binom{s}{l} \binom{n-s}{r-l}$. Thus

$$C_{n,r}^s = |\delta(S)| = \sum_{l=1}^{\min\{r-1, s\}} |\delta_l(S)| = \sum_{l=1}^{\min\{r-1, s\}} \binom{s}{l} \binom{n-s}{r-l}.$$

On the other hand, there are $\binom{s}{r}$ hyperedges inside S , and $\binom{n-s}{r}$ hyperedges inside $V \setminus S$. Thus

$$C_{n,r}^s = \binom{n}{r} - \binom{s}{r} - \binom{n-s}{r}.$$

Theorem 1.1 $C_{n,r}^s$ is strictly increasing when $1 \leq s \leq \lfloor \frac{n}{2} \rfloor$ (and is strictly decreasing when $\lfloor \frac{n}{2} \rfloor + 1 \leq s \leq n-1$). $C_{n,r}^s$ is increasing while n is increasing.

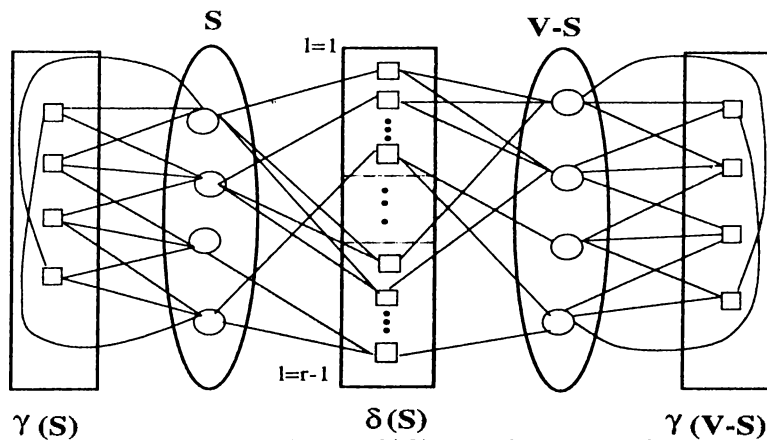


Figure 1.2: A cut $\delta(S)$ in a hypergraph.

Proof:

Let $1 \leq s \leq \lfloor \frac{n}{2} \rfloor$. $C_{n,r}^{s+1} = \binom{n}{r} - \binom{s+1}{r} - \binom{n-s-1}{r}$, and $C_{n,r}^s = \binom{n}{r} - \binom{s}{r} - \binom{n-s}{r}$. Recall that $\binom{n}{k} + \binom{n}{k-1} = \binom{n+1}{k}$. Then,

$$C_{n,r}^{s+1} - C_{n,r}^s = -\binom{s+1}{r} + \binom{s}{r} - \binom{n-s-1}{r} + \binom{n-s}{r} = \binom{n-s-1}{r-1} - \binom{s}{r-1} \geq 0.$$

Thus $C_{n,r}^{s+1} \geq C_{n,r}^s$, $1 \leq s \leq \lfloor \frac{n}{2} \rfloor$. This completes the first part.

$C_{n+1,r}^s = \binom{n+1}{r} - \binom{s}{r} - \binom{n+1-s}{r}$, and $C_{n,r}^s = \binom{n}{r} - \binom{s}{r} - \binom{n-s}{r}$. Recall that, $\binom{n}{r} = \binom{n-1}{r-1} + \cdots + \binom{n-m}{r-1} + \binom{n-m}{r}$. Then,

$$\begin{aligned} C_{n+1,r}^s - C_{n,r}^s &= \left[\binom{n+1}{r} - \binom{n+1-s}{r} \right] - \left[\binom{n}{r} - \binom{n-s}{r} \right] \\ &= \left[\binom{n}{r-1} + \cdots + \binom{n+1-s}{r-1} \right] - \left[\binom{n-1}{r-1} + \cdots + \binom{n-s}{r-1} \right] \geq 0. \end{aligned}$$

The above inequality is from the fact that each term on the left is greater than each term on the right respectively, i.e., $n+1-i > n-i$, $\forall i = 1, \dots, s$. Thus, $C_{n+1,r}^s \geq C_{n,r}^s$, $\forall n$. \square

Corollary 1.1 For $1 \leq s \leq n-1$,

$$C_{n,r}^1 = \binom{n-1}{r-1} \leq C_{n,r}^s \leq \binom{n}{r} - \binom{\lfloor \frac{n}{2} \rfloor}{r} - \binom{\lceil \frac{n}{2} \rceil}{r} = C_{n,r}^{\lfloor \frac{n}{2} \rfloor}.$$

An *incidence vector* $\mathcal{X}(S)$ of a set $S \subset V$ is a binary vector of length n whose i^{th} component is 1 if $i \in S$, and 0 otherwise. Similarly, an *incidence vector* $\mathcal{X}(\gamma(S))$ of a $\gamma(S)$ of $S \subset V$ is a binary vector of length $\binom{n}{r}$ whose i^{th} component is 1 if $e_i \subseteq S$, and 0 otherwise. Let $\mathcal{X}(C_{H_n}^s)$ be the *incidence matrix of cuts of size s in H_n^r* . This incidence matrix has $\binom{n}{s}$ rows and $\binom{n}{r}$ columns. Both rows and columns of $\mathcal{X}(C_{H_n}^s)$ are ordered lexicographically. Rows are indexed by $s[i]$, i.e., $s[i]$ is the set S whose incidence vector of the cut $\mathcal{X}(\delta(S))$ is the i^{th} row of $\mathcal{X}(C_{H_n}^s)$.

We denote by \mathbb{R} the set of real numbers. The set of natural (respectively, binary) numbers are denoted by \mathbb{N} (respectively, \mathbb{B}). Let \mathbb{R}^n , \mathbb{N}^n , \mathbb{B}^n be the set of vectors with n components. Let $u_i \in \mathbb{B}^{n+\binom{n}{r}}$ be a canonical unit vector whose first n

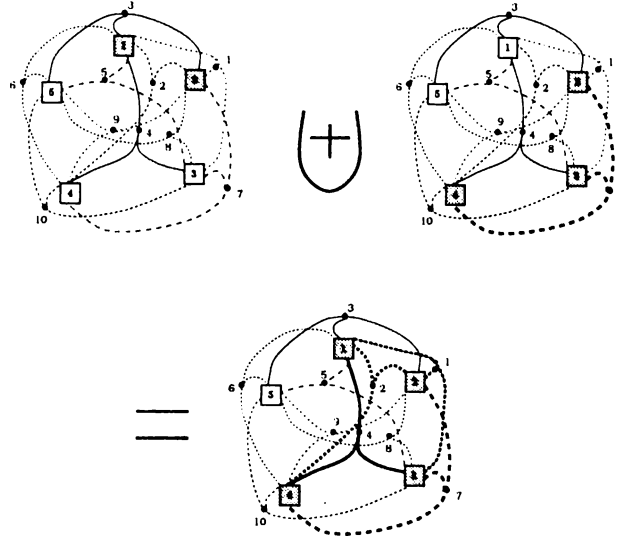


Figure 1.3: Union operation in H_5^3 .

components are $\mathcal{X}(\{i\})$ and others are zero. Similarly, let $v_j \in \mathbb{B}^{n+\binom{n}{r}}$ be a vector whose first n components are $\mathcal{X}(e_j)$ and last $\binom{n}{r}$ components form a canonical unit vector $\mathcal{X}(\gamma(e_j) = e_j)$. Let $z^S = (\mathcal{X}(S), \mathcal{X}(\gamma(S)))$, and $z^T = (\mathcal{X}(T), \mathcal{X}(\gamma(T)))$. Then, the *union operation* is defined as $z^S \uplus z^T = (\mathcal{X}(S \cup T), \mathcal{X}(\gamma(S \cup T)))$. The union operation in H_5^3 is illustrated in Figure 1.3:

$$z^1 = (1, 1, 0, 0, 0 | 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$$

$$z^2 = (0, 1, 1, 1, 0 | 0, 0, 0, 0, 0, 0, 1, 0, 0, 0)$$

$$z^1 \uplus z^2 = (1, 1, 1, 1, 0 | 1, 1, 0, 1, 0, 0, 1, 0, 0, 0)$$

If A is an $n \times m$ matrix, then A_i denotes i^{th} row of A , A^j denotes j^{th} column of A , and A_i^j denotes $(i, j)^{th}$ element of A . $\mathbf{1}$ is a matrix whose elements are all equal 1. Similarly, e is the vector whose components are all ones, and e_i, e_{ij}, e_{ijk} are appropriate canonical unit vectors. $\mathbf{0}$ is a matrix whose elements are all equal 0. I_n is the identity matrix of size n . A^* is an operation similar to taking a transpose: $(A^*)^i = A_{n+1-i}$. For instance,

$$(I_3)^* = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}.$$

Chapter 2

PARTITIONING: DEFINITIONS, COMPLEXITIES and TECHNIQUES

2.1 Preliminaries

In this section, a family of partitioning problems is defined from a combinatorial optimization viewpoint. A partitioning problem is referred as *hypergraph* partitioning when the underlying structure is a hypergraph. It is called *graph* partitioning when the associated data structure is a (simple) graph.

The partitioning of a (hyper) graph is obtained either by deleting (hyper) edges, or deleting vertices and their incident (hyper) edges. The former operation is known as *partitioning* whereas the latter one is called *separation*. In our taxonomy, there are three kinds of partitioning: bipartitioning, multiple partitioning and free partitioning. The cost of the partition operation is the total costs of the removed elements. Another criterion is the *balance* criterion which tries to make subparts to have equal number of vertices as much as possible.

Definition of some symbols are needed before defining *separation* and the

three kinds of *partitioning*: A (hyper) graph $G = (V, E)$ consisting of $n = |V|$ vertices is used to represent the circuit to be partitioned into $p \in \mathbb{N}$ parts. For each edge $e \in E$, the incident vertices are denoted by $v(e)$. There is an edge cost function $c : E \mapsto \mathbb{N}$, a vertex weight function $w : V \mapsto \mathbb{N}$. There are also size limits for parts: $s(j) \in \mathbb{N}$ is the lower limit and $S(j) \in \mathbb{N}$ is the upper limit for part $j = 1, \dots, p$.

In **multiple partitions**, the node set V is partitioned into $p > 1$ disjoint nonempty sets satisfying the size limits $s(i)$ and $S(i)$. In order to formulate this as a mathematical program, a binary decision variable $x_{i,j}$ is needed to be defined:

$$x_{i,j} \doteq \begin{cases} 1, & \text{if } i^{\text{th}} \text{ vertex is assigned to } j^{\text{th}} \text{ part;} \\ 0, & \text{otherwise.} \end{cases}$$

Multiple partitioning problem is formalized by the following mathematical formulation:

$$\text{Maximize } \sum_{j=1}^p \sum_{e \in E} c_e \prod_{i \in v(e)} x_{i,j} \quad (2.1)$$

subject to:

$$\sum_{j=1}^p x_{i,j} = 1 \quad \forall i = 1, \dots, n \quad (2.2)$$

$$s(j) \leq \sum_{i=1}^n w(i) \cdot x_{i,j} \leq S(j) \quad \forall j = 1, \dots, p \quad (2.3)$$

$$\begin{aligned} x_{i,j} &= 0, 1 & \forall i = 1, \dots, n \\ & & \forall j = 1, \dots, p \end{aligned} \quad (2.4)$$

The objective of the multiple partitioning formulation is to minimize the total cost of the (hyper) edges that are in the cut set. In the formulation, a complementary objective is used. The objective is to maximize the total weight of the (hyper) edges not cut by the partition. The product in the objective function identifies whether the edge has all of its incident vertices inside a part. The constraints (2.2) satisfy the partition criterion. That is, every vertex is assigned to only one part, and the disjoint parts span all vertices. The constraints (2.3) are size constraints and the constraints (2.4) are integrality constraints.

Not surprisingly, the multiple partition problem is \mathcal{NP} -Hard, even if it is severely restricted. For instance, the multiple partition problem in which G is a graph, $w \equiv 1$, $c \equiv 1$, $s(i) = 1$ and $S(i) = n$, is strongly \mathcal{NP} -Complete [74].

The special case of the multiple partition problem in which $p=2$ is called the **bipartition** problem. The additional restrictions – n is even, $s(i) = 1$, $S(i) = \frac{n}{2}$, $w \equiv 1$, define the *bisection* problem. The hypergraph bipartitioning is also \mathcal{NP} -Hard. In fact, the bisection problem restricted to graphs is strongly \mathcal{NP} -Complete [74].

There is a variant of the multiple partition problem in which p is not specified. In this case, $s(i)$ values forcing balanced partitions cannot be defined, and usually upper limits on part sizes $S(i)$ are the same S . This variant is called the **free** partition problem. The free partition problem is also \mathcal{NP} -Hard. Actually, The associated decision problem is strongly \mathcal{NP} -Complete [74] even if $S \geq 3$ is fixed, $w \equiv 1$, $c \equiv 1$.

The **separation** problem is to remove as few vertices (the set C) as possible such that the (hyper-)graph is disconnected into two sides (the sets L and R), each of which has a total vertex weight not exceeding S . The separation problem belongs to \mathcal{NP} -Hard class [74].

Multiple partitioning restricted to planar graphs where $w \equiv 1$, $c \equiv 1$, $s(i)=1$, and $S(i)=n$, for $p=3$ and vertex degree of at most 4 can be solved in polynomial time. The bipartition problem for trees can be solved in quadratic time using dynamic programming techniques. The complexity of bisection problem restricted to planar graphs is open. It is also not known whether the separation problem for planar graphs is \mathcal{NP} -Complete. Lipton and Tarjan [76] presented a linear time algorithm for separating planar graphs with a separator of size $O(\sqrt{n})$. The tree partition problem is weakly \mathcal{NP} -Complete. If all vertex weights are the same, the free partition problem of planar graphs is in \mathcal{P} .

2.2 Partitioning Techniques

The approaches reported in the literature has focused mainly on the graph bipartitioning problem. There are very powerful iterative improvement heuristics, bounding schemes, and network flow based approaches. The methods proposed to solve the bipartitioning problem are modified to handle the multiple partitioning problem. On the other hand, the node separation problem has not attracted the researches' interest much. Furthermore, the free partitioning problem remains almost untouched.

2.2.1 Local Search Heuristics

The most famous local search technique designed to solve the graph bisection problem is due to Kernighan and Lin [63]. This technique has served as a comparison base for the last 20 years. It starts with an initial bisection, and tries to exchange pairs of vertices across the cut of bisection. Kernighan and Lin proposed to calculate the gains (reduction in the cost) of all possible pair interchanges and to select the one with the maximum gain or the one with the smallest increase, to reduce the chance of being trapped in a local optima. Then the selected pair is locked, and a second vertex pair whose exchange improves the cut of the bisection is searched, and so on.

Although this heuristic in its original version is quite simple and effective, there are some disadvantages as reported in [74]. The Kernighan-Lin heuristic handles only unit weights, operate on exact bisection, cannot process hypergraphs, myopic and greedy, and finally one pass is not linear. Later on, Schweikert and Kernighan [88] discussed the ways of extensions to handle the hypergraphs by approximating hypergraphs via cliques.

Ten years later, Fiduccia and Mattheyses come with the idea of moving only a single vertex across the cut in a single move [37], and updating the gain calculations to handle the hypergraphs. They used buckets to make their algorithm run in linear time. Krishnamurty [67] introduced more look ahead mechanisms into this heuristic.

Kernighan and Lin [63] suggested a way to achieve balanced multiple partitions by applying the heuristic to pairs of multiple subsets to improve the partition. However, this method seems not promising because of computational intractability, that is the time it takes until a stabilization occurs is too long.

The analogy between a combinatorial optimization problem and the problem of determining lowest energy ground state of a physical system with many interacting atoms was first observed by Kirkpatrick *et al.* [66] in 1983. They developed a local search technique called simulated annealing using a very early result of Metropolis *et al.* [78]. Johnson *et al.* tailored general simulated annealing algorithm for the bipartitioning problem [55] by moving a single vertex at a time across the cut. They penalized the squared difference between sizes of parts, added to the objective function, and performed a local search on this objective function. Penalizing the imbalance in this way is quite similar to a trivial Lagrange relaxation method. They reported on detailed experiments using appropriate search strategies to bipartition graphs with unit vertex weights. The results seem not promising.

2.2.2 Network Flow Based Approaches

Network optimization approaches to the partitioning problem are usually based on the maximum flow - minimum cut theorem. These approaches attempt to solve underlying maximum flow between specified pair of nodes yielding a minimum cut. This cut bipartitions the node set in such a way that each node of the pair lies in different parts. The above method has two main disadvantages. It is very difficult to adjust the sizes of each part, and fixing the terminal pair of nodes is troublesome.

Bui *et al.* [18] suggested the following method for bisections to take care of size restrictions. After fixing a pair of distinguished nodes, shrink the neighborhood of size s around these nodes. That is, shrink all vertices that can be reached from the distinguished node along the paths of length at most s edges. Consequently, assign infinite capacities to the edges of the resulting graph that

are incident to the shrunk nodes, and make a maximum flow computation. Output the smallest minimum cut over all vertex pairs that is a bisection. Kahng [56] extended this idea to treat hypergraph bisections.

The idea of flow equivalent cut tree [41] is used in the partitioning problem. The crude ideas appeared first in [54]. Recently Vanelli and Hadley represented the same idea [93]. This idea is used to reduce the time complexity of solving a maximum flow problem for each distinct pair of vertices of the graph to be partitioned.

Garbers *et al.* [38] suggested to use another approach for free partitioning, using k - l -connectedness. Two vertices u and w are called k - l -connected if and only if there exists k edge disjoint paths connecting u and w such that each of these paths has length at most l . They suggested to cluster vertices that are k - l -connected to each other as a part. Unfortunately, they did not investigate the ways of performing such an operation. However they only indicate that this kind of partition can be obtained in polynomial time when k and l are fixed.

Multi-commodity flow techniques can be used to assist graph bipartitioning [74]. A special commodity is assigned to each pair of vertices having demands/supplies of $\frac{z}{2}$. The edge costs of the multi-commodity problem are ignored and capacities are kept equal to edge costs of the partitioning problem. The objective is to maximize z . For each bipartition (V', V'') , the minimum multi-commodity flow across the cut is $z \cdot |V'| \cdot |V''|$. On the other hand, it is necessary to have some nonnegative slack in the arc capacities for a multi-commodity flow to exist; that is $C(V', V'') - z \cdot |V'| \cdot |V''| \geq 0$, where $C(V', V'')$ is the capacity of the cut (V', V'') . The above argument creates an upper bound to the maximum flow:

$$z \leq \min_{(V', V'')} \frac{C(V', V'')}{|V'| \cdot |V''|}.$$

The term at the right hand side is called the sparsity of a cut. A cut with the minimum sparsity is the sparsest cut. Since only one direction of the maximum flow minimum cut theorem holds for the multi-commodity flows, the maximum value of a multi-commodity flow is only a lower bound on the minimum sparsity over all cuts.

The notion of the sparsest cut relaxes the balance criterion. Cuts of any balance are allowed; however, the more balanced a cut is, the higher the denominator is, and the sparsest a cut is. Thus, the sparsest cut is likely to be an attractive cut for balanced bipartitioning. The investigation of the connections between partitioning and multi-commodity flow is still in its infancy as reported in [74].

2.2.3 Nonlinear Optimization Based Methods

A number of lower bounds are developed on the minimum cut sizes of the multiple graph partitioning problem formulated as a quadratic assignment problem. The bounds are stated in terms of eigen values ($\lambda_i(C), i = 1, \dots, n$) of the cost matrix (C) that are sorted in decreasing order. The bounds are usually based on the Hoffman-Wieland inequality: *Let A and B are two real symmetric matrices, then $Trace(A \cdot B^T) \leq \sum_{i=1}^n \lambda_i(A) \cdot \lambda_i(B)$, where $\lambda_i(\cdot)$ is the i^{th} largest eigen value of (\cdot) .*

Barnes [9] suggested the following lower bound: $c(E) - \frac{1}{2} \sum_{i=1}^p S(i) \cdot \lambda_i(C)$, where $c(E)$ is the sum of the costs of all edges. Recall that p denotes number of parts and $S(i)$ denotes upper bound on the size of i^{th} part in the multiple partition. Barnes [10] reported a procedure to obtain tight lower bounds on the number of edges that must be cut when the nodes of a given graph are multiple partitioned into the parts that are sorted in decreasing order of their fixed sizes. The procedure substitutes the decision variables of the quadratic assignment formulation by orthonormal decision variables, and constraints by orthogonal eigenvectors.

Donath and Hoffman [36] suggested a more general lower bound using a diagonal matrix U such that $Trace(U) = 2c(E) - \frac{1}{2} \sum_{i=1}^p S(i) \cdot \lambda_i(C - U)$. They observed that this bound is a concave function of the diagonal entries of the matrix U . This fact helps to employ nonlinear optimization methods to maximize the bound family by a good choice of the diagonal entries of U . Boppana [13] extended this lower bounding scheme to derive tight bounds on the cutsize of bisections.

2.2.4 Clustering Methods

Clustering is grouping objects such that the groups have objects that are close to each other in certain specified properties. If the node set of a (hyper)graph is viewed as the set of objects and the edges in between them are treated as properties, or vice versa, the multiple partitioning is a special kind of clustering.

A hypergraph can be represented by its incidence matrix. A specific element (i, j) of the incident matrix A identifies whether the node i is incident to the hyperedge j ($A(i, j) = 1$) or not ($A(i, j) = 0$). Either the columns of A can be viewed as objects to be clustered or the rows are to be clustered. In any case, there are a number of binary vectors to be grouped.

There is a large amount of work in the area of clustering [1, 62]. A clustering algorithm depends on two factors: the distance function and the method of representing clusters. When computing a similarity $\bar{d}(o_1, o_2)$ or dissimilarity (distance) $d(o_1, o_2)$ between two objects o_1 and o_2 that take two values (0,1) in variables denoting existence properties, one draws a 2×2 contingency table such as in Table 2.1. Here a is the number of variables that are equal to 1 for both objects. Similarly, b , c , and d are defined. Obviously, $a + b + c + d = t$, the total number of variables. The values a , b , c , and d are combined in a coefficient describing to what extent objects o_1 and o_2 agree with regard to the collection of binary variables. Table 2.2 gives a number of similarity and distance functions reported so far in the clustering literature.

The method of identifying a specific cluster usually names itself. If a representative object in a cluster is selected, the technique is called representative

		object o_2		
		1	0	
object o_1	1	a	b	a+b
	0	c	d	c+d
		a+c	b+d	t

Table 2.1: A 2×2 contingency table.

Name	$d(o_1, o_2)$	$d(o_1, o_2)$
Zubin (1938), Dumas (1955), Sokal and Michener (1958)	$\frac{a+d}{a+b+c+d}$	$\frac{b+c}{a+b+c+d}$
Sneath (1962), Hill <i>et al.</i> (1965).		
Rogers and Tanimoto (1960).	$\frac{a+d}{(a+d)+2(b+c)}$	$\frac{b+c}{(a+d)+2(b+c)}$
Sokal and Sneath (1963), Duran and Odell (1974).	$\frac{2(a+d)}{(a+d)-(b+c)}$	$\frac{(b+c)}{2(a+d)+(b+c)}$
Hamman	$\frac{a+b+c+d}{a+d-bc}$	-
Yule	$\frac{ad+bc}{a+d}$	-
Back, Romezbug	$\frac{a+d}{(b+c)+2(a+d)}$	-

Table 2.2: Similarity-density functions in clustering binary data.

partitioning. If the centroid of a cluster is used, the underlying method is named partitioning around medoids, and so on.

In measuring inter-cluster similarity (distance), there are again several ways. Group average is the average of similarities (distances) inbetween every pair of objects. The similarity (distance) between the nearest neighbor pair [single linkage methods] and the similarity (distance) between the furthest pair [complete linkage methods] are other ways of measuring similarities (distances) between clusters that are used in hierarchical clustering methods.

Kwaterna and Simeone [72] developed a clustering based technique first to solve a graph related combinatorial optimization problem. They proposed clustering methods as a new class of set covering heuristics. In the meantime, Simeone with Antenucci and Nicoloso applied elementary clustering heuristics in hypergraph partitioning [2]. Feo and Khellaf [74] investigated the relationship between clustering, matching (a special case of clustering in which the size of each cluster is fixed at 2) and partitioning.

2.2.5 Polyhedral Approaches

Polyhedral approaches are based on the determination of the minimal representation of the constraint set via facet defining inequalities provided that the

associated polyhedra is full dimensional. After determining the set of “easy-to-list” facet defining equalities, one can find a solution to that constraint set and checks whether the solution satisfies all facet defining inequalities.

Works on cut polytope has begun with the appearance of the *max-cut* problem: Given a graph $G=(V,E)$ and a weight function $w:E\rightarrow\mathbb{Z}$, the problem of finding a cut $\delta(S)$ such that its weight $\sum_{e\in\delta(S)} w(e)$ is as large as possible. Yannakakis [98] showed that this problem is \mathcal{NP} -Hard even for graphs having degree at most three. On the other hand, Hadlock [49] proved that the max-cut problem can be solved polynomially for planar graphs. It is also polynomial for weakly bipartite graphs [46], and for graphs without odd long cycles [45].

Barahona [6] used the decomposition schema [95] for graphs not contractible to the complete graph with 5 nodes K_5 , and facial description of bipartite subgraph polytope $P_B(G)$ [44], to derive a polynomial algorithm for the graphs not contractible to K_5 . His procedure is quite similar to that of Cornuéjols *et al.*'s [27] which is designed for solving Travelling Salesman Problem in graphs with 3-edge cutsets. He also proved that the max-cut problem in graphs with no subgraph contractible to K_6 is \mathcal{NP} -Hard. Later, Barahona and Mahjoub [8] gave complete polyhedral description of the cut polytope $P_C(G)$ for the graphs not contractible to K_5 . They argued that some facet defining inequalities of $P_B(G)$ work in describing $P_C(G)$. Furthermore they developed a facet defining procedure to characterize facets associated with edges and cycles and proved that these facets uniquely define the cut polytope $P_C(G)$ for graphs not contractible to K_5 . Based on the above partial characterization of the cut polytope for general graphs, Barahona *et al.* [7] designed a cutting plane algorithm and report on computational experience with it. Boros and Hammer [14, 15], Deza and Laurent [33, 34] presented a class of valid inequalities as well as a class of facets for the cut polytope of the complete graph $P_C(K_n)$.

If the set to be clustered X is made up of a finite number of points in a Euclidean space, the partition of X_1, X_2, \dots, X_p of X into p groups which maximizes the sum of unsimilarities of all those pairs which do not belong to the same group is studied by Boros and Hammer [16]. They discussed that there are cases that

the optimal partition has special properties which make the optimization problem easier. They reported two cases. One such case consists of the minimization of the objective function which is a quasi-convex function depending on the sum of the coordinates of the points in each set of the partition. In this case, the unsimilarity function has no influence on the problem: $\text{conv } X_i \cap \text{conv } X_j = \emptyset$ for $i \neq j$. Another case is that the non-negative real valued dissimilarity function is defined between all pairs of points in X . In this case, there exists an optimal partition such that the intersection of X_j with the convex hull of X_i is empty for all $i < j$.

Grötschel and Wakabayashi [42] transformed the clustering problem to a clique partitioning problem. A set A of edges in a graph $G=(V, E)$ is called a *clique partitioning* of G if there is a partition of $V=W_1 \cup W_2 \cup \dots \cup W_p$ upon the removal of A such that the subgraph induced by W_i is a clique for $i=1, \dots, p$. In case G is complete, every partition of the node set of G induces a clique partitioning. The clique partitioning problem is then defined as follows: Given a complete graph $K_n=(V_n, E_n)$ with weights $w_e \in \mathbb{Z} \forall e \in E_n$, find a clique partitioning $A \subseteq E_n$ such that $w(A)$ is as small as possible. Grötschel and Wakabayashi [43, 47] investigate facet defining inequalities of the clique partitioning polytope $P_K(K_n)$. They proposed a cutting plane algorithm and reported on its applications and computational results [42].

Chapter 3

POLYHEDRAL APPROACHES TO PARTITIONING

Polyhedral Combinatorics deals with the interactions between linear (affine) algebra, linear programming and combinatorial optimization. A problem of *combinatorial optimization* generally has the following form. There is a finite ground set G , a family \mathcal{F} of “feasible” subsets of G and w_ϵ associated with each element $\epsilon \in G$. The aim is to find an optimal set S^* among all feasible sets $S \in \mathcal{F}$, for which $\sum_{\epsilon \in S} w_\epsilon$ is maximized/minimized over all members of \mathcal{F} . A fundamental theorem in polyhedral theory states that *every polytope is the solution set of a finite system of linear inequalities and equations*. If one is successful to find such a system, then it is sufficient to solve the combinatorial optimization problem at hand by using a linear programming technique.

A technique based on polyhedral combinatorics usually employs a five step process:

- S1. Represent the members of \mathcal{F} by vectors, usually 0–1 incidence vectors of the sets $S \in \mathcal{F}$;
- S2. Define the polyhedron P to be the convex hull of the vectors of S1;
- S3. Obtain a linear system sufficient to determine P ;
- S4. Apply linear programming duality to obtain optimality criteria special to the problem at hand;

S5. Develop an algorithm using the optimality criterion as a stopping condition.

In this chapter, studies on cut polytope induced by the max-cut problem are reviewed. In Section 3.1, a polytope family related with the graph partitioning problem is introduced. In the last section, current results on the cut polytope are presented.

3.1 Partitioning Related Polyhedra

In this section, various polytopes related with graph partitioning are listed together with their intersection pattern. These are cut polytope, bipartite subgraph polytope, equipartition polytope, clique partitioning polytope, multi-cut polytope, free cut polytope and boolean quadratic polytope.

The bipartite subgraph polytope $P_B(G)$ is the convex hull of incidence vectors of bipartite subgraphs of $G = (V, E)$ [44]. Similarly, the cut polytope $P_C(G)$ is the convex hull of incidence vectors of bipartitions of G [8]. Since any bipartition or cut $\delta(S)$ defines a bipartite subgraph of G which is $(S, V \setminus S, \delta(S))$, $P_C(G) \subseteq P_B(G)$. However, in general $P_C(G) \neq P_B(G)$. Equipartition polytope $P_E(G)$ is the convex hull of incidence vectors of bisections of G [25, 26]. Clearly, $P_E(G) \subseteq P_C(G)$.

The multi-cut polytope $P_C^k(G)$ is the convex hull of incidence vectors of multiple partitions of G when the number of parts is exactly k [24]. Thus, $P_C(G) = P_C^{k=2}(G)$.

The convex hull of the incidence vectors of clique partitionings (W_1, \dots, W_p) of G is called the clique partitioning polytope [43] and is denoted by $P_K^p(G)$. By the above definition, $P_K^p(G) \subseteq P_C^{k=p}(G)$. In particular, $P_K^p(K_n) = P_C^{k=p}(K_n)$.

The free cut polytope $P_F(G)$ is the convex hull of free partitionings of G [30]. Hence, $P_C^k(G) \subseteq P_F(G)$, $\forall k = 2, 3, \dots, n$.

The boolean quadratic polytope $BQ(G)$ is the convex hull of boolean vectors $(z, y) \in \mathbb{B}^{|V|+|E|}$ such that $y_{ij} = z_i z_j$ [83]. $BQ(G)$ is actually the image of $P_C(G + v)$ under a bijective linear transformation [89]. Thus, $BQ(G) \equiv P_C(G + v)$.

3.2 Cut Polytope

The cut polytope $P_C(G)$ is investigated in detail in this section. It is the most studied partitioning related polyhedra since early 1980s. Works on cut polytope has begun with the appearance of the max-cut problem: Given a graph $G = (V, E)$ and a weight function $w: E \rightarrow \mathbb{Z}$, the problem of finding a cut $\delta(C)$ such that its weight $w(\delta(C)) = \sum_{e \in \delta(C)} w(e)$ is as large as possible. Yannakakis [98] showed that this problem is \mathcal{NP} -Hard even for graphs having degree at most three. On the other hand, Hadlock [49] proved that the max-cut problem can be solved polynomially for planar graphs. It is also polynomial for weakly bipartite graphs [46], and for graphs without odd long cycles [45].

Wagner developed a decomposition schema [95] for graphs not contractible to the complete graph with 5 nodes K_5 which can be expressed by planar graphs and a special cycle with eight vertices and three chords V_8 . Barahona [6] gave a polynomial algorithm that uses this decomposition to reduce the max-cut problem for graphs not contractible to K_5 to a sequence max-cut problems in planar graphs and in V_8 . His procedure is quite similar to that of Cornuéjols *et al.*'s [27] which is designed for solving Travelling Salesman Problem in graphs with 3-edge cutsets. Barahona also proved that the max-cut problem in graphs with no subgraph contractible to K_6 is \mathcal{NP} -Hard.

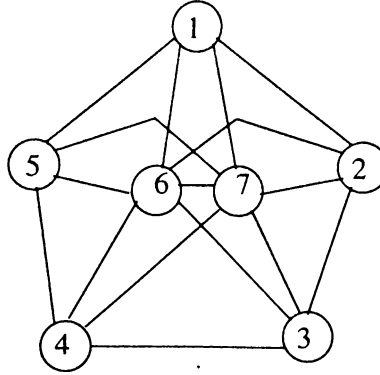
Barahona, Grötschel and Mahjoub [44] showed that $P_C(G)$ is full dimensional. We know from polyhedral theory that if $P \subseteq \mathbb{R}^m$ is a full dimensional ($\dim P = m$) polyhedron, a linear system $Ax \leq b$ that defines P is minimal if and only if the inequalities define all the facets of P .

Barahona, Grötschel and Mahjoub stated that some of the facet defining inequalities of $P_B(G)$ are also facet defining inequalities of $P_C(G)$:

Facet 1 (K_k inequalities) *Let $K_k = (W, F)$ be a complete subgraph of order k of G . Then the K_k inequality*

$$x(F) = \sum_{e \in F} x_e \leq \left\lfloor \frac{k}{2} \right\rfloor \left\lfloor \frac{k}{2} \right\rfloor$$

is valid for $P_C(G)$; this equality defines a facet if and only if k is odd.



$$x_{12} + x_{23} + x_{34} + x_{45} + x_{15} + x_{67} + x_{16} + x_{26} + x_{36} + x_{46} + x_{56} + x_{17} + x_{27} + x_{37} + x_{47} + x_{57} \leq 10.$$

Figure 3.1: A bicycle 5-wheel and its inequality.

Facet 2 (Bicycle $(2k+1)$ -wheel inequalities) Let $G=(V, E)$ be a graph and let (W, F) be a bicycle $(2k+1)$ -wheel, $k \geq 1$, contained in G . Then the inequality

$$x(F) \leq 2(2k + 1)$$

defines a facet of $P_C(G)$.

An example is illustrated in Figure 3.1.

Facet 3 (Generalized K_k inequalities) Let $H=(W, F)$ be a complete subgraph of order q where $W=\{1, 2, \dots, q\}$. Let positive integers t_i ($1 \leq i \leq q$) satisfy $\sum_{i=1}^q t_i = 2k + 1$, $k \geq 3$ and $\sum_{t_i > 1} t_i \leq k - 1$. Set

$$a_{ij} := \begin{cases} t_i t_j, & 1 \leq i < j \leq q, \\ 0, & \{i, j\} \notin W. \end{cases}$$

Then

$$a^T x \leq \alpha := k(k + 1)$$

defines a facet of $P_C(G)$.

Barahona and Mahjoub [8] gave complete polyhedral description of the cut polytope $P_C(G)$ for the graphs not contractible to K_5 . They reported facets associated with edges and cycles:

Facet 4 (Unit hypercube equalities) *Given a graph $G=(V, E)$, an incidence vector x must verify the inequalities*

$$0 \leq x(e) \leq 1, \forall e \in E.$$

The unit hypercube inequalities define a facet of $P_C(G)$ if and only if e does not belong to a triangle.

Facet 5 (Odd cycle inequalities) *Let $C \subseteq E$ be a cycle and $F \subseteq C$, $|F|$ odd, then*

$$x(F) - x(C \setminus F) \leq 1$$

defines a facet of $P_C(G)$ if and only if C is a chordless cycle.

Furthermore, Barahona and Mahjoub proved that the facets 4 and 5 uniquely define the cut polytope $P_C(G)$ for graphs not contractible to K_5 .

Grötschel, Lovász and Schrijver [77] reported that one can use the ellipsoid method to optimize a linear function over a polyhedron P in a polynomial time, if there is a polynomial algorithm to solve the so called *separation* problem: Given a vector x , prove that $x \in P$ or else find an hyperplane that separates x from P . The knowledge of an efficient method to solve separation problem gives an answer to some theoretical and practical questions. It proves that a problem is polynomially solvable, and it permits the design of linear programming based cutting plane algorithms.

Barahona and Mahjoub stated that the separation problem can be solved in polynomial time by a polynomial algorithm designed to solve max-cut problem for graphs not contractible to K_5 . The separation problem is trivial for unit hypercube inequalities. For odd cycle inequalities, Grötschel, Barahona and Mahjoub [44] reported a procedure to the separation problem in polynomial time. For bicycle wheel inequalities, Gerards [40] showed that the separation problem can be reduced to a sequence of shortest path calculations. Moreover, it is possible to check all K_k inequalities in polynomial time for a fixed k . However, it is not known whether there is a polynomial algorithm to solve the separation problem for all K_k inequalities.

Barahona *et al.* [7] implemented a standard cutting plane algorithm based on the simplex method. They start with a very coarse LP relaxation of the max-cut problem and use the simplex method to solve it. If the optimum solution is the incidence vector of a cut, it is done. Otherwise, a separation phase is entered to find inequalities violated by the current optimum solution. If such inequalities are found, they are added and the process is repeated. If not, branch and bound is applied. Barahona *et al.* [7] tested their algorithm for special graphs resulted from statistical physics and reported their computational experience.

Boros and Hammer [15, 17], and Deza and Laurent [33, 34, 31, 32] presented a class of valid inequalities as well as a class of facets for the cut polytope of the complete graph $P_C(K_n)$. $P_C(K_n)$ gives some insight for general cut polytopes $P_C(G)$. For instance, every facet defining inequality of $P_C(K_n)$ also defines a facet of $P_C(G)$ if G is any graph containing K_n [90]. It was proven in [8] that the cut polytope has the following property; namely, a description of the facets that contain any particular extreme point gives the description of the whole polytope. For this reason, it is enough to study the facets that contain the origin, i.e., the facets of the cut cone $C_c(G)$.

The first known class of valid inequalities of the cut cone $C_c(G)$ is *spanning tree inequalities*.

Facet 6 (Spanning Tree inequalities) *Let $G=(V, E)$ be a graph of order n . Let $l_j \geq 0, j = 1, \dots, p$ and $l_j < 0, j = p + 1, \dots, n - 1$ be given integers. Let T be a spanning tree on the vertices $\{1, \dots, p\}$. The following inequality is valid for $P_C(G)$:*

$$\sum_{1 \leq i < j \leq n} l_i l_j x_{ij} - \binom{k+1}{2} \left(\sum_{i=1}^p (2 - d_T(i)) x_{in} + \sum_{ij \in E(T)} x_{ij} \right) \leq 0$$

where $l_n = (2k + 1) - \sum_{i=1}^{n-1} l_i$ and $d_T(i)$ denotes the degree of vertex i in T .

Furthermore, if $l_{p+1} = \dots = l_{n-1} = -1, l_j \geq k, j = 1, \dots, p, \sum_{j=1}^p l_j \leq n + k - p$, and if T is such that there are disjoint subsets C_i of $\{1, 2, \dots, p\}$ with $\sum_{j \in C_i} l_j \geq k + 1, i = 1, 2$, which induce connected subgraphs of T , then the above inequality is facet defining for $P_C(G)$.

In particular, let $r=2$, $n \geq 8$, $p=\lfloor \frac{n+1}{3} \rfloor$ for $k=1$; and $r=k+1$, $n \geq 3k+8$, $p=\lfloor \frac{n+k}{k+2} \rfloor$ for $k \geq 2$ and let $l_{p+1} = \dots = l_{n-1} = -1$. Then for any spanning tree T on vertices $\{1, \dots, p\}$ the above inequality is facet defining for $P_C(G)$.

Since the number of trees on p points is p^{p-2} , the above facet provides at least $(\frac{n}{3})^{\frac{n}{3}}$ facets for $P_C(G)$, any fixed n . Remark also that the separation problem for this class of valid inequalities for fixed n, k, p and l is equivalent with a minimum spanning tree problem.

The second but more general class of valid inequalities for the cut cone $C_c(G)$ is *hypermetric inequalities*. They are introduced first by Deza in 1960 [29]. This class generalizes generalized K_k inequalities and unit hypercube inequalities. For small values of $n \leq 6$ hypermetric facets are sufficient for $C_C(K_n)$.

Facet 7 (Hypermetric inequalities) Let $b = (b_1, \dots, b_n)$ where b_i 's are integers satisfying $\sum_{i=1}^n b_i = 1$. The inequality

$$\sum_{i=1}^{n-1} \sum_{j=i+1}^n b_i b_j x_{ij} \leq 0$$

is valid for $C_C(K_n)$, it is called the hypermetric inequality defined by b and denoted by $Hyp_n(b)$. Some known hypermetric facets are

1. $Hyp_3(1, 1, -1)$ (triangle facet),
2. $Hyp_5(1, 1, 1, -1, -1)$ (pentagonal facet),
3. $Hyp_6(2, 1, 1, -1, -1, -1)$,
4. $Hyp_7(1, 1, 1, 1, -1, -1, -1)$,
5. $Hyp_7(3, 1, 1, -1, -1, -1, -1)$,
6. $Hyp_8(3, 2, 2, -1, -1, -1, -1, -2)$,
7. $Hyp_9(2, 2, 1, 1, -1, -1, -1, -1, -1)$.

Furthermore, if $Hyp_n(b)$ is facet inducing then $Hyp_{n+1}(b, 0)$ is also facet inducing.

The third class of valid inequalities for the cut cone $C_c(G)$ is *cycle inequalities*. This class generalizes odd cycle inequalities.

Facet 8 (Cycle inequalities) Let $b = (b_1, \dots, b_n)$ where b_i 's are integers satisfying $\sum_{i=1}^n b_i = 3$. The set $B_+ = \{i \in \mathbf{N} : b_i > 0\}$ is called positive support of b . Set $f = |B_+|$ and $B_+ = \{i_1, \dots, i_f\}$ with $1 \leq i_1 < \dots < i_f \leq n$ and let C be a cycle with node set B_+ . The inequality

$$\sum_{i=1}^{n-1} \sum_{j=i+1}^n b_i b_j x_{ij} - \sum_{(i,j) \in C} x_{ij} \leq 0$$

is called the cycle inequality and denoted by $Cyc_n(b)$. The following cycle inequalities are all facet inducing:

1. $Cyc_7(3, 2, 2, -1, -1, -1, -1)$,
2. $Cyc_7(2, 2, 1, 1, -1, -1, -1)$,
3. $Cyc_7(1, 1, 1, 1, 1, -1, -1)$,
4. $Cyc_8(2, 2, 2, 1, -1, -1, -1, -1)$,
5. $Cyc_8(2, 1, 1, 1, 1, -1, -1, -1)$,
6. $Cyc_8(3, 3, 2, -1, -1, -1, -1, -1)$,
7. $Cyc_8(3, 2, 1, 1, 1, -1, -1, -1)$,
8. $Cyc_9(1, 1, 1, 1, 1, 1, -1, -1, -1)$.

Again, if $Cyc_n(b)$ is facet inducing then $Cyc_{n+1}(b, 0)$ is also facet inducing.

Third class is pure clique-web inequalities which generalizes bicycle wheel inequalities:

Facet 9 (Pure clique-web inequalities) The pure clique-web inequality CW_n^k with parameters n, k, p, q satisfying $n = p + q$, $p - q = 2k + 1$, $q \geq 2$ is the inequality

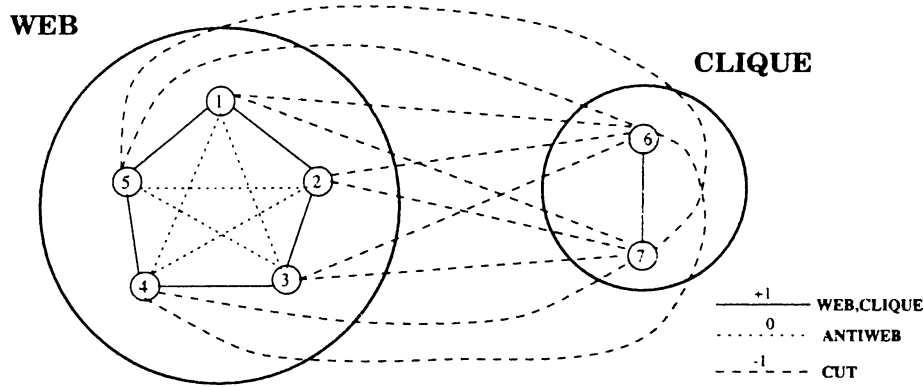
$$CW_n^k \cdot x = \sum_{i=1}^{n-1} \sum_{j=i+1}^n b_i b_j x_{ij} - \sum_{(i,j) \in AW_p^k} x_{ij} \leq 0$$

with $b = (1, \dots, 1, -1, \dots, -1)$ whose first p coefficients are $+1$ and last $q = n - p$ coefficients are -1 .

The terminology “clique-web” inequality is chosen since the above inequality can also be written as

$$\sum_{(ij) \in W_p^k} x_{ij} + \sum_{i=p+1}^{n-1} \sum_{j=i+1}^n x_{ij} - \sum_{i=1}^p \sum_{j=p+1}^n x_{ij} \leq 0.$$

Hence, there is a web on the first p nodes (the nodes for which $b_i = +1$) and a clique on the remaining $q = n - p$ nodes (the nodes for which $b_i = -1$). The edges in the web and in the clique have the coefficient $+1$, the edges between the clique and the web take the coefficient -1 , and the edges in the antiweb have zero coefficients. An example is illustrated in Figure 3.2.



$$(x_{12} + x_{23} + x_{34} + x_{45} + x_{15}) + (x_{67}) - (x_{16} + x_{26} + x_{36} + x_{46} + x_{56} + x_{17} + x_{27} + x_{37} + x_{47} + x_{57}) \leq 0.$$

Figure 3.2: A CW_7^1 and its inequality.

Finally, there is a superset of valid inequalities for $C_C(K_n)$ which generalizes all, called *generalized clique-web inequalities*:

Facet 10 (General clique-web inequalities) Let $b = (b_1, \dots, b_n)$ be integers whose sum is $b_1 + \dots + b_n = 2k + 1$ ($k \geq 0$), and let p (q) denote the number of positive (nonpositive) b_i 's; so $n = p + q$. Let B_+ be positive support of b , so $|B_+| = p$. Without loss of generality we may assume that $B_+ = \{1, \dots, p\}$.

Consider the collapsed antiweb $AW_p^k(b_1, \dots, b_p)$ on the node set B_+ . Then the general clique-web is the inequality

$$CW_n^k(b) \cdot x = \sum_{i=1}^{n-1} \sum_{j=i+1}^n b_i b_j x_{ij} - \sum_{(i,j) \in AW_p^k(b)} x_{ij} \leq 0.$$

Chapter 4

BOOLEAN R-ATIC POLYTOPE

Boolean R-atic Polytope is the convex hull of the incidence vectors of vertex induced subhypergraphs in a complete r-uniform hypergraph. Each hypergraph can be transformed into an r-uniform hypergraph in such a way that the endpoints of all hyperedges are increased to (r) terminal nodes by adding at most (r-1) pseudo vertices. The vertex induced subhypergraphs are invariant under this transformation if all of the pseudo nodes are kept among the selected vertices in the r-uniform hypergraph. In this chapter, results on the Boolean R-atic Polytope are reported.

4.1 Preliminaries

The problem of picking the best portion of a hypergraph is the selection of a subset S of the node set V such that the total weight of the nodes and the hyperedges in the selection is the best. Combinatorially speaking, the problem is

$$\max_{S \subseteq V} c(S) + d(\gamma(S)).$$

An instance of the problem on an example hypergraph is illustrated in Figure 4.1. Vertices 3, 4, 5 and 6 form the set S . Hyperedges 6 and 7 lies inside $\gamma(S)$.

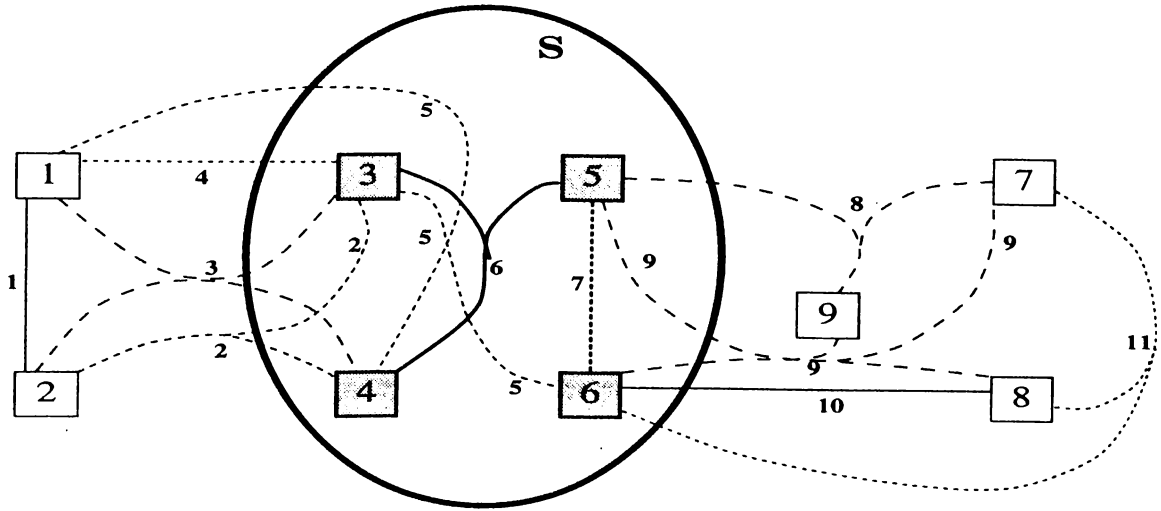


Figure 4.1: An instance of the problem on an example hypergraph representation.

In this case, the objective function value is equal to $c_3 + c_4 + c_5 + c_6 + d_6 + d_7$. If we add vertex 1 into the selected set, hyperedges 4 and 5 will also be selected. In this case, we can get an extra $d_4 + d_5 + c_1$ units in the objective function.

The two set of binary variables defined for nodes, x_i , $i = 1, \dots, n$, and for hyperedges, y_j , $j = 1, \dots, m$ indicate whether the associated elements are in the selection or not. Then, the problem can be formulated as:

$$\text{Maximize } \sum_{i=1}^n c_i x_i + \sum_{j=1}^m d_j y_j$$

subject to:

$$\begin{aligned} y_j &= x_{i_1} \cdot x_{i_2} \cdot \dots \cdot x_{i_{k_j}}, & j &= 1, \dots, m & e_j &= \{i_1, i_2, \dots, i_{k_j}\} \\ x_i &= 0 \text{ or } 1, & i &= 1, \dots, n \\ y_j &= 0 \text{ or } 1, & j &= 1, \dots, m \end{aligned}$$

The constraint $y_j = x_{i_1} \cdot x_{i_2} \cdot \dots \cdot x_{i_{k_j}}$ prevents the hyperedge e_j from being selected unless all of its end-points are chosen. These constraints are nonlinear. If $k_j = 2$, $\forall j = 1, \dots, m$, that is if the structure constitutes a graph, the constraints are quadratic. In this case, the problem is an unconstrained quadratic zero-one problem which is \mathcal{NP} -Hard in general. Therefore, our problem is one

of the hard combinatorial problems. A real life example arises from manufacturing. The problem of identifying the best manufacturing cell is the best selection of machine types that are dedicated to the manufacturing of a specific range of parts whose fabrication operations can be made completely using the selected machines. In this case, machines constitute the node set whereas hyperedges represent parts. The objective function is then to maximize profit over a specific period of time which is the revenue obtained from the parts that can be produced within the cell minus the cost of machines that form the cell.

The constraints $y_j = \prod_{i=1}^{k_j} x_{i_l}$ can be linearized using the following inequalities:

$$\begin{aligned} y_j &\leq x_{i_l} \quad l = 1, \dots, k_j \\ x_{i_1} + x_{i_2} + \dots + x_{i_{k_j}} &\leq (k_j - 1) + y_j \\ y_j &\geq 0 \\ x_{i_l} &\leq 1, \quad l = 1, \dots, k_j \\ y_j &\text{ integer; and } x_{i_l} \text{ integer, } l = 1, \dots, k_j \end{aligned}$$

By the inequalities $y_j \leq x_{i_l}$, the hyperedge is not contained in the subhypergraph if any of the end-points is missing. By the next inequality, if all of the end-points are in the selected set, the hyperedge should lie inside the selection. On the other hand, if a hyperedge is in the subhypergraph, all of its end-points should also be in the selection. Moreover, if the hyperedge is not in the subhypergraph, then at most $(k_j - 1)$ end-points can be in the selected set. The inequalities $x_{i_l} \leq 1$ are implied by the other inequalities when $k_j = 2$.

If $r(H) - \rho(H) + 1$ pseudo nodes are added to the set of vertices with zero weights, and each hyperedge is extended to have exactly $r = r(H)$ vertices, the original hypergraph H is transformed into an r -uniform hypergraph. Let $|e_j| = l < r$, then this hyperedge is extended to have r end-points such that $\epsilon_j = e_j \cup \{l + 1, \dots, r\}$ where $l + 1, \dots, r$ are new dummy vertices.

This transformation is illustrated in Figure 4.2. In our example, $\rho = 2$, $r = 5$ (part 9), hence we add 3 pseudo nodes. Since hyperedge 6 is incident to only three nodes, it is connected to the last two dummy vertices, 4 and 5. If we extend

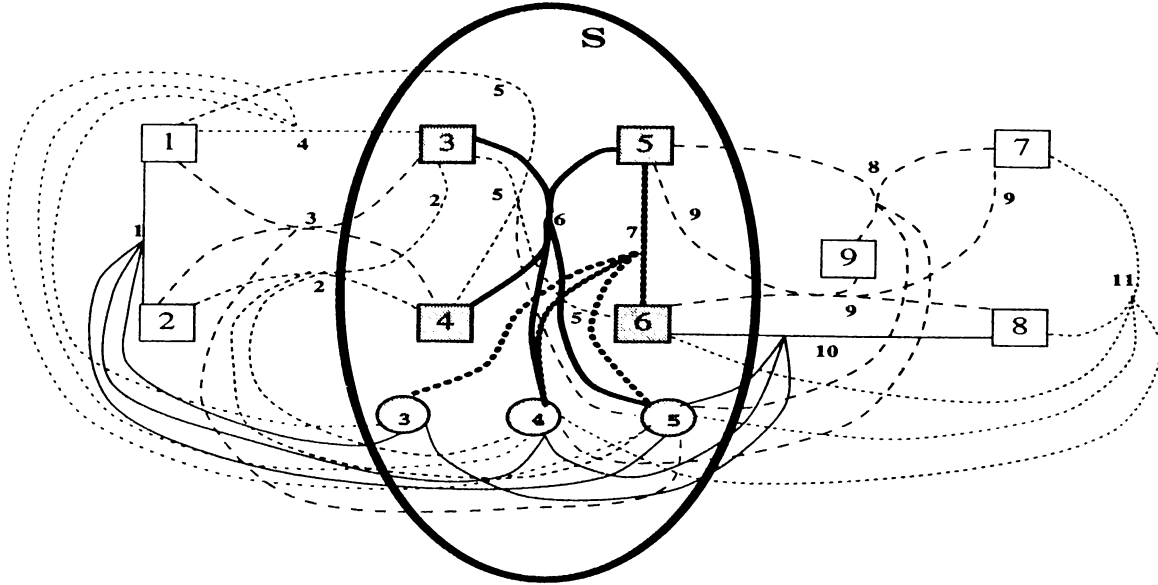


Figure 4.2: The same instance of the problem on the transformed 5-uniform hypergraph.

every hyperedge similarly, we will obtain the 5-uniform hypergraph as given in the Figure 4.2.

The pseudo nodes are always in the selected node set because of zero weights. Let S be an arbitrary node subset of the node set V . Let $T = S \cup \{\rho(H) + 1, \dots, r(H)\}$ be the corresponding node set. Then, $\gamma(S) = \gamma(T)$. Thus, the solution is invariant under this transformation.

If we add all of the missing hyperedges with zero weights, we will have complete r -uniform hypergraph, H_n^r .

4.2 Definition and Dimension

The problem in a complete r -uniform hypergraph of order n , H_n^r , is restated as follows:

$$\begin{aligned} & \text{Maximize } \sum_{i=1}^n c_i x_i + \sum_{j=1}^{\binom{n}{r}} d_j y_j \\ & \text{subject to:} \\ & \quad z = (x, y) \in RP(H_n^r) \end{aligned}$$

where the Boolean R-atic Polytope

$$\begin{aligned} RP(H_n^r) = \{ (x, y) \in \mathbb{B}^{n+\binom{n}{r}} : & y_j \leq x_i \quad \forall i \in e_j, \forall j = 1, \dots, \binom{n}{r} \\ & \sum_{i \in e_j} x_i - y_j \leq r - 1 \quad \forall j = 1, \dots, \binom{n}{r} \\ & y_j \geq 0, \quad \forall j = 1, \dots, \binom{n}{r} \\ & x_i \leq 1, \quad \forall i = 1, \dots, n \} \\ RP(H_n^r) = \text{conv} \{ (\mathcal{X}(S), \mathcal{X}(\gamma(S))) : S \subseteq V \}. \end{aligned}$$

As a consequence of the last definition, $RP(H_n^r)$ is a polytope and it has 2^n vertices. Moreover, $RP(H_n^r)$ is full dimensional. Let $u_i \in \mathbb{B}^{n+\binom{n}{r}}$ be a canonical unit vector whose first n components are $\mathcal{X}(\{i\})$ and others are zero. Similarly, let $v_j \in \mathbb{B}^{n+\binom{n}{r}}$ be a vector whose first n components are $\mathcal{X}(e_j)$ and last $\binom{n}{r}$ components form a canonical unit vector $\mathcal{X}(\gamma(e_j) = e_j)$. The vectors u_i 's and v_j 's satisfy the constraints. In fact, they are vertices of the polytope. Let A be a matrix whose rows are u_i 's and v_j 's.

$$A = \left[\begin{array}{c|c} I & \mathbf{0} \\ \hline B & I \end{array} \right]$$

Since A has full row rank, $\dim(RP(H_n^r)) = n + \binom{n}{r}$.

If $r = 2$, the polytope $RP(H_n^{r=2})$ is the Boolean Quadratic Polytope defined by Padberg [83]. The following theorems in the next section are generalization of his results for $RP(H_n^r)$.

4.3 Facet Defining Inequalities

Theorem 4.1 $y_j \geq 0$ defines a facet for $RP(H_n^r)$.

Proof:

Let $F = \{(x, y) \in RP : y_j = 0\}$.

Let $b(x, y) \leq b_0$ be facet defining such that $\forall (x, y) \in F, b(x, y) = b_0$.

- i. $0 \in F \Rightarrow b_0 = 0 = b_0$.
- ii. $u_i \in F, \forall i = 1, \dots, n \Rightarrow bu_i = b_i = b_0 = 0, \forall i = 1, \dots, n$.
- iii. $v_k \in F, e_k = \{i_1, \dots, i_r\} \neq e_j \Rightarrow bv_k = b_{i_1} + \dots + b_{i_r} + b_k = b_k = b_0 = 0, \forall k \neq j$.

Then $b(x, y) \leq b_0$ reduces to $b_j y_j \leq 0$.

$b \neq 0$, otherwise $b(x, y) \leq b_0$ is not facet defining since $RP(H_n^r)$ itself satisfies the facet defining inequality which yields that the facet is not a proper face.

Hence, $b_j < 0 \Rightarrow y_j \geq 0$ is facet defining. \square

Theorem 4.2 $x_i \leq 1$ defines a facet for $RP(H_n^r)$, when $r > 2$.

Proof:

Let $F = \{(x, y) \in RP : x_i = 1\}$.

Let $b(x, y) \leq b_0$ be facet defining such that $\forall (x, y) \in F, b(x, y) = b_0$.

- i. $u_i \in F \Rightarrow bu_i = b_i = b_0$.
- ii. $z = u_i \uplus u_j \in F, \forall j \neq i \Rightarrow bz = b_i + b_j = b_0 (r > 2!) \Rightarrow b_j = 0, \forall j \neq i$
- iii. $v_k \in F, e_k = \{i, i_2, \dots, i_r\} \Rightarrow bv_k = b_i + b_{i_2} + \dots + b_{i_r} + b_k = b_i + b_k = b_0 \Rightarrow b_k = 0, \forall k \ni i \in e_k$.
- iv. $z = u_i \uplus v_k \in F, i \notin e_k = \{i_1, \dots, i_r\} \Rightarrow bz = b_i + b_{i_1} + \dots + b_{i_r} + b_k + bl_1 + \dots + bl_r = b_0$, where $e_{l_k} = \{i, i_1, \dots, i_{k-1}, i_{k+1}, \dots, i_r\}$. Then, $bz = b_i + b_k = b_0$.
 $\Rightarrow b_k = 0, \forall k \ni i \notin e_k$.

Then $b(x, y) \leq b_0$ reduces to $b_i x_i \leq b_0 \Rightarrow b_i x_i \leq b_i$.

$b \neq 0$, if otherwise $b(x, y) \leq b_0$ is not facet defining. $0 \in RP(H_n^r) \Rightarrow 0 \leq b_i$.

Hence, $b_i > 0 \Rightarrow x_i \leq 1$ is facet defining. \square

Theorem 4.3 $y_j \leq x_i, i \in e_j$ defines a facet for $RP(H_n^r)$, only if $r = 2$.

Proof:

Let $F = \{(x, y) \in RP : y_j = x_i, i \in e_j\}$.

Let $b(x, y) \leq b_0$ be facet defining such that $\forall (x, y) \in F, b(x, y) = b_0$.

i. $0 \in F \Rightarrow b0 = 0 = b_0$.

ii. $u_k \in F, k \neq i \Rightarrow bu_k = b_k = b_0 = 0, \forall k \neq i$.

iii. $v_j \in F, e_j = \{i, i_2, \dots, i_r\} \Rightarrow bv_j = b_i + b_{i_2} + \dots + b_{i_r} + b_j = b_i + b_j = b_0 = 0$
 $\Rightarrow b_i = -b_j$.

iv. $v_k \in F, e_k = \{i_1, \dots, i_r\}, i \notin e_k \Rightarrow bv_k = b_{i_1} + \dots + b_{i_r} + b_k = b_k = b_0 = 0$,
 $\Rightarrow b_k = 0, \forall k \ni i \notin e_k$.

v. $z = u_k \uplus v_j \in F, k \notin e_j \Rightarrow bz = b_i + b_{i_2} + \dots + b_{i_{r-1}} + b_{i_r} + b_k + b_j + b_{i_1} +$
 $\dots + b_{i_r} = b_0$

where,

$$e_{l_1} = \{i, k, i_2, \dots, i_{r-2}, i_{r-1}\}$$

$$e_{l_2} = \{i, k, i_2, \dots, i_{r-2}, i_r\}$$

\vdots

$$e_{l_{r-1}} = \{i, k, i_3, \dots, i_{r-1}\}$$

$$e_{l_r} = \{k, i_2, \dots, i_r\}$$

$$\Rightarrow bz = (b_i + b_j) + b_{l_1} + \dots + b_{l_{r-1}} = b_{l_1} + \dots + b_{l_{r-1}} = 0 \quad (\diamond)$$

e.g. $r = 3, i = 1, e_j = \{1, 2, 3\}, k = 4; (\diamond) b_1 + b_{123} + b_{124} + b_{134} =$
 $b_{124} + b_{134} = 0;$

If $r = 2$, then $b(x, y) = b_i x_i + b_j y_j = b_i x_i - b_i y_j \leq 0$.

$b_i \neq 0$, if otherwise $b = 0 \Rightarrow b(x, y) \leq b_0$ is not facet defining.

Hence, $y_j \leq x_i, i \in e_j$ is facet defining for $r=2$. \square

Theorem 4.4 $\sum_{i \in e_j} x_i \leq (r-1) + y_j$, $e_j = \{i_1, \dots, i_r\}$ is facet defining for $RP(H_n^r)$, when $r = 2, 3$.

Proof:

Let $F = \{(x, y) \in RP : \sum_{i \in e_j} x_i = (r-1) + y_j\}$.

Let $b(x, y) \leq b_0$ be facet defining such that $\forall (x, y) \in F$, $b(x, y) = b_0$.

- i. $z = \uplus_{k=1}^{r-1} u_{i_k} \in F \Rightarrow bz = b_{i_1} + \dots + b_{i_{r-1}} = b_0$.
- ii. $v_j \in F \Rightarrow bv_j = b_{i_1} + \dots + b_{i_{r-1}} + b_{i_r} + b_j = b_0$.
 $\rightsquigarrow b_{i_k} = -b_j, \forall k = 1, \dots, r$ and $b_0 = -(r-1)b_j$.
- iii. $z = (\uplus_{k=1}^{r-1} u_{i_k}) \uplus u_l \in F, l \notin e_j \Rightarrow bz = b_{i_1} + \dots + b_{i_{r-1}} + b_l + b_q = b_0$,
 where $e_q = \{l, i_1, \dots, i_{r-1}\}$
 $\Rightarrow b_l = -b_q, (\heartsuit) \forall l \notin e_j, |e_q \cap e_j| = r-1, l \in e_q$.
- iv. $z = v_j \uplus u_l \in F, l \notin e_j \Rightarrow bz = (b_{i_1} + b_{i_2} + \dots + b_{i_{r-1}}) + (b_{i_r} + b_j) + b_l + b_{q_1} + \dots + b_{q_r} = b_0 + 0 + b_{q_1} + \dots + b_{q_r} = b_0$
 where,
 $e_{q_1} = \{l, i_1, i_2, \dots, i_{r-2}, i_{r-1}\}$
 $e_{q_2} = \{l, i_1, i_2, \dots, i_{r-2}, i_r\}$
 \vdots
 $e_{q_r} = \{l, i_2, i_3, \dots, i_r\}$
 $\Rightarrow bz = b_l + b_{q_1} + \dots + b_{q_r} = 0$ (\diamond) $\Rightarrow b_l = 0, \forall l \notin e_j$ and $b_{q_k} = 0, \forall k = 1, \dots, r$.
 e.g. $r = 2, e_j = \{1, 2\}, l = 3$;
 $(\heartsuit) b_3 = -b_{13}$
 $(\heartsuit) b_3 = -b_{23} \quad \rightsquigarrow b_3 = b_{13} = b_{23} = 0$
 $(\diamond) b_3 + b_{13} + b_{23} = 0$
 e.g. $r = 3, i = 1, e_j = \{1, 2, 3\}, l = 4$;
 $(\heartsuit) b_4 = -b_{124}$
 $(\heartsuit) b_4 = -b_{134} \quad \rightsquigarrow b_4 = b_{124} = b_{134} = b_{234} = 0$
 $(\heartsuit) b_4 = -b_{234}$
 $(\diamond) b_4 + b_{124} + b_{134} + b_{234} = 0$

$$\begin{aligned}
 \text{v. } z &= (\uplus_{k=1}^{r-1} u_{i_k}) \uplus u_{l_1} \uplus u_{l_2}, \in F \quad l_1, l_2 \notin e_j \\
 &\Rightarrow bz = (b_{i_1} + \cdots + b_{i_{r-1}}) + b_{l_1} + b_{l_2} + b_{q_1} + b_{q_2} + b_{w_1} + \cdots + b_{w_{r-1}} = \\
 &b_0 + b_{l_1} + b_{l_2} + b_{q_1} + b_{q_2} + b_{w_1} + \cdots + b_{w_{r-1}} = b_0,
 \end{aligned}$$

where

$$e_{q_1} = \{l_1, i_1, \dots, i_{r-1}\}$$

$$e_{q_2} = \{l_2, i_1, \dots, i_{r-1}\}; \text{ and}$$

$$e_{w_1} = \{l_1, l_2, i_2, \dots, i_{r-2}, i_{r-1}\},$$

$$e_{w_2} = \{l_1, l_2, i_1, i_3, \dots, i_{r-2}, i_{r-1}\},$$

\vdots

$$e_{w_{r-1}} = \{l_1, l_2, i_1, \dots, i_{r-3}, i_{r-2}\}$$

$$\begin{aligned}
 &\Rightarrow b_{w_1} + \cdots + b_{w_{r-1}} = 0, (\spadesuit) \forall l_1, l_2 \notin e_j, |e_{w_k} \cap e_j| = r - 2, l_1, l_2 \in e_{w_k}, \\
 &k = 1, \dots, r - 1 \text{ for each choice of } \{i_k\}_{k=1}^{r-1}.
 \end{aligned}$$

$$\text{e.g. } r = 3, e_j = \{1, 2, 3\}, l_1 = 4, l_2 = 5;$$

$$(\spadesuit) b_{1.3} = -b_{245}$$

$$(\spadesuit) b_{1.3} = -b_{345} \quad \rightsquigarrow b_{1.3} = b_{245} = b_{345} = 0.$$

$$(\spadesuit) b_{245} = -b_{345}$$

$$\text{vi. } z = v_j \uplus v_k \in F \quad e_k = \{l_1, \dots, l_r\} \cap e_j = \emptyset$$

$$\Rightarrow bz = b_{i_1} + \cdots + b_{i_r} + b_{l_1} + \cdots + b_{l_r} + b_j + b_l + \sum_{k \in \delta(e_j; e_l)} b_k = b_0$$

$$\Rightarrow b_l + \sum_{k \in \delta(e_j; e_l)} b_k = 0 \quad (\clubsuit).$$

$$\text{e.g. } r = 3, e_j = \{1, 2, 3\}, e_l = \{4, 5, 6\};$$

$$\begin{aligned}
 (\clubsuit) b_{456} + b_{124} + b_{1.3} + b_{126} + b_{134} + b_{1.3} + b_{136} + b_{234} + b_{235} + \\
 b_{236} + b_{1.3} + b_{245} + b_{345} + b_{146} + b_{246} + b_{346} + b_{156} + b_{256} + b_{356} = 0 \\
 e_j = \emptyset. \quad \rightsquigarrow b_l = 0, \forall e_l \cap
 \end{aligned}$$

If $r = 2, 3$ then $b(x, y) = (\sum_{i \in e_j} b_i x_i) + b_j y_j = b_0$

$$\Rightarrow b(x, y) = -b_j (\sum_{i \in e_j} x_i) + b_j y_j = -(r-1)b_j.$$

$b_i = -b_j \neq 0$, if otherwise $b = 0 \Rightarrow b(x, y) \leq b_0$ is not facet defining.

$$\text{Since } 0 \in P, \Rightarrow b_0 = 0 \leq b_0 = -(r-1)b_j \Rightarrow -b_j \geq 0.$$

Hence, $\sum_{i \in e_j} x_i \leq (r-1) + y_j$, $e_j = \{i_1, \dots, i_r\}$ is facet defining for $r=2,3$. \square

4.4 Clique and Cut Inequalities

A wide range of valid inequalities namely *clique* and *cut* inequalities are analyzed next. Let us focus on the special case $n = r + 1$ to gain some insight. The following fractional points satisfy the previous inequalities that define $RP(H_n^r)$:

$$x_i = \frac{r-1}{r}, i = 1, \dots, r+1 \quad y_j = 0 \quad j = 1, \dots, r+1$$

$$x_i = \frac{r-1}{r}, i = 1, \dots, r+1 \quad y_k = 0 \quad y_j = \frac{r-1}{r} \quad j = 2, \dots, r+1; j \neq k.$$

The first point can be cut off by means of the following inequality:

$$\sum_i x_i - \sum_j y_j \leq r-1$$

One can generalize this inequality for any n as follows:

Theorem 4.5 *Let $S \subseteq V$. The clique inequality*

$$\sum_{i \in S} x_i - \sum_{e_j \in \gamma(S)} y_j \leq r-1$$

is valid for $RP(H_n^r)$.

Proof:

For $n = |S| \leq r-1$, it is trivial. For $n = |S| = r$, $\sum_{i \in S} x_i - y_e = r-1$. For $n = |S| \geq r+1$, $|\gamma(S)| \geq |S| \Rightarrow \sum_{i \in S} x_i - \sum_{e \in \gamma(S)} y_e \leq 0 < r-1$ is valid. \square .

The second set of points can be cut off by the following inequality:

$$-x_k + \frac{1}{r-1} \sum_{e_j \in \delta(k)} y_j - y_l \leq 0, \quad k \notin e_l.$$

This inequality can be generalized as

$$-\sum_{i \in S} x_i + \alpha \sum_{e_j \in \delta(S;T)} y_j - \sum_{e_j \in \gamma(S)} y_j - \sum_{e_j \in \gamma(T)} y_j \leq 0, \quad S, T \subseteq V \quad S \cap T = \emptyset.$$

The above set of inequalities are called as cut inequalities. The problem is to find the minimum value for α . Let $s = |S|$, $t = |T|$, $|\{i \in S : x_i = 1\}| = a$, $|\{i \in T : x_i = 1\}| = b$. Then,

$$\alpha_{a,b} = \frac{a + \binom{a}{r} + \binom{b}{r}}{\binom{a+b}{r} - \binom{a}{r} - \binom{b}{r}} \geq \alpha_{min} \quad 1 \leq a \leq s, \quad 1 \leq b \leq t.$$

Let

$$\alpha_{min} = \begin{cases} \alpha_{s,t} & s < r, t < r \\ \alpha_{r+2,t} & s \geq r, t < r; t = 1, r \geq 5 \\ \alpha_{r+1,t} & s \geq r, t < r; t = 1, r \leq 5; t \geq 2 \\ \alpha_{s,r} & s < r, t \geq r \\ \alpha_{r,r} & s = r, t = r \\ \alpha_{r,r+1} & s > r, t > r \end{cases}$$

The next theorem is proved with the help of the following propositions:

Proposition 4.1 *Let $s < r, t < r$. Then*

$$\min_{1 \leq a \leq s, 1 \leq b \leq t} \alpha_{a,b} = \alpha_{s,t}.$$

Proof:

$$\alpha_{a,b} = \frac{a}{\binom{a+b}{r}} \Rightarrow \min_{1 \leq a \leq s, 1 \leq b \leq t} \alpha_{a,b} = \min_{1 \leq a \leq s} \alpha_{a,t}.$$

Let $F(a) = \frac{a}{\binom{t+a}{r}}$. Then $F(a+1) = \frac{a+1}{\binom{t+a+1}{r}}$.

$$F(a) \text{ vs } F(a+1) \Leftrightarrow \frac{a}{a+1} \text{ vs } \frac{\binom{t+a}{r}}{\binom{t+a+1}{r}} = \frac{t+a+1-r}{t+a+1}$$

$$at + a^2 + a \text{ vs } at + a^2 + a(2-r) + (t+1-r) \Rightarrow F(a) \geq F(a+1)$$

$$\leadsto \min_{1 \leq a \leq s} \alpha_{a,t} = \alpha_{s,t}.$$

□

Proposition 4.2 *Let $s \geq r, t < r$. Then*

$$\min_{1 \leq a \leq s, 1 \leq b \leq t} \alpha_{a,b} = \begin{cases} \alpha_{r+2,1}, & r \geq 5, t = 1 \\ \alpha_{r+1,t}, & \text{otherwise} \end{cases}$$

Proof:

$$\alpha_{a,b} = \frac{a + \binom{a}{r}}{\binom{a+b}{r} - \binom{a}{r}}.$$

In order to minimize $\alpha_{a,b}$, one should maximize b . Hence, $b = t$. Let $a \geq r$.

$$F(a) = \alpha_{a,t} = \frac{a + \binom{a}{r}}{\binom{a+t}{r} - \binom{a}{r}} \text{ vs } F(a+1) = \frac{a+1 + \binom{a+1}{r}}{\binom{a+t+1}{r} - \binom{a+1}{r}} = \frac{a + \binom{a}{r} + 1 + \binom{a}{r-1}}{\binom{a+t}{r} - \binom{a}{r} + \binom{a+t}{r-1} - \binom{a}{r-1}}$$

$$\begin{aligned} LHS &= \frac{a + \binom{a}{r}}{1 + \binom{a}{r-1}} = \frac{a - \frac{a-r+1}{r}}{1 + \binom{a}{r-1}} + \frac{a-r+1}{r} \\ RHS &= \frac{\binom{a+t}{r} - \binom{a}{r}}{\binom{a+t}{r-1} - \binom{a}{r-1}} = \frac{\binom{a+t-1}{r-1} + \dots + \binom{a}{r-1}}{\binom{a+t-1}{r-2} + \dots + \binom{a}{r-2}} \geq \frac{a-r+2}{r-1} \end{aligned}$$

$$RHS - LHS \geq \frac{a-r+1}{r(r-1)} + \frac{1}{r-1} - \frac{a - \frac{a-r+1}{r}}{1 + \binom{a}{r-1}} \geq \frac{a-r+1}{r(r-1)} + \frac{1}{r-1} - \frac{a}{\binom{a}{r-1}}$$

$RHS \geq LHS$, $a \geq r+2$.

Let $t = 1$, then

$$\alpha_{r,1} = \frac{r+1}{r}, \alpha_{r+1,1} = \frac{2(r+1)}{\binom{r+2}{r} - (r+1)} = \frac{4}{r}, \alpha_{r+2,1} = \frac{r+2 + \binom{r+2}{r}}{\binom{r+3}{r} - \binom{r+2}{r}} = \frac{3(r+3)}{r(r+1)}$$

$$\min_{a=r, r+1, r+2} \alpha_{r,1} = \begin{cases} \alpha_{r+1,1}, & r \leq 5 \\ \alpha_{r+2,1}, & r \geq 5 \end{cases}$$

$$\text{Let } t \geq 2. \alpha_{r,t} = \frac{r+1}{\binom{r+t}{r} - 1}, \alpha_{r+1,t} = \frac{2(r+1)}{\binom{r+t+1}{r} - (r+1)}, \alpha_{r+2,t} = \frac{r+2 + \binom{r+2}{r}}{\binom{r+t+2}{r} - \binom{r+2}{r}}.$$

$$\alpha_{r,t} \text{ vs } \alpha_{r+1,t} \Leftrightarrow \binom{r+t+1}{r} - (r+1) \text{ vs } 2 \binom{r+t}{r} - 2 \Leftrightarrow (r-t-1) \binom{r+t}{r} \text{ vs } (r-1)(t+1)$$

$$\leadsto \alpha_{r,t} \geq \alpha_{r+1,t}.$$

$$\alpha_{r+1,t} \text{ vs } \alpha_{r+2,t} \Leftrightarrow$$

$$LHS = \left[2(r+1) \frac{r+2+t}{t+2} - \frac{1}{2}(r+2)(r+3) \right] \binom{r+t+1}{r} \text{ vs } RHS = \frac{1}{2}(r+1)^2(r+2)$$

$$LHS \leq \left[2(r+1) \frac{r+3}{4} - \frac{1}{2}(r+2)(r+3) \right] = \frac{1}{2}[(r+1)(r+4) - (r+2)(r+3)] < 0 \leq$$

$$RHS \leadsto \alpha_{r+1,t} \leq \alpha_{r+2,t}. \quad \square$$

Proposition 4.3 *Let $s < r$, $t \geq r$. Then*

$$\min_{1 \leq a \leq s, 1 \leq b \leq t} \alpha_{a,b} = \alpha_{s,r}$$

Proof:

$$\alpha_{a,b} = \frac{a + \binom{b}{r}}{\binom{a+b}{r} - \binom{b}{r}}.$$

Let $b \geq r$ be fixed. Let $F_b(a) = \alpha_{a,b}$, for a given b .

$$\begin{aligned} F_b(a) &= \frac{a + \binom{b}{r}}{\binom{a+b}{r} - \binom{b}{r}} \text{ vs } F_b(a+1) = \frac{a + \binom{b}{r} + 1}{\binom{a+b}{r} - \binom{b}{r} + \binom{a+b}{r-1}} \\ &= \left[a + \binom{b}{r} \right] \binom{a+b}{r-1} \text{ vs } \binom{a+b}{r} - \binom{b}{r} \\ &= \left[\frac{r}{a+b-r+1} \left(a + \binom{b}{r} \right) - 1 \right] \binom{a+b}{r} + \binom{b}{r} \text{ vs } 0 \end{aligned}$$

$$a + b - r + 1 < a + \binom{b}{r} \Rightarrow F_b(a) \geq F_b(a+1) \geq F_b(s).$$

Let a be fixed at s . Let $G(b) = \alpha_{s,b}$.

$$G(b) = \frac{s + \binom{b}{r}}{\binom{s+b}{r} - \binom{b}{r}} \text{ vs } G(b+1) = \frac{s + \binom{b+1}{r}}{\binom{s+b+1}{r} - \binom{b+1}{r}}$$

$$LHS = \frac{s + \binom{b}{r}}{s + \binom{b+1}{r}} \text{ vs } RHS = \frac{\binom{s+b}{r} - \binom{b}{r}}{\binom{s+b+1}{r} - \binom{b+1}{r}}$$

$$LHS = \frac{s \left[1 - \frac{b-r+1}{b+1} \right] + \left[\frac{b-r+1}{b+1} \right] s + \frac{\binom{b+1}{r}}{s + \binom{b+1}{r}} = \frac{s \left[1 - \frac{b-r+1}{b+1} \right] + \frac{b-r+1}{b+1}}{s + \binom{b+1}{r}}$$

$$RHS = \frac{\binom{a+b-1}{r-1} + \dots + \binom{b}{r-1}}{\binom{a+b}{r-1} + \dots + \binom{b+1}{r-1}} \geq \frac{b-r+2}{b+1}$$

$$RHS - LHS \geq \frac{1}{b+1} - \frac{s \frac{r}{b+1}}{s + \binom{b+1}{r}}$$

$$s(r-1) \leq \binom{b+1}{r}, b \geq r+1 \Rightarrow RHS \geq LHS, b \geq r+1 F_s(b) \geq F_s(b+1), b \geq r+1$$

$$\alpha_{s,r-1} = \frac{s}{\binom{s+r-1}{r}} \text{ vs } \alpha_{s,r} = \frac{s+1}{\binom{s+r}{r} - 1} = \frac{s+1}{\binom{s+r-1}{r} + \binom{s+r-1}{r-1} - 1}$$

$$s \binom{s+r-1}{r-1} - s \text{ vs } \binom{s+r-1}{r} \Leftrightarrow \left[s - \frac{s}{r} \right] \binom{s+r-1}{r-1} \text{ vs } s$$

$$\rightsquigarrow \alpha_{s,r-1} \geq \alpha_{s,r}.$$

$$\alpha_{s,r+1} = \frac{s+r+1}{\binom{s+r+1}{r}} = \frac{(s+1)+r}{\binom{s+r}{r} - 1 + \binom{s+r}{r-1} - r} \text{ vs } \alpha_{s,r} = \frac{s+1}{\binom{s+r}{r} - 1}$$

$$r \left[\binom{s+r}{r} - 1 \right] \text{ vs } (s+1) \left[\binom{s+r}{r-1} - r \right] \Leftrightarrow sr \text{ vs } \left[(s+1) - r \left(\frac{s+1}{r} \right) \right] \binom{s+r}{r-1} = 0$$

$$\rightsquigarrow \alpha_{s,r+1} \geq \alpha_{s,r}. \text{ Hence } \min_{1 \leq a \leq s, 1 \leq b \leq t} \alpha_{a,b} = \alpha_{s,r}. \quad \square$$

Proposition 4.4 *Let $s > r$, $t > r$. Then*

$$\min_{1 \leq a \leq s, 1 \leq b \leq t} \alpha_{a,b} = \min\{\alpha_{k,k}, \alpha_{k,k+1} : k = 1, \dots, \min\{s, t-1\}\}.$$

Proof:

Let $a < b$.

$$\alpha_{a,b} = \frac{a + \binom{a}{r} + \binom{b}{r}}{\binom{a+b}{r} - \binom{a}{r} - \binom{b}{r}} = \frac{A}{B} \text{ and } \alpha_{a+1,b-1} = \frac{(a+1) + \binom{a+1}{r} + \binom{b-1}{r}}{\binom{a+b}{r} - \binom{a+1}{r} - \binom{b-1}{r}} = \frac{C}{D}$$

$C < A$ and $B < D$, since $b > a$. Thus $\alpha_{a,b} \geq \alpha_{a+1,b-1} \geq \dots \geq \alpha_{\lfloor \frac{a+b}{2} \rfloor, \lceil \frac{a+b}{2} \rceil}$. \square

Proposition 4.5

$$\min_{1 \leq a \leq r-1} \alpha_{a,a} = \alpha_{r-1,r-1}.$$

Proof:

$$\alpha_{a,a} = \frac{a}{\binom{2a}{r}}. \text{ Then } \alpha_{a,a} = a < a+1 = \alpha_{a+1,a+1}, \quad 1 \leq a \leq \frac{r-2}{2}.$$

$$\alpha_{\lfloor \frac{r}{2} \rfloor + 1, \lfloor \frac{r}{2} \rfloor + 1} = \frac{\lfloor \frac{r}{2} \rfloor + 1}{r+1} \text{ or } \frac{\frac{r+1}{2}}{\frac{(r+1)(r+2)}{2}}.$$

$$\alpha_{a,a} = \frac{a}{\binom{2a}{r}} \text{ vs } \alpha_{a+1,a+1} = \frac{a+1}{\binom{2a+2}{r}} = \frac{a+1}{\binom{2a}{r} + \binom{2a}{r-1} + \binom{2a+1}{r-1}}$$

$$a \binom{2a+1}{r-1} + a \binom{2a}{r-1} \text{ vs } \binom{2a}{r}$$

$$a \frac{r(2a+1)}{(2a+2-r)(2a+1-r)} + a \frac{2a-r+1}{r} \text{ vs } 1$$

$$\frac{r(2a+1)}{(2a+2-r)(2a+1-r)} > 1 \Rightarrow \alpha_{a,a} \geq \alpha_{a+1,a+1} \rightsquigarrow \min_{1 \leq a \leq r-1} \alpha_{a,a} = \alpha_{r-1,r-1}. \quad \square$$

Proposition 4.6

$$\alpha_{r,r} \leq \alpha_{r-1,r-1}.$$

Proof:

$$\begin{aligned} \alpha_{r,r} &= \frac{r+2}{\binom{2r}{r} - 2} \text{ vs } \alpha_{r-1,r-1} = \frac{r-1}{\binom{2r-2}{r}} \\ \frac{r+2}{r-1} \text{ vs } \frac{\binom{2r}{r}}{\binom{2r-2}{r}} - \frac{2}{\binom{2r-2}{r}} &\geq \frac{2(2r-1)}{r-1} - \frac{2}{r+1} = \frac{4r^2}{(r-1)(r+1)} \end{aligned}$$

$$\Rightarrow \alpha_{r,r} \leq \alpha_{r-1,r-1}. \quad \square$$

Proposition 4.7

$$\alpha_{r,r} \geq \alpha_{r+1,r+1}.$$

Proof:

$$\begin{aligned} \alpha_{r,r} &= \frac{r+2}{\binom{2r}{r} - 2} \text{ vs } \alpha_{r+1,r+1} = \frac{3(r+1)}{\binom{2r+2}{r} - 2(r+1)} = \frac{3}{\frac{2}{r+2} \binom{2r+1}{r} - 2} \\ &\quad 2 \binom{2r+1}{r} - 2(r+2) \text{ vs } 3 \binom{2r}{r} - 6 \\ &\quad \left[\frac{2(2r+1)}{r+1} - 3 \right] \binom{2r}{r} = \frac{r-1}{r+1} \binom{2r}{r} \text{ vs } 2(r-1) \end{aligned}$$

$$\Rightarrow \alpha_{r,r} \geq \alpha_{r+1,r+1} \quad \square$$

Proposition 4.8

$$\alpha_{r,r} \geq \alpha_{r,r+1}$$

Proof:

$$\begin{aligned} \alpha_{r,r} &= \frac{r+2}{\binom{2r}{r} - 2} \text{ vs } \alpha_{r,r+1} = \frac{2(r+1)}{\binom{2r+1}{r} - (r+2)} \\ (r+2) \binom{2r+1}{r} - (r+2)^2 &\text{ vs } 2(r+1) \binom{2r}{r} - 4(r+1) \\ \left[\frac{(r+2)(2r+1)}{r+1} - 1 \right] \binom{2r}{r} &= 2(r+1) \binom{2r}{r} \text{ vs } (r+2)^2 - 4(r+1) = r(r-2) \end{aligned}$$

$$\Rightarrow \alpha_{r,r} \geq \alpha_{r,r+1}. \quad \square$$

Proposition 4.9

$$\alpha_{r+1,r+1} \leq \alpha_{r+2,r+2}.$$

Proof:

$$\alpha_{r+1,r+1} = \frac{3(r+1)}{\binom{2r+2}{r}-2(r+1)} = \frac{3(r+2)}{2\binom{2r+1}{r}-2(r+2)} \text{ vs}$$

$$\alpha_{r+2,r+2} = \frac{(r+2)+2\binom{r+2}{r}}{\binom{2r+4}{r}-2\binom{r+2}{r}} = \frac{(r+2)(r+4)}{2\binom{2r+3}{r}-(r+4)(r+1)}$$

$$6\binom{2r+3}{r} - 3(r+4)(r+1) \text{ vs } 2(r+4)\binom{2r+1}{r} - 2(r+2)(r+4)$$

$$\left[\frac{6(2r+3)(2r+2)}{(r+2)(r+3)} - 2(r+4) \right] \binom{2r+1}{r} \text{ vs } (r+4)(3(r+1) - 2(r+2))$$

Let $A = 6(2r+3)(2r+2) - 2(r+4)(r+3)(r+2) = 12(2r^2 + 5r + 3) - 2(r+4)(r^2 + 5r + 6)$. Then $A < 0$, $r \geq 4 \Rightarrow \alpha_{r+1,r+1} < \alpha_{r+2,r+2}$. Special case: $r = 3 \Rightarrow \alpha_{4,4} = \alpha_{5,5}$. \square

Proposition 4.10 *Let $a \geq r + 2$. Then*

$$\alpha_{a,a} \leq \alpha_{a+1,a+1}.$$

Proof:

$$\alpha_{a,a} = \frac{a+2\binom{a}{r}}{\binom{2a}{r}-2\binom{a}{r}} \text{ vs } \alpha_{a+1,a+1} = \frac{(a+1)+2\binom{a+1}{r}}{\binom{2a+2}{r}-2\binom{a+1}{r}} = \frac{[a+2\binom{a}{r}] + [2\binom{a}{r-1}+1]}{[\binom{2a}{r}-2\binom{a}{r}] + [\binom{2a+1}{r-1} + \binom{2a}{r-1} - 2\binom{a}{r-1}]}$$

$$\left[a + 2\binom{a}{r} \right] \left[\binom{2a+1}{r-1} + \binom{2a}{r-1} - 2\binom{a}{r-1} \right] \text{ vs } \left[2\binom{a}{r-1} + 1 \right] \left[\binom{2a}{r} - 2\binom{a}{r} \right]$$

$$LHS = \frac{a + 2\binom{a}{r}}{1 + 2\binom{a}{r-1}} \text{ vs } \frac{\binom{2a}{r} - 2\binom{a}{r}}{\binom{2a+1}{r-1} + \binom{2a}{r-1} - 2\binom{a}{r-1}} = RHS$$

$$LHS = \frac{a - \frac{a-r+1}{r}}{1 + 2\binom{a}{r-1}} + \frac{\frac{a-r+1}{r} + 2\binom{a}{r}}{1 + 2\binom{a}{r-1}} = \frac{(a+1)(r-1)}{r[1 + 2\binom{a}{r-1}]} + \frac{a-r+1}{r}$$

$$RHS = \frac{\binom{2a}{r} - \frac{2a-r+1}{2r}2\binom{a}{r}}{\binom{2a+1}{r-1} + \binom{2a}{r-1} - 2\binom{a}{r-1}} + \frac{[\frac{2a-r+1}{2r} - 1]2\binom{a}{r}}{\binom{2a+1}{r-1} + \binom{2a}{r-1} - 2\binom{a}{r-1}}$$

$$RHS \geq \frac{\binom{2a}{r} - \frac{2a-r+1}{2r}2\binom{a}{r}}{2\binom{2a+1}{r-1} - 2\binom{a}{r-1}} + \frac{[\frac{2a-3r+1}{r}]\binom{a}{r}}{\binom{2a+1}{r-1} + \binom{2a}{r-1} - 2\binom{a}{r-1}} = \frac{2a-r+1}{2r} + \frac{[\frac{2a-3r+1}{r}]\binom{a}{r}}{\binom{2a+1}{r-1} + \binom{2a}{r-1} - 2\binom{a}{r-1}}$$

i. Case: $2a + 1 \geq 3r$:

$$RHS \geq \frac{2a-r+1}{2r} \Rightarrow RHS - LHS \geq \frac{r-1}{2r} - \frac{(a+1)(r-1)}{r[1+2\binom{a}{r-1}]}$$

$$\frac{r-1}{2r} \text{ vs } \frac{(a+1)(r-1)}{r[1+2\binom{a}{r-1}]} \Leftrightarrow \frac{1}{2} \text{ vs } \frac{(a+1)}{1+2\binom{a}{r-1}}$$

$$1 + 2\binom{a}{r-1} \geq 2a + 2, a \geq \frac{3r-1}{2} \Rightarrow RHS \geq LHS.$$

ii. Case: $2r + 3 \leq 2a + 1 < 3r$:

$$RHS \geq \frac{2a-r+1}{2r} - \left[\frac{3r-2a-1}{r} \right] \frac{\binom{a}{r}}{\binom{2a+1}{r-1} + \binom{2a}{r-1} - 2\binom{a}{r-1}}$$

Let $A = \frac{\binom{a}{r}}{\binom{2a+1}{r-1} + \binom{2a}{r-1} - 2\binom{a}{r-1}}$, and $B = A^{-1}$. Then,

$$B = \frac{\binom{2a+1}{r-1}}{\binom{a}{r}} + \frac{\binom{2a}{r-1}}{\binom{a}{r}} - \frac{2\binom{a}{r-1}}{\binom{a}{r}} \geq 2 \left[\frac{\binom{2a}{r-1}}{\binom{a}{r}} - \frac{2\binom{a}{r-1}}{\binom{a}{r}} \right]$$

Since $\frac{\binom{2a}{r-1}}{\binom{a}{r}}$ is an increasing function of a , its minimum is attained when $a=r+2$. Furthermore, $\frac{\binom{2r+4}{r-1}}{\binom{r+2}{r}}$ is also an increasing function of r . Hence, its minimum is achieved when $r=3$. Similarly, the maximum value of $\frac{1}{a-r+1}$ is $\frac{1}{3}$. Then,

$$B \geq 2r \left[\frac{\binom{2r+2}{r-1}}{\binom{r+2}{r}} - \frac{1}{a-r+1} \right] = 2r \left[\frac{9}{2} - \frac{1}{3} \right] = \frac{25}{3}r \Rightarrow -A \geq -\frac{3}{25r}.$$

$$RHS \geq \frac{2a-r+1}{2r} - \left[\frac{3(3r-2a-1)}{25r^2} \right] \geq \frac{2a-r+1}{2r} - \frac{6}{75}$$

$$RHS - LHS \geq \frac{r-1}{2r} - \frac{6}{75} \frac{(a+1)(r-1)}{r[1+2\binom{a}{r-1}]} \geq \frac{r-1}{r} \left[\frac{1}{2} - \frac{r+3}{1+2\binom{r+2}{r-1}} \right] - \frac{6}{75} \geq \frac{2}{3} \left[\frac{39}{126} \right] - \frac{6}{75} \geq 0.$$

$$\Rightarrow RHS \geq LHS.$$

Hence, $\alpha_{a,a} \leq \alpha_{a+1,a+1}$. □

Proposition 4.11 *Let $a \geq r$. Then,*

$$\alpha_{a,a} \leq \alpha_{a-1,a}.$$

Proof:

$$\begin{aligned} \alpha_{a,a} &= \frac{a + 2\binom{a}{r}}{\binom{2a}{r} - 2\binom{a}{r}} \text{ vs } \alpha_{a-1,a} = \frac{a - 1 + \binom{a}{r} + \binom{a-1}{r}}{\binom{2a-1}{r} - \binom{a}{r} - \binom{a-1}{r}} = \frac{a + 2\binom{a}{r} - 1 - \binom{a-1}{r-1}}{\binom{2a}{r} - 2\binom{a}{r} - \binom{2a-1}{r-1} + \binom{a-1}{r-1}} \\ LHS &= \frac{\binom{2a}{r} - 2\binom{a}{r}}{\binom{2a-1}{r-1} - \binom{a-1}{r-1}} = 1 + \frac{\binom{2a-1}{r} - \binom{a-1}{r} - \binom{a}{r}}{\binom{2a-1}{r-1} - \binom{a-1}{r-1}} \text{ vs } RHS = \frac{a + 2\binom{a}{r}}{\binom{a-1}{r-1} + 1} = 2 + \frac{2\binom{a-1}{r} + a - 2}{\binom{a-1}{r-1} + 1} \\ LHS &\leq 1 + \frac{\binom{2a-2}{r-1} + \dots + \binom{a-1}{r-1}}{\binom{2a-2}{r-2} + \dots + \binom{a-1}{r-2}} \leq 1 + \frac{2(a-1)}{2a-r-1} \leq 3 < 2 + \frac{2(a-1)}{a-r} \leq RHS \end{aligned}$$

$$\leadsto \alpha_{a,a} \leq \alpha_{a-1,a}. \quad \square$$

Theorem 4.6 *Let $S, T \subseteq V$, $S \cap T = \emptyset$, and $s = |S|$, $t = |T|$. Then the cut inequality*

$$-\sum_{i \in S} x_i + \alpha_{\min} \sum_{e_j \in \delta(S:T)} y_j - \sum_{e_j \in \gamma(S)} y_j - \sum_{e_j \in \gamma(T)} y_j \leq 0$$

is valid for $RP(H_n^r)$.

4.5 Computational Results

In this section, computational results on random problems are reported. There are two test sets in the experimentation. The first set consists of nonnegative edge weights and nonpositive node weights. That is, every node has a cost figure and every hyperedge has a profit value. In the second set, mixed values are allowed. In any of the cases, there are specific range of objective function coefficients. If c_i 's are relatively large as compared to the d_j 's, then the solution is $x_i = 0 = y_j$, $\forall i, j$. On the other hand, if d_j 's are relatively larger than c_i 's, the trivial solution is $x_i = 1 = y_j$, $\forall i, j$. Hence, c_i and d_j values should be balanced. Hyperedge profits are drawn from uniform distribution between 0 and 600 for the first set, and between -200 and 600 for the second set. Node costs are also drawn from the

Inequality Type	Subclass	# Constraints				STAGE
		n=10		n=15		
		r=3	r=4	r=3	r=4	
	$y_j \leq x_i$	360	840	1365	5460	I
	$\sum_{i \in e} x_i - y_j \leq r - 1$	120	210	455	1365	I
	$x_i \leq 1$	10	10	15	15	I
Clique	k=r+1	210	252	1365	3003	II
	k=r+2	252	210	3003	5005	II
	k=r+3	210	120	5005	6435	III
	k=n-3	120	120	455	455	II
	k=n-2	48	48	105	105	II
	k=n-1	10	10	15	15	II
Cut	l=r+1	840	1260	5460	15015	II
	l=r+2	2520	3150	♠30030	♠75075	III

♠: These constraints are added up to the limit of LP solver!

Table 4.1: The stages of computation.

uniform distribution from 0 to 9500 for the first set, and from -2000 to 9500 for the second set.

The first test set consists of 25 random problems on complete 3-uniform and 4-uniform hypergraphs of order 10 and 15. The second set contains 40 random problems of the above sizes. The size of test problems seems to be small at the first glance. However, typical values for the case of complete graphs are 20, 30 up to 40. There are $\binom{20}{2}=190$ edges in the case of $n=20$; $\binom{40}{2}=780$ edges when $n=40$. In our case, if $n=10$, there are $\binom{10}{3}=120$ and $\binom{10}{4}=210$ hyperedges; and if $n=15$, there are $\binom{15}{3}=455$ and $\binom{15}{4}=1365$ hyperedges.

The set of constraints that are considered are tabulated in Table 4.1. We have designed four stages. The first stage contains easy-to-list constraints. A quick solution is obtained immediately after solving these constraints. In the second stage, the constraints force the current solution to be an incidence vector. Finally, in the third stage, there are lots of constraints that are checked whether they are satisfied by the solution found in the previous run. In this stage, the clique and cut constraints are added up to the limit of LP solver. If an integer solution is

$c_i \in U[-9500, 0]$, and $d_j \in U[1; 600]$,								
STAGE	CASE	NO	PRIMAL CPLEX			BARRIER CPLEX		
			Mean	Min	Max	Mean	Min	Max
ONE	10,3 Integer	25	1.1353	0.6000	1.9833	30.7020	27.4500	33.7502
	10,4 Integer	25	3.6920	2.5833	5.4167	349.9741	291.5667	393.7500
	15,3 Integer	25	17.9760	10.2833	24.7833	1167.1556	923.3333	1451.9833
	15,4 Integer	25	158.1273	127.3666	248.1833	Not Enough Memory!		

Table 4.2: The results of the first set of the computational study on $RP(H_n^r)$ for $n=10,15$ and $r=3,4$.

not found at the end of stage III, we resort to branch and bound. The initial bound is obtained from clustering using similarity coefficients followed by an interchange routine across the cut of clustering. These techniques are introduced in the second chapter.

The experiments are done at a 22 MIPS Sun system and Cplex 3.0 is used. Two different Cplex routines are compared in this computational study. The first routine is a primal simplex routine. The second routine is based on an interior point method, called barrier routine. For the details of these routines, the interested reader is referred to [28].

Each problem set consists of randomly generated 40 problems. In each set, the mean, the minimum and the maximum values for the CPU seconds used by the random problems are reported. These are net CPU values obtained from the "sys/times" routine. Furthermore, each test set are divided into two subcases in each stage according to the solution characteristics: integer and fractional. The CPU times are reported for separately each subcase together with the number of problems that are in the subcase. The fractional problems are fed into the next stage, and so on. In the fourth stage, branch and bound is performed by using the primal method only.

The results for the first set are tabulated in Table 4.2. All of the test problems in the first set have integer solutions at the end of the first stage. The barrier method is relatively inferior than the primal simplex method as the number of CPU seconds is considered. This case is investigated in detail in the next section.

$c_i \in U[-9000; 2000]$, and $d_j \in U[-200; 600]$,								
STAGE	CASE	NO	PRIMAL CPLEX			BARRIER CPLEX		
			Mean	Min	Max	Mean	Min	Max
ONE	10,3 Integer	30	1.0244	0.3833	1.7000	36.0494	27.4833	43.0667
	10,3 Fractional	10	0.6483	0.0167	1.0833	37.4467	30.5667	42.9167
	10,4 Integer	33	3.7772	1.1167	6.9000	359.0763	290.8833	427.2667
	10,4 Fractional	7	1.8262	1.2833	4.4000	402.7071	359.2667	461.0333
	15,3 Integer	30	7.2827	4.6333	10.2833	1445.0133	1294.7333	1671.2833
	15,3 Fractional	10	7.1500	5.3000	11.4166	1445.1833	1296.7500	1668.8000
	15,4 Integer	30	148.8794	43.3333	372.0166	Not Enough Memory!		
	15,4 Fractional	10	95.8350	42.7500	156.0000	Not Enough Memory!		
TWO	10,3 Integer	3	4.0500	1.4833	6.0000	1014.2363	923.4556	1134.2966
	10,3 Fractional	7	7.4595	4.3000	11.1166	978.3884	912.3333	1098.6776
	10,4 Integer	2	3.3583	3.0666	3.6500	2876.3333	2456.3211	3296.3455
	10,4 Fractional	5	16.0333	2.9833	60.3667	2912.6667	2678.9251	3451.7000
	15,3 Integer	4	609.0944	236.3333	795.8833	7814.2290	7145.9065	8491.9300
	15,3 Fractional	6	147.8666	26.1000	736.3666	8345.6667	7910.2397	8993.1455
	15,4 Integer	3	1114.8900	754.5264	1447.3270	Not Enough Memory!		
	15,4 Fractional	7	800.1535	422.1990	1411.4450	Not Enough Memory!		
THREE	10,3 Integer	3	7.8930	2.0567	9.1634	2161.7222	1858.8167	2402.6333
	10,3 Fractional	4	8.2375	1.9463	12.9834	2610.4976	2041.2833	3669.1833
	10,4 Integer	2	6.8622	5.9367	7.7877	4684.0649	4389.9876	4978.1422
	10,4 Fractional	3	14.3457	7.5337	30.6534	4958.8992	4602.0133	5318.5555
	15,3 Integer	2	435.9608	85.0486	786.8756	28720.8004	25982.6667	31458.9399
	15,3 Fractional	4	327.2303	124.7623	837.9631	29736.8967	26545.2937	36754.9033
	15,4 Integer	3	1294.2846	616.8333	1936.6667	Not Enough Memory!		
	15,4 Fractional	4	1014.3356	714.7667	1744.5255	Not Enough Memory!		
FOUR	10,3 Integer	4	3.8166	2.0166	6.2333	Primal Cplex is used!		
	10,4 Integer	3	6.7166	1.2667	13.8433	Primal Cplex is used!		
	15,3 Integer	4	109.4330	17.7166	289.4641	Primal Cplex is used!		
	15,4 Integer	4	1536.5533	174.5442	2926.6667	Primal Cplex is used!		

Table 4.3: The results of the second set of the computational study on $RP(H_n^r)$ for $n=10,15$ and $r=3,4$.

The results for the second set are presented in Table 4.3. We have obtained integer solutions of the 145 test problems out of 160 by using the inequalities listed in this study. The percentage integer solutions obtained before branch and bound is 90.625 which is quite high. At the end of the first stage, more than $\frac{3}{4}$ of the test problems have integer solutions. By adding the inequalities in the second stage, we have obtained 12 integer solutions out of 37 problems which is approximately 32.43 percent. Among the remaining problems, we have found 10 integer solutions out of 25 problems in stage IV, yielding a percentage of 40.

The simplex method outperforms the barrier method in terms of speed. As the number of constraints is increased, the barrier method's CPU usage jumps up. As the number of variables is increased, its time scores get worse. The behavior of the primal simplex method against these changes is relatively tolerable.

4.6 A Special Case

Let us consider the case where the weights of all hyperedges are nonnegative and all node weights are nonpositive. If we take nodes as machines and hyperedges as parts and use annual operating costs and annual profit values as node and hyperedge weights, we will be in this special case. Then our problem is

$$\begin{aligned} & \text{Maximize } \sum_{j=1}^{\binom{n}{r}} d_j y_j - \sum_{i=1}^n c_i x_i \\ & \text{subject to:} \\ & \quad y_j \leq x_i \quad \forall i \in e_j, \forall j = 1, \dots, \binom{n}{r} \\ & \quad \sum_{i \in e_j} x_i - y_j \leq r - 1 \quad \forall j = 1, \dots, \binom{n}{r} \\ & \quad y_j \geq 0, \quad \forall j = 1, \dots, \binom{n}{r} \\ & \quad x_i \leq 1, \quad \forall i = 1, \dots, n \quad \} \\ & \quad x_i = 0 \text{ or } 1, \quad i = 1, \dots, n \\ & \quad y_j = 0 \text{ or } 1, \quad j = 1, \dots, m \end{aligned}$$

The constraints $\sum_{i \in e_j} x_i \leq (r-1) + y_j$ prevents the case $y_j = 0$ when $x_i = 1, \forall i \in e_j$. These constraints are unnecessary in the special case because the objective

function forces y_j to take higher and higher values. Let us consider the following linear relaxation:

$$\text{Maximize } \sum_{j=1}^{\binom{n}{r}} d_j y_j - \sum_{i=1}^n c_i x_i$$

subject to:

$$\begin{aligned} y_j - x_i &\leq 0 \quad \forall i \in e_j, \forall j = 1, \dots, \binom{n}{r} \\ x_i &\leq 1, \quad \forall i = 1, \dots, n \\ x_i &\geq 0, \quad \forall i = 1, \dots, n \\ y_j &\geq 0, \quad \forall j = 1, \dots, \binom{n}{r} \end{aligned}$$

Theorem 4.7 *The LP solution of the above formulation is always integral.*

Proof:

Let A be the coefficient matrix of the above formulation. A has only $0, \pm 1$ entries. Furthermore, there are at most two nonzero elements in any row of A . The Heller, Tompkins & Gale criterion for total unimodularity states that if a matrix contains no elements other than $0, +1, -1$ and if no row contains more than two nonzero elements then the matrix is totally unimodular if and only if the columns are partitioned into two such that for any row, unequal nonzero elements are in the same component and equal nonzero elements are not in the same component. Here, we have a component containing all columns and one null component. Thus A is totally unimodular. Hence, every solution to a totally unimodular coefficient matrix with integral right hand side is integral. \square

Corollary 4.1 *The above problem is polynomially solvable, i.e., in \mathcal{P} .*

Corollary 4.2 *Let $c_i \geq 0$, and $d_j \geq 0$. The following problem is polynomially solvable:*

Maximize $z = \sum_{j=1}^m d_j y_j - \sum_{i=1}^n c_i x_i$

subject to:

$$\begin{array}{lll} y_j = x_{i_1} \cdot x_{i_2} \cdot \dots \cdot x_{i_{k_j}} & j = 1, \dots, m & e_j = \{i_1, i_2, \dots, i_{k_j}\} \\ x_i = 0 \text{ or } 1, & i = 1, \dots, n & \\ y_j = 0 \text{ or } 1, & j = 1, \dots, m & \end{array}$$

Chapter 5

R-UNIFORM HYPERGRAPH CUT POLYTOPE

R-uniform hypergraphs are generalizations of graphs in which every hyperedge has exactly r end-points. The complete r -uniform hypergraph cut polytope $P_C(H_n^r)$ is the convex hull of the incidence vectors of bipartitions in a complete r -uniform hypergraph H_n^r . This family includes cut polytope of the complete graph $P_C(K_n)$. In this chapter, $P_C(H_n^r)$ is proved to be full dimensional, except for the case $r=3$. Moreover, a number of valid inequalities some of which define facets are presented. Finally, a procedure based on polyhedral combinatorics is experimented for solving max-cut and hypergraph bipartitioning problems.

5.1 Dimension

The first step in studying a polyhedron is to check its dimension. If a polyhedron is full dimensional, then facet defining inequalities are the minimal representation of that polyhedron. Moreover, full dimensionality simplifies some facetness proofs. The main result of this section is that complete r -uniform hypergraph cut polytope is full dimensional except $r=3$.

Let $\mathcal{X}(\mathcal{C}_{H_n^r}^s)$ be the incidence matrix of the cuts of size s in a complete r -uniform hypergraph H_n^r :

$$\mathcal{X}(\mathcal{C}_{H_n^r}^s)_i^j = \begin{cases} 1, & \text{if } e_j \in \delta(s[i]); \\ 0, & \text{if } e_j \in \gamma(s[i]) \text{ or } e_j \in \gamma(V \setminus s[i]), \end{cases}$$

where $s[i]$, $i = 1, \dots, \binom{n}{s}$, is i^{th} specific cut set whose size is s , $s = 1, \dots, \lfloor \frac{n}{2} \rfloor$. Without loss of generality we may assume that these cut sets are ordered lexicographically, i.e., $s[1] = \{1, 2, \dots, s\}$, $s[2] = \{1, \dots, (s-1), (s+1)\}$, $s[\binom{n}{s}] = \{(n-s+1), \dots, n\}$.

Let $\mathcal{Y}(\mathcal{C}_{H_n^r}^s)$ be modified incidence matrix of the cuts of size s in a complete r -uniform hypergraph H_n^r . In $\mathcal{Y}(\mathcal{C}_{H_n^r}^s)$, the edges inside the complementary part take the value 1:

$$\mathcal{Y}(\mathcal{C}_{H_n^r}^s)_i^j = \begin{cases} 1, & \text{if } e_j \in \delta(s[i]) \text{ or } e_j \in \gamma(V \setminus s[i]); \\ 0, & \text{if } e_j \in \gamma(s[i]). \end{cases}$$

Let $\mathcal{Z}(\mathcal{C}_{H_n^r}^s)$ be modified incidence matrix of the cuts of size s in a complete r -uniform hypergraph H_n^r in which the hyperedges inside the cut set also have the value of 1:

$$\mathcal{Z}(\mathcal{C}_{H_n^r}^s)_i^j = \begin{cases} 1, & \text{if } e_j \in \delta(s[i]) \text{ or } e_j \in \gamma(s[i]); \\ 0, & \text{if } e_j \in \gamma(V \setminus s[i]). \end{cases}$$

Proposition 5.1

$$\mathcal{X}(\mathcal{C}_{H_n^r}^s) = \left[\begin{array}{c|c} \mathcal{Y}(\mathcal{C}_{H_{n-1}^{r-1}}^{s-1}) & \mathcal{X}(\mathcal{C}_{H_{n-1}^{r-1}}^{s-1}) \\ \hline \mathcal{Z}(\mathcal{C}_{H_{n-1}^{r-1}}^s) & \mathcal{X}(\mathcal{C}_{H_{n-1}^{r-1}}^s) \end{array} \right].$$

Proof:

$\mathcal{X}(\mathcal{C}_{H_n^r}^s) \in \mathbb{B}^{\binom{n}{s} \times \binom{n}{r}}$. Let $\binom{n}{s} = \binom{n-1}{s-1} + \binom{n-1}{s}$ and $\binom{n}{r} = \binom{n-1}{r-1} + \binom{n-1}{r}$. Then $\mathcal{X}(\mathcal{C}_{H_n^r}^s)$ is partitioned into four parts:

$$\mathcal{X}(\mathcal{C}_{H_n^r}^s) = \left[\begin{array}{c|c} I & II \\ \hline III & IV \end{array} \right]$$

where $I \in \mathbb{B}^{\binom{n-1}{s-1} \times \binom{n-1}{r-1}}$, $II \in \mathbb{B}^{\binom{n-1}{s-1} \times \binom{n-1}{r}}$, $III \in \mathbb{B}^{\binom{n-1}{s} \times \binom{n-1}{r-1}}$, $IV \in \mathbb{B}^{\binom{n-1}{s} \times \binom{n-1}{r}}$.

There are two immediate observations:

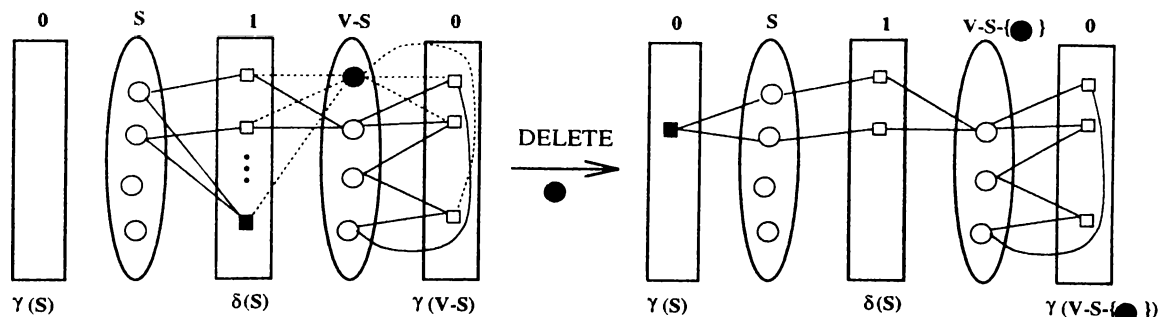


Figure 5.1: A cut in region I.

$1 \in s[i], \forall i = 1, \dots, \binom{n-1}{s-1}$ and $1 \in V \setminus s[i], \forall i = \binom{n-1}{s-1} + 1, \dots, \binom{n}{s}$, and $1 \in e_j, \forall j = 1, \dots, \binom{n-1}{r-1}$ and $1 \notin e_j, \forall j = \binom{n-1}{r-1} + 1, \dots, \binom{n}{r}$.

i.) $I = \mathcal{Y}(\mathcal{C}_{H_{n-1}^{r-1}}^{s-1})$:

Node 1 is in the set S and all hyperedges are in the star of 1. The situation is illustrated in Figure 5.1. Node 1 is represented as \bullet . In the incidence matrix, hyperedges inside S and $V \setminus S$ have value of 0 and hyperedges in the cut take the value of 1. Since all hyperedges should be incident to node 1, $\gamma(V \setminus S) = \emptyset$. If node 1 is deleted, then the resulting hypergraph has 1 less uniformity, all the hyperedges loose dotted connections. Moreover, the size of the set S is reduced by one. We have then $\mathcal{C}_{H_{n-1}^{r-1}}^{s-1}$. Observe that the hyperedges that have incidence to only node 1 in S , like the one \blacksquare in the figure, are moved to $\gamma(V \setminus S)$ which is empty beforehand. These hyperedges have the value 1 in region I. Thus, region I is $\mathcal{Y}(\mathcal{C}_{H_{n-1}^{r-1}}^{s-1})$.

ii.) $III = \mathcal{Z}(\mathcal{C}_{H_{n-1}^{r-1}}^s)$:

In this case, node 1 is in the set $V \setminus S$ and all hyperedges are in the star of 1. The situation is illustrated in Figure 5.2. Since all hyperedges should be incident to node 1, and $1 \in V \setminus S$, $\gamma(S) = \emptyset$. If node 1 is deleted, the resulting hypergraph has 1 less uniformity, all the hyperedges loose dotted connections. Moreover, the size of the set S is not reduced. We then have $\mathcal{C}_{H_{n-1}^{r-1}}^s$. Observe that the hyperedges that have incident to only node 1 in S , like the one \blacksquare in the figure, is moved to $\gamma(S)$ which is empty beforehand. These hyperedges has the value 1 in region III. Thus, III is equivalent to

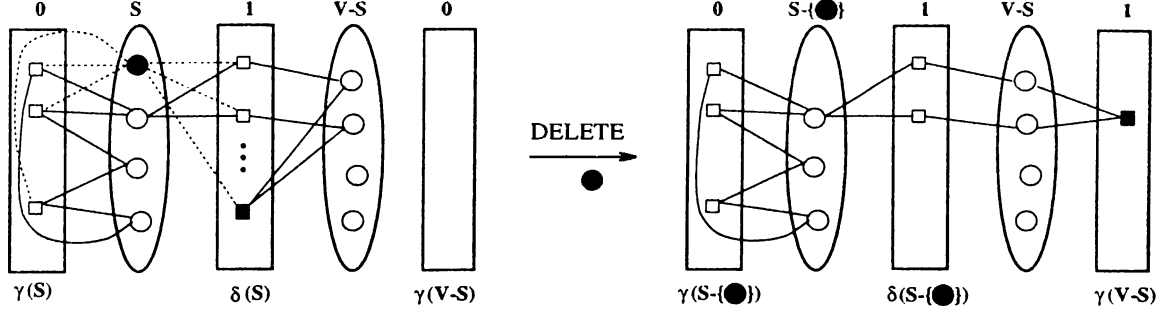


Figure 5.2: A cut in region III.

$$\mathcal{Z}(\mathcal{C}_{H_{n-1}^r}^s).$$

iii.) $II = \mathcal{X}(\mathcal{C}_{H_{n-1}^r}^{s-1})$:

Since no hyperedge is incident to node 1 in this region, deletion of node 1 do not alter uniformity. Since node 1 is previously in S , cut size is decreased by one. Hence, we have $\mathcal{X}(\mathcal{C}_{H_{n-1}^r}^{s-1})$.

iv.) $IV = \mathcal{X}(\mathcal{C}_{H_{n-1}^r}^s)$:

Like in the above case, r -uniformity is preserved. Furthermore, the size of the cut set remain unchanged. Hence, region IV is $\mathcal{X}(\mathcal{C}_{H_{n-1}^r}^s)$.

□.

Proposition 5.2

$$\mathcal{Y}(\mathcal{C}_{H_n^r}^s) = \left[\begin{array}{c|c} \mathcal{Y}(\mathcal{C}_{H_{n-1}^r}^{s-1}) & \mathcal{Y}(\mathcal{C}_{H_{n-1}^r}^{s-1}) \\ \hline \mathbf{1} & \mathcal{Y}(\mathcal{C}_{H_{n-1}^r}^s) \end{array} \right] \text{ and } \mathcal{Z}(\mathcal{C}_{H_n^r}^s) = \left[\begin{array}{c|c} \mathbf{1} & \mathcal{Z}(\mathcal{C}_{H_{n-1}^r}^{s-1}) \\ \hline \mathcal{Z}(\mathcal{C}_{H_{n-1}^r}^s) & \mathcal{Z}(\mathcal{C}_{H_{n-1}^r}^s) \end{array} \right].$$

Proof is quite similar to the proof of the previous proposition.

Proposition 5.3 Let $\mathcal{X}_1, \mathcal{X}_2, \dots, \mathcal{X}_m$ be some incidence vectors of cuts of size $s \neq 1$ in H_n^r . $P_C(H_n^r)$ is full dimensional if $(e - \mathcal{X}_1), (e - \mathcal{X}_2), \dots, (e - \mathcal{X}_m)$ are linearly independent and $m = \binom{n}{r}$.

Proof:

Assume that we generate all $\mathcal{X}(\mathcal{C}_{H_n^r}^s)$'s $\forall s = 1, \dots, \lfloor \frac{n}{2} \rfloor$. Put all these incidence

matrices into one binary matrix, A , having $\sum_{s=1}^{\lfloor \frac{n}{2} \rfloor} \binom{n}{s}$ rows and $\binom{n}{r}$ columns. The dimension of $P_C(H_n^r)$ is equal to the row rank of A .

The first n rows of A is the incidence vectors of stars of the nodes, $\mathcal{X}(C_{H_n^r}^{s=1})$. There are exactly r 1s in any column of $\mathcal{X}(C_{H_n^r}^1)$. If every row of $\mathcal{X}(C_{H_n^r}^1)$ is added and the solution is divided by r , the vector e of size $\binom{n}{r}$ is obtained. Delete first n rows of A and add the vector e into A . Call the new matrix as B . Then the row rank of B is a lower bound to the row rank of A . Now, multiply every row of B by -1 and add the first row to all other rows. Then, delete the first row. Call the resultant matrix as C . Actually, $C_i = e - A_{n+i}$, $\forall i = 1, \dots, \sum_{s=2}^{\lfloor \frac{n}{2} \rfloor} \binom{n}{s}$.

Assume that we have found $m = \binom{n}{r}$ independent rows in C . They make the row rank of C , B , and A be equal to the number of columns. So, if $(e - \mathcal{X}_1), (e - \mathcal{X}_2), \dots, (e - \mathcal{X}_m)$ are linearly independent and $m = \binom{n}{r}$, then $P_C(H_n^r)$ is full dimensional. \square

Proposition 5.4 Let $A = \left[\begin{array}{c|c} B & I_n^* \\ \hline I_n & I_n^* \end{array} \right]$, where every row of $B \in \mathbb{B}^{n \times n}$ contains more than one nonzero elements. Then, A has full row rank.

Proof:

Let $A = \left[\begin{array}{c} C \\ D \end{array} \right]$, where $C = [B \mid I_n^*]$ and $D = [I_n \mid I_n^*]$. Since, I_n^* has full rank, all rows of C and D are linearly independent. Suppose that rows of A are not linearly independent. Then, for some $i \in \{1, \dots, n\}$, $A_i = \sum_{j \neq i} \beta_j A_j$ where β_j 's are linear coefficients. Since, last n columns of C and D are the same, $\beta_j = 0$, $j \neq i + n$; and $\beta_{i+n} = 1$. However, B_i has more than one nonzero elements and e_{n-i} has just one nonzero element. It is a contradiction. Hence, A has full row rank. \square

Proposition 5.5 Let $A = \left[\begin{array}{c|c} I_n & B \\ \hline B^T & I_n \end{array} \right]$, where every row or column of $B \in \mathbb{B}^{n \times n}$ contains more than one nonzero elements. Then, A has full row rank.

Proof:

Let $A = \left[\begin{array}{c} C \\ D \end{array} \right]$, where $C = [I_n \mid B]$ and $D = [B^T \mid I_n]$. Since, I_n^* has full rank,

all rows of C and D are linearly independent. Suppose that rows of A are not linearly independent. Then, for some $i \in \{1, \dots, n\}$, $A_i = \sum_{j \neq i} \beta_j A_j$. Since, C contains an identity matrix in first n columns, $\beta_j = 0$, $j = 1, \dots, i-1, i+1, \dots, n$. So, $A_i = \sum_{j=n+1}^{2n} \beta_j A_j$. For each nonzero element of B_j , say B_i^k , D_k is used with multiplicity $\beta_{n+k} = 1$. For each such k , there must be more than one nonzero element in the first n elements of A_i . There is a contradiction. Thus, A has full row rank. \square

Proposition 5.6 $\mathcal{X}(\mathcal{C}_{H_n^s}^{s=r})$ is symmetric.

Proof:

The proof follows from the fact that both columns and rows are lexicographically ordered, and each $s[i]$, $i = 1, \dots, \binom{n}{r}$ and each e_j , $j = 1, \dots, \binom{n}{r}$ has the same (r) number of elements. So, $\mathcal{X}(\mathcal{C}_{H_n^s}^{s=r})_i = \mathcal{X}(\mathcal{C}_{H_n^s}^{s=r})^i$. Thus, $\mathcal{X}(\mathcal{C}_{H_n^s}^{s=r})^T = \mathcal{X}(\mathcal{C}_{H_n^s}^{s=r})$. \square

Proposition 5.7 $\mathcal{Y}(\mathcal{C}_{H_n^s}^{s=r}) = \mathbf{1} - I_{\binom{n}{r}}$

Proof:

For each set $s[i]$, there is only one internal edge in $\gamma(s[i])$ which is exactly the hyperedge e_i . So, $\mathcal{Y}(\mathcal{C}_{H_n^s}^{s=r})_i$ has one zero on the i^{th} column, and all other elements are all 1. Hence, $\mathcal{Y}(\mathcal{C}_{H_n^s}^{s=r}) = \mathbf{1} - I_{\binom{n}{r}}$. \square

Proposition 5.8 There are $\binom{n}{r}$ linearly independent row vectors in $\mathcal{X}(\mathcal{C}_{H_n^s}^s)$, $s = 2, \dots, \lfloor \frac{n}{2} \rfloor$; for $4 \leq r \leq \lfloor \frac{n}{2} \rfloor$.

Proof: The proposition is proved by cases:

i.) $n = 2r$:

Consider the cuts of size $s = r = \frac{n}{2}$. In any such cut, S and $V \setminus S$ contain

only one edge. Then $\mathcal{X}(\mathcal{C}_{H_{2r}}^r) = \left[\begin{array}{c|c} \mathbf{1} - I_{\frac{\binom{2r}{r}}{2}} & \mathbf{1} - I_{\frac{\binom{2r}{r}}{2}}^* \\ \hline \mathbf{1} - I_{\frac{\binom{2r}{r}}{2}}^* & \mathbf{1} - I_{\frac{\binom{2r}{r}}{2}} \end{array} \right]$.

$\mathcal{X}(\mathcal{C}_{H_{2r}}^r)$ is a skew-symmetric binary matrix. So, it has half row rank. Let

$\mathbf{1} - D$ be the matrix having first $\binom{n-1}{r}$ rows of $\mathcal{X}(C_{H_{2r}^r}^r)$. D has the form $[I_{\binom{2r-1}{r}} \mid I_{\binom{2r-1}{r-1}}^*]$, and it has full row rank.

Consider $\mathcal{X}(C_{H_{2r}^{r-1}}^{r-1}) = \left[\begin{array}{c|c} \mathcal{Y}(C_{H_{2r-1}^{r-2}}^{r-2}) & \mathcal{X}(C_{H_{2r-1}^{r-2}}^{r-2}) \\ \mathcal{Z}(C_{H_{2r-1}^{r-1}}^{r-1}) & \mathcal{X}(C_{H_{2r-1}^{r-1}}^{r-1}) \end{array} \right]$. Consider a cut $\delta(S) \in$

$C_{H_{2r-1}^{r-1}}^{r-1}$. The only hyperedge not in the cut is the one contained in $V \setminus S$.

Hence, $\mathcal{X}(C_{H_{2r-1}^{r-1}}^{r-1}) = \mathbf{1} - I_{\binom{2r-1}{r-1}}^*$. Consider a cut $\delta(S) \in C_{H_{2r-1}^{r-1}}^{r-1}$. There are r

hyperedges contained in $V \setminus S$. So, there are r 0s in any row of $\mathcal{Z}(C_{H_{2r-1}^{r-1}}^{r-1})$. Let

$\mathbf{1} - B = \mathcal{Z}(C_{H_{2r-1}^{r-1}}^{r-1})$, and let $C = [B \mid \mathbf{1} - \mathcal{X}(C_{H_{2r-1}^{r-1}}^{r-1})]$. Then, $C = [B \mid I_{\binom{2r-1}{r-1}}^*]$.

So, C has full row rank. Let $A_1 = \left[\begin{array}{c} C \\ D \end{array} \right]$, and apply Proposition 5.4.

ii.) $n = 2r + 1$:

Consider $\mathcal{X}(C_{H_{2r+1}^r}^r) = \left[\begin{array}{c|c} \mathcal{Y}(C_{H_{2r}^{r-1}}^{r-1}) & \mathcal{X}(C_{H_{2r}^{r-1}}^{r-1}) \\ \mathcal{Z}(C_{H_{2r}^{r-1}}^{r-1}) & \mathcal{X}(C_{H_{2r}^{r-1}}^{r-1}) \end{array} \right]$. By the propositions 5.2,

5.6, and 5.7 we have

$$A_2 = \mathbf{1} - \mathcal{X}(C_{H_{2r+1}^r}^r) = \left[\begin{array}{c|c|c|c} I & \mathbf{0} & \mathbf{0} & \mathbf{1} - \mathcal{X}(C_{H_{2r-1}^{r-1}}^{r-1}) \\ \mathbf{0} & I & \mathbf{1} - \mathcal{Y}(C_{H_{2r-1}^{r-2}}^{r-2}) & \mathbf{1} - \mathcal{X}(C_{H_{2r-1}^{r-2}}^{r-2}) \\ \mathbf{0} & \mathbf{1} - \mathcal{Z}(C_{H_{2r-1}^{r-1}}^{r-1}) & \mathbf{1} - \mathcal{Z}(C_{H_{2r-1}^{r-1}}^{r-1}) & \mathbf{1} - \mathcal{X}(C_{H_{2r-1}^{r-1}}^{r-1}) \\ \mathbf{1} - \mathcal{Z}(C_{H_{2r-2}^{r-2}}^{r-2}) & \mathbf{1} - \mathcal{Z}(C_{H_{2r-1}^{r-1}}^{r-1}) & I^* & I \end{array} \right]$$

$$= \left[\begin{array}{c|c|c|c} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \end{array} \right]$$

Since $A_1 = \left[\begin{array}{c} 7 \ 8 \\ 11 \ 12 \end{array} \right]$ has full row rank, $\left[\begin{array}{c} 5 \ 6 \ 7 \ 8 \\ 9 \ 10 \ 11 \ 12 \end{array} \right]$, has full row

rank. Since 5 and 9 are $\mathbf{0}$ and 1 is I, $\left[\begin{array}{c} 1 \ 2 \ 3 \ 4 \\ 5 \ 6 \ 7 \ 8 \\ 9 \ 10 \ 11 \ 12 \end{array} \right]$ has full row rank.

Since 16 is I, $[13 \ 14 \ 15 \ 16]$ has full row rank. Furthermore, A_2 has full row rank since 2,3,5, and 9 are $\mathbf{0}$; and 1 is I; but 13, 14 and 15 are nonzero.

iii.) $n \geq 2r + 1$:

Investigate whether $\mathbf{1} - \mathcal{X}(C_{H_n}^r)$ has full row rank or not. This is proved by induction on n . Initialization step ($n = 2r$) is proved in the above case. Now, assume that $A_{n-1} = \mathbf{1} - \mathcal{X}(C_{H_{n-1}}^r)$ has full row rank.

$$\text{Let } A_n = \left[\begin{array}{c|c|c|c} 1 & 2 & 3 & 4 \\ \hline 5 & 6 & 7 & 8 \\ \hline 9 & 10 & 11 & 12 \\ \hline 13 & 14 & 15 & 16 \end{array} \right] = \left[\begin{array}{c|c|c|c} I & \mathbf{0} & \mathbf{0} & 4 \\ \hline 0 & I & B & 8 \\ \hline 0 & B^T & I & 12 \\ \hline 4^T & 8^T & 12^T & 16 \end{array} \right]. \text{ Let } A = \left[\begin{array}{c|c} I & B \\ \hline B^T & I \end{array} \right]$$

has full row rank by Proposition 5.5. Therefore, $\left[\begin{array}{c|c} 5 & 6 & 7 & 8 \\ \hline 9 & 10 & 11 & 12 \end{array} \right]$ has full

row rank. Since 1 is I , 5 and 9 are $\mathbf{0}$, $\left[\begin{array}{c|c} 1 & 2 & 3 & 4 \\ \hline 5 & 6 & 7 & 8 \\ \hline 9 & 10 & 11 & 12 \end{array} \right]$ has full row rank.

Since A_{n-1} has full row rank, the rows of $[13 \ 14 \ 15 \ 16]$ are linearly independent. A_n has full row rank, since 1 is I ; 2, 3, 5 and 9 are $\mathbf{0}$; and 13, 14, 15 are nonzero matrices.

□

Theorem 5.1 $P_C(H_n^r)$ is full dimensional except $r = 3$. In this case, $\dim P_C(H_n^{r=3}) = \binom{n}{2}$.

Proof:

The theorem is proved by means of four cases.

i. $r > \frac{n}{2}$:

Choose a cut set S containing arbitrary r vertices. In this case, just one hyperedge is contained inside S . All other hyperedges should be in the cut since $|V \setminus S| = n - r < r$. That is, the complementary set $V \setminus S$ is so small in size that it cannot contain any hyperedge. The incidence vector of this cut has all 1s except the element that corresponds to the hyperedge in S . Since for all edges, one can find such a cut, and the corresponding incidence vectors are linearly independent, $P_C(H_n^r)$ is full dimensional.

ii. $4 \leq r \leq \lfloor \frac{n}{2} \rfloor$:

$P_C(H_n^r)$ is full dimensional by Proposition 3.8 and Proposition 3.3.

iii. $n = 2$:

The cut polytope $P_C(K_n)$ is full dimensional [44].

iv. $n = 3^1$:

Suppose that we have complete 3-uniform hypergraph of order n and a complete graph of order n . Consider an arbitrary three nodes, nodes 1, 2, and 3; and focus their induced subhypergraphs. We have a triangle in the subgraph, and a single hyperedge in the subhypergraph. Any cut in the triangle either no edges or exactly two edges. So $x_{12} + x_{13} + x_{23} = 0$ or 2. The same cut in the hypergraph includes the hyperedge in the subhypergraph or not, i.e., $y_{123} = 0$ or 1. Let us define a function f from $\mathbb{R}^{\binom{n}{2}} \mapsto \mathbb{R}^{\binom{n}{3}}$, which maps $P_C(K_n)$ to $P_C(H_n^3)$ such that $y_{123} = \frac{x_{12} + x_{13} + x_{23}}{2}$. The claim is f is one-to-one.

Let $y = 0 \Rightarrow x = 0$, for $n = 3$. Let $n \geq 4$. Pick arbitrarily 4 vertices, say h, i, j, k .

$$y = 0 \implies x_{ij} + x_{ik} + x_{jk} = x_{ij} + x_{ih} + x_{jh} \implies x_{ik} + x_{jk} = x_{ih} + x_{jh}.$$

Similarly,

$$x_{ih} + x_{ik} = x_{jh} + x_{jk}.$$

Subtracting the last equation from the previous one, we have

$$x_{jk} - x_{ih} = x_{ih} - x_{jk} \implies x_{jh} = x_{jk}.$$

So, $y = 0 \Rightarrow x = 0$. For any nonzero y , there must be a nonzero x .

In particular, let $y \neq 0$ and y defines a cut. This cut in complete 3-uniform hypergraph induces a cut set S . There is a $x \neq 0$ which induces the same cut. So, the vertices of $P_C(H_n^3)$ is mapped to $P_C(K_n)$.

Hence, f is one-to-one, and $P_C(H_n^3)$ is the image of $P_C(K_n)$ under a bijective linear transformation. Thus, $\dim P_C(H_n^3) = \dim P_C(K_n) = \binom{n}{2}$.

¹This part of the theorem is also proven by Deza and Laurent [31].

□

Corollary 5.1 *Let H be an r -uniform hypergraph with $r \neq 3$, then $P_C(H)$ is full dimensional.*

5.2 Complete Subhypergraph Inequalities

The corollary to Theorem 1.1 states that for $1 \leq s \leq n-1$,

$$\binom{n-1}{r-1} \leq C_{n,r}^s \leq \binom{n}{r} - \binom{\lfloor \frac{n}{2} \rfloor}{r} - \binom{\lceil \frac{n}{2} \rceil}{r}.$$

that is, if the cut $\delta(S)$ is nontrivial, i.e., $S \neq \emptyset$, $S \neq V$, its incidence vector $\mathcal{X}(\delta(S))$ should have at least $\binom{n-1}{r-1}$ 1s, and at most $\binom{n}{r} - \binom{\lfloor \frac{n}{2} \rfloor}{r} - \binom{\lceil \frac{n}{2} \rceil}{r}$ 1s. Since our polytope $P_C(H_n^r)$ is inside the unit cube, left hand side of above inequality cuts $2^{\binom{n-1}{r-1}} \approx 2^{\frac{\binom{n}{r}}{2}}$ integral vertices. Moreover, right hand side of the above inequality is facet defining when $r = \lfloor \frac{n}{2} \rfloor$ or $\lceil \frac{n}{2} \rceil$. The claim is trivial, and the proof lies in the proof of Theorem 1.1. Remember, $r = s$, the incidence matrix $\mathcal{X}(C_{H_n^r}^r)$ has full row rank, i.e., there are $\binom{n}{r}$ affinely independent vectors in the equality set. The following result is a direct consequence of the corollary which defines complete subhypergraph inequalities:

Theorem 5.2 *Let H_k^r be a complete subhypergraph of H . Then the complete subhypergraph inequality*

$$\sum_{e \in H_k^r} x_e \leq \binom{k}{r} - \binom{\lfloor \frac{k}{2} \rfloor}{r} - \binom{\lceil \frac{k}{2} \rceil}{r}$$

is valid for $P_C(H)$.

These inequalities are observed to be facet defining when k is odd, especially in case $k \geq \lfloor \frac{n}{2} \rfloor$. In particular, they are facet defining whenever $r = 3$:

Theorem 5.3 *Suppose k is odd. Then,*

$$\sum_{e \in H_k^3} x_e \leq \binom{k}{3} - \binom{\lfloor \frac{k}{2} \rfloor}{3} - \binom{\lceil \frac{k}{2} \rceil}{3}$$

is facet defining for $P_C(H_n^3)$.

Proof:

Consider the function f from $\mathbb{R}^{\binom{n}{2}} \mapsto \mathbb{R}^{\binom{n}{3}}$, which maps $P_C(K_n)$ to $P_C(H_n^3)$ such that $y_{123} = \frac{x_{12} + x_{13} + x_{23}}{2}$. The function f is one-to-one, and $P_C(H_n^3)$ is the image of $P_C(K_n)$ under a bijective linear transformation. Thus, facets of $P_C(K_n)$ are also facets of $P_C(H_n^3)$. Take the equality set of k -clique inequalities that define facets for $P_C(K_n)$.

$$\begin{aligned} \sum_{1 \leq i < j \leq k} x_{i,j} &= \lfloor \frac{k}{2} \rfloor \cdot \lceil \frac{k}{2} \rceil \\ &= \binom{k-1}{2} \cdot \binom{k+1}{2} = \binom{k-1}{3} \cdot \binom{3(k+1)}{4} = \binom{k-1}{3} \cdot \left(k - \frac{k-3}{4} \right) = \frac{k(k-1)}{3} - \frac{\binom{k-1}{2} \cdot \binom{k-3}{2}}{3} \\ &= \frac{k(k-1)}{3} - \frac{\lfloor \frac{k}{2} \rfloor (\lfloor \frac{k}{2} \rfloor - 1)}{3} = \frac{2}{k-2} \left[\frac{k(k-1)(k-2)}{6} - \frac{\lceil \frac{k}{2} \rceil \lfloor \frac{k}{2} \rfloor (\lfloor \frac{k}{2} \rfloor - 1)}{6} - \frac{\lfloor \frac{k}{2} \rfloor (\lfloor \frac{k}{2} \rfloor - 1) (\lfloor \frac{k}{2} \rfloor - 2)}{6} \right] \\ &= \frac{2}{k-2} \left[\binom{k}{3} - \binom{\lceil \frac{k}{2} \rceil}{3} - \binom{\lfloor \frac{k}{2} \rfloor}{3} \right] \iff \sum_{1 \leq i < j < l \leq k} y_{i,j,l} = \binom{k}{3} - \binom{\lceil \frac{k}{2} \rceil}{3} - \binom{\lfloor \frac{k}{2} \rfloor}{3}. \end{aligned}$$

Hence, the result follows. \square

5.3 Trivial Inequalities

The result of this small section is that the inequalities $x_i \leq 1$, $i \in H_n^r$ are facet defining for the complete r -uniform hypergraph cut polytopes, whereas $x_i \geq 0$, $i \in H_n^r$ are not.

Theorem 5.4 *The trivial inequality*

$$x_i \leq 1$$

is facet defining for $P_C(H_n^r)$.

Proof:

Let $F = \{x \in P_C(H_n^r) : x_i = 1\}$. Suppose there exists $bx \leq b_0$ be facet defining for $P_C(H_n^r)$ such that every point $x \in F$ satisfies $bx = b_0$. Since the canonical unit vector e_i lies in F , $be_i = b_i = b_0$. Let j be the index of another hyperedge. Then, $e_i + e_j \in F$, $j \neq i$. Hence, $e_i + e_j$ satisfies $bx = b_0$ which results in $b_j = 0$, $j \neq i$. Thus, $bx \leq b_0$ becomes $b_i x_i \leq b_0$. Since $0 \in P_C(H_n^r)$, $0 \leq b_0$. If $b_0 = 0$ then $bx \leq b_0$ will become meaningless. Thus, $b_0 > 0$. If we scale our last inequality by $\frac{1}{b_0}$, we will have the trivial inequality $x_i \leq 1$. Since $bx \leq b_0$ is facet defining for $P_C(H_n^r)$, so is $x_i \leq 1$. \square

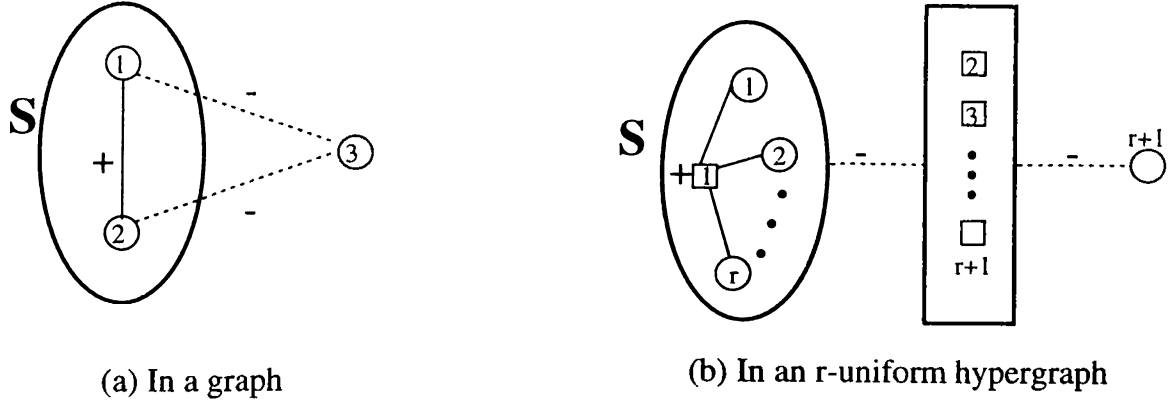


Figure 5.3: Situations defining homogeneous triangular inequalities.

solid lines whereas the negative valued edges are represented by dotted lines in the figure. The meaning of the inequality $x_{12} - x_{13} - x_{23} \leq 0$ is that, whenever the edge 12 is in any cut $\delta(A)$, at least one of the edges 13, 23 should also be in the cut $\delta(A)$.

Consider the situation illustrated in Figure 5.3(b). There are $r + 1$ nodes and $r + 1$ r -uniform hyperedges in the example subhypergraph. Nodes are divided into two sets: S and S^c . The size of S is r , and the size of the complementary part is less than r . The hyperedges in the cut will take a negative value and the hyperedge inside S will take a positive value in the valid inequality to be formed. Assume that the inequality is the following form:

$$\alpha \left(\sum_{e \text{ labelled } +} x_e \right) - \sum_{e \text{ labelled } -} x_e \leq 0$$

where α is a positive real number. The meaning of the above homogeneous inequality is that, whenever the hyperedge inside S is in a cut $\delta(A)$, at least α of the negative labeled hyperedges should also be in $\delta(A)$. Let

$$\alpha_{\delta(A)} = \frac{\alpha_{\delta(A)}^-}{\alpha_{\delta(A)}^+},$$

where $\alpha_{\delta(A)}^-$ is the number of negative hyperedges in $\delta(A)$, and $\alpha_{\delta(A)}^+$ is the number of positive hyperedges in the cut. The inequality defined above is defining a face

if

$$\alpha = \alpha_{min} \doteq \min_{\delta(A)} \alpha_{\delta(A)} = \frac{\alpha_{min}^-}{\alpha_{min}^+}.$$

In our case, there is just one (+)-labeled hyperedge, i.e., $\alpha_{\delta(A)}^+ = 1, \forall \delta(A) \Rightarrow \alpha = \alpha_{min}^-$. So, the minimum number of (-)-labeled hyperedges among all cuts $\delta(A)$ is required,

$$\alpha_{min} = \min_{1 \leq s \leq r} C_{r+1,r}^s - 1.$$

Corollary 4.2 leads

$$\alpha_{min} = C_{r+1,r}^1 - 1 = \binom{r+1-1}{r-1} - 1 = r - 1.$$

Hence, we have the following valid inequality:

$$(r-1)x_{12\dots r} - x_{12\dots(r-1)(r+1)} - \dots - x_{2\dots(r+1)} \leq 0.$$

If $r = 2$ the above inequality is reduces to $x_{12} - x_{13} - x_{23} \leq 0$. For each subset of $(r+1)$ nodes, $(r+1)$ inequalities of the above form can be constructed.

The following theorem generalizes the result found above. We have r nodes in the set S and l nodes in the complementary set.

Theorem 5.5 *Let $1 \leq l \leq r - 1$. Then the inequality*

$$\left[\binom{r+l-1}{r-1} - 1 \right] x_{12\dots r} - x_{12\dots(r-1)(r+1)} - \dots - x_{l+1\dots(r+l)} \leq 0.$$

is valid for $P_C(H_n^r)$.

Proof:

Let $S = \{1, \dots, r\}$ and $S^C = \{r+1, \dots, r+l\}$. So, $\gamma(S)$ contains only one hyperedge. Thus

$$\alpha_{\delta(A)}^+ = 1, \forall \delta(A) \Rightarrow \alpha_{min} = \alpha_{min}^- = \min_{\delta(A)} \alpha_{\delta(A)}^- = \min_{1 \leq s \leq r+l-1} C_{r+l,r}^s - 1.$$

By Corollary 4.2, $\alpha_{min} = \binom{r+l-1}{r-1} - 1$. □

Let the size of S and S^C be the same, r . Then $\alpha_{\delta(A)}^- = 1$ or 2 . The following theorem states the form of the valid inequality in this case.

Theorem 5.6 *Let $r \geq 3$. The inequality*

$$\left[\binom{2r-1}{r-1} + \binom{2r-2}{r-1} - 2 \right] (x_{12\dots r} + x_{(r+1)(r+2)\dots 2r}) - 2(x_{12\dots(r-1)(r+1)} - \dots - x_{r(r+2)\dots 2r}) \leq 0.$$

is valid for $P_C(H_n^r)$.

Proof:

We can divide the set of $\delta(A)$'s into two classes. Class I consists of the cuts that have only one internal hyperedges ($\alpha_{\delta(A)}^+ = 2$), and the other class, class II contains cuts having the two internal hyperedges ($\alpha_{\delta(A)}^+ = 1$). By Corollary 4.2, we have

$$\min_{\delta(A) \in I} \alpha_{\delta(A)} = C_{2r,r}^1 - 1,$$

and

$$\min_{\delta(A) \in II} \alpha_{\delta(A)} = \frac{C_{2r,r}^2 - 2}{2}.$$

The minimum is the minimum of the two:

$$\alpha_{min} = \min \left\{ C_{2r,r}^1 - 1, \frac{C_{2r,r}^2 - 2}{2} \right\}.$$

$$\frac{C_{2r,r}^2 - 2}{2} = \frac{\binom{2r-1}{r-1} + \binom{2r-2}{r-1}}{2} - 1 < \binom{2r-1}{r-1} - 1 = C_{2r,r}^1 - 1.$$

Thus,

$$\alpha_{min} = \frac{\binom{2r-1}{r-1} + \binom{2r-2}{r-1} - 2}{2}.$$

The inequality in the theorem has the following form:

$$\alpha_{min}^- \left(\sum_{e \text{ labelled } +} x_e \right) - \alpha_{min}^+ \sum_{e \text{ labelled } -} x_e \leq 0$$

□

Let the size of S^C is $r+1$ and keep the set S as it is. Then there are $r+1$ hyperedges inside S^C . Let $\alpha_{s_1 s_2}$ denote the value of $\alpha_{\delta(A)}$ when A has s_1 vertices in S and s_2 vertices in S^C .

Theorem 5.7 *Let $r \geq 3$. The inequality*

$$\left[\binom{2r}{r-1} - r \right] (x_{12\dots r} + x_{(r+1)(r+2)\dots 2r} + \dots + x_{(r+2)\dots(2r+1)}) - r(x_{12\dots(r-1)(r+1)} - \dots - x_{r(r+3)\dots 2r+1}) \leq 0.$$

is valid for $P_C(H_n^r)$.

Proof:

Since $\frac{\binom{2r}{r-1}}{r} < \binom{2r}{r-1}$, $\alpha_{01} < \alpha_{10}$. Moreover, $\alpha_{11} = \alpha_{02} = \frac{C_{2r+1, r-r-1}^2}{r+1}$. Furthermore, $\alpha_{0s} > \alpha_{1(s-1)}$, and $\alpha_{1s} > \alpha_{1(s-1)}$, $s \geq 3$.

$$\begin{aligned} \alpha_{01} \text{ vs } \alpha_{02} &\Leftrightarrow 1 + \alpha_{01} \text{ vs } 1 + \alpha_{02} \\ \Leftrightarrow \frac{\binom{2r+1-1}{r-1}}{r} \text{ vs } \frac{\binom{2r+1-1}{r-1} + \binom{2r+1-2}{r-1}}{r+1} &\Leftrightarrow (r+1) \binom{2r}{r-1} \text{ vs } r \left[\binom{2r}{r-1} + \binom{2r-1}{r-1} \right] \\ &\Leftrightarrow \binom{2r}{r-1} \text{ vs } r \binom{2r-1}{r-1} \Leftrightarrow \frac{2}{r+1} \text{ vs } 1 \implies \alpha_{01} < \alpha_{02} \end{aligned}$$

$$\begin{aligned} \alpha_{01} \text{ vs } \alpha_{12} &\Leftrightarrow 1 + \alpha_{01} \text{ vs } 1 + \alpha_{12} \\ \Leftrightarrow \frac{\binom{2r+1-1}{r-1}}{r} \text{ vs } \frac{\binom{2r+1-1}{r-1} + \binom{2r+1-2}{r-1} + \binom{2r+1-3}{r-1} - \binom{3}{r}}{r+1}. \end{aligned}$$

Now, assume that $r > 3$, then

$$\begin{aligned} \alpha_{01} \text{ vs } \alpha_{12} &\Leftrightarrow \frac{\binom{2r+1-1}{r-1}}{r} \text{ vs } \frac{\binom{2r}{r-1} + \binom{2r-1}{r-1} + \binom{2r-2}{r-1}}{r+1} \\ &\Leftrightarrow (r+1) \binom{2r}{r-1} \text{ vs } r \left[\binom{2r}{r-1} + \binom{2r-1}{r-1} + \binom{2r-2}{r-1} \right] \\ &\Leftrightarrow 2 \binom{2r}{r-1} \text{ vs } r \left[\binom{2r-1}{r-1} + \binom{2r-1}{r-1} \right] \\ &\Leftrightarrow \frac{2}{r+1} + \frac{4r-2}{r(r+1)} \text{ vs } 1 + 1 \implies \alpha_{01} < \alpha_{12} \end{aligned}$$

For $r = 3 \Rightarrow 1 + \alpha_{01} = 5 < 6 = 1 + \alpha_{12}$.

Thus,

$$\alpha_{\min} = \alpha_{01} = \frac{\binom{2r}{r-1} - r}{r}.$$

□

The above inequalities can be generalized further into clique-clique inequalities. Suppose that a complete r -uniform subhypergraph of order $k > r$ is taken such that the k vertices is divided into two sets of sizes $k_1 > 0$ and $k_2 > 0$. In the clique-clique inequality, the hyperedges inside sets K_1 and K_2 have $+$ labels whereas the hyperedges in the cut set $\delta(K_1; K_2)$ have $-$ labels. Then the inequality has the following form:

$$\alpha_{min}^-(x[\gamma(K_1)] + x[\gamma(K_2)]) - \alpha_{min}^+(x[\delta(K_1; K_2)]) \leq 0$$

Let, $\alpha_{min} = \frac{\alpha_{min}^-}{\alpha_{min}^+}$. We conjecture the following:

Conjecture 5.1 *Clique-clique inequalities*

$$\alpha_{min}^-(x[\gamma(K_1)] + x[\gamma(K_2)]) - \alpha_{min}^+(x[\delta(K_1; K_2)]) \leq 0$$

are valid for $P_C(H)$ when

$$\alpha_{min} = \begin{cases} \alpha_{10} & k_1 > k_2, \\ \alpha_{11} & k_1 = k_2, \\ \alpha_{01} & k_1 < k_2. \end{cases}$$

5.5 Congestion Inequalities

Let us consider an $r \geq 3$ uniform complete graph of order $r + 1$ whose edges are numbered lexicographically. If we label the first two hyperedges with $+1$ and the remaining ones with -1 , we will have the following inequalities:

$$\alpha_1 \leq x_1 + x_2 - x_3 - \cdots - x_{r+1} \leq \beta_1$$

The situation is illustrated in Figure 5.4. The $+1$ labeled hyperedges are indicated by solid lines whereas the hyperedges with -1 labels are shown by dashed lines. There are only three different cut types as indicated in the figure. Then $\alpha_1 = -(r - 2)$ and $\beta_1 = \max\{0, -(r - 4)\}$.

Similarly, if we consider a complete subhypergraph of order $r + 2$ and label the first three hyperedges with $+1$ and the others with -1 , we will have

$$\alpha_2 \leq x_1 + x_2 + x_3 - x_4 - \cdots - x_{0.5(r+2)(r+1)} \leq \beta_2.$$

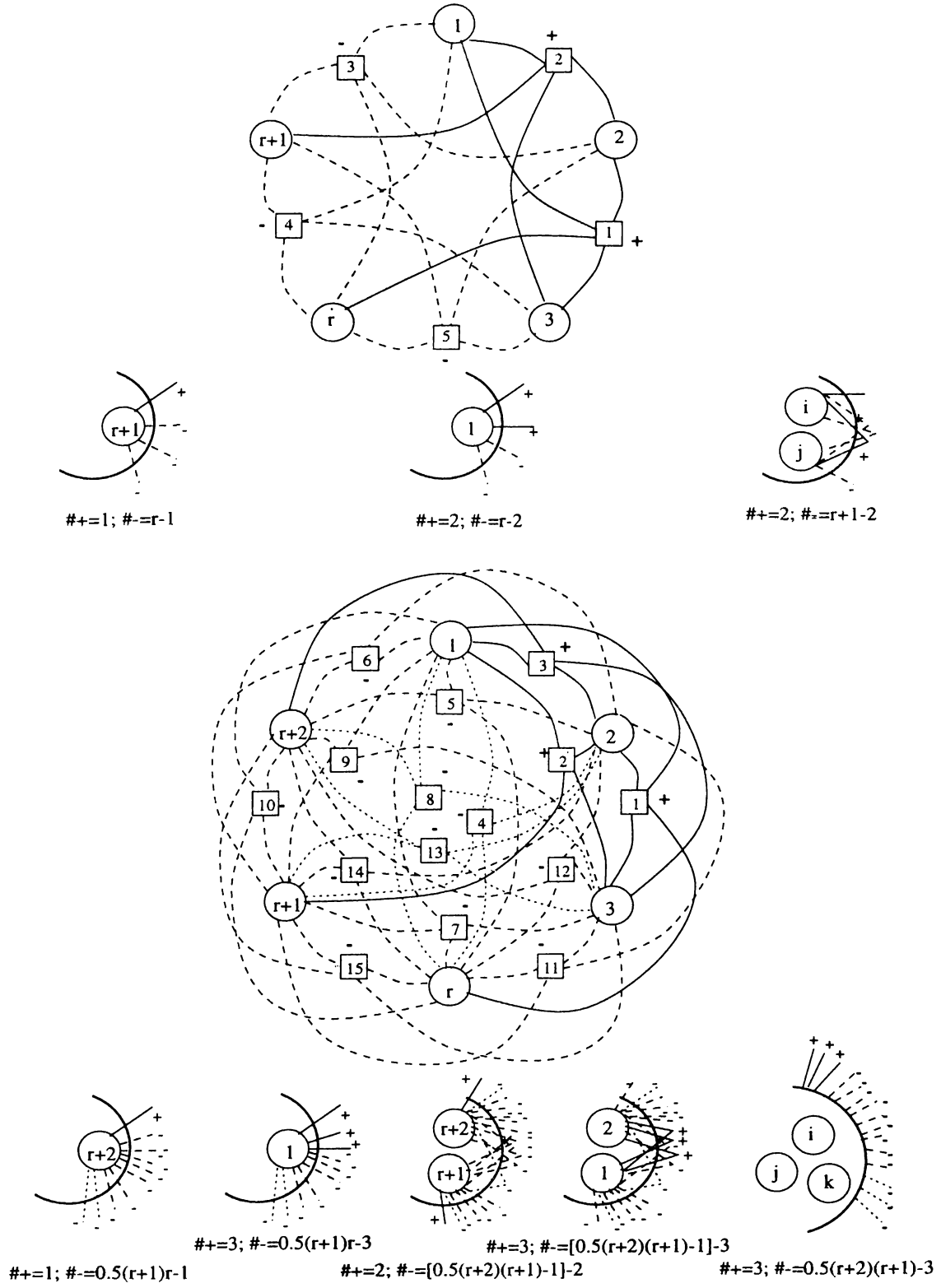


Figure 5.4: Congestion inequalities, $r = 4$.

This situation is also illustrated in Figure 5.4. In this case, we have five different cut situations which results in $\alpha_2 = 5 - 0.5(r + 2)(r + 1)$ and $\beta_2 = 0$. Hence, we have proved the following theorem:

Theorem 5.8

- i. Let D_1 be a complete r uniform subhypergraph of order $r + 1$ of a hypergraph of order n , where $3 \leq r < n$. For any numbering of hyperedges of D_1 between 1 and $r + 1$, the inequalities

$$-(r - 2) \leq x_1 + x_2 - x_3 - \cdots - x_{r+1} \leq \max\{0, -(r - 4)\}$$

are valid for $P_C(H)$.

- ii. Let D_2 be a complete r uniform subhypergraph of order $r + 2$ of a hypergraph of order n , where $3 \leq r < n - 1$. For any numbering of hyperedges of D_2 between 1 and $\binom{r+1}{r}$, the inequalities

$$5 - \binom{r+2}{r} \leq x_1 + x_2 + x_3 - x_4 - \cdots - x_{\binom{r+2}{r}} \leq 0$$

are valid for $P_C(H)$.

Corollary 5.2

- i. Let D_1 be a complete r uniform subhypergraph of order $r + 1$ of a hypergraph of order n , where $2 \leq r < n$. For any numbering of hyperedges of D_1 between 1 and $r + 1$, the inequalities

$$[r - 2](x_1 + x_2) - 2(x_3 + \cdots + x_{r+1}) \leq 0$$

$$[r - 1](x_1 + x_2) - x_3 - \cdots - x_{r+1} \geq 0$$

are valid for $P_C(H)$.

- ii. Let D_2 be a complete r uniform subhypergraph of order $r + 2$ of a hypergraph of order n , where $2 \leq r < n - 1$. For any numbering of hyperedges of D_2 between 1 and $\binom{r+1}{r}$, the inequalities

$$\left[\binom{r+2}{r} - 3 \right] (x_1 + x_2 + x_3) - 3(x_4 + \cdots + x_{\binom{r+2}{r}}) \leq 0$$

$$\left[\binom{r+2}{r} - 1 \right] (x_1 + x_2 + x_3) - x_4 - \dots - x_{\binom{r+2}{r}} \geq 0$$

are valid for $P_C(H)$.

5.6 Computational Results

Our code used in this section solves two \mathcal{NP} - Complete problems, maximum cut (max-cut) and bipartitioning (min-cut), on complete r-uniform hypergraphs. We report about computational results on random problems. The test set for each problem consists of 40 complete 3-uniform and 4-uniform hypergraphs of orders 10 and 15. The size of test problems seems to be small at the first glance. However, typical values for the case of complete graphs are 20, 30 up to 40. There were $\binom{40}{2}=780$ edges in the case of $n=40$. In our case, for instance $n=15$, there are $\binom{15}{3}=455$ and $\binom{15}{4}=1365$ hyperedges. The edge costs are drawn from a uniform distribution of integers between 1 and 999. The experiments are done at a 22 MIPS Sun system and Cplex 3.0 is used.

The set of constraints that are considered are tabulated in Table 5.1. We have

Inequality Type	n=10 Subclass	r=3 Total #	n=10 Subclass	r=4 Total #	n=15 Subclass	r=3 Total #	n=15 Subclass	r=4 Total #	ST.
Complete Sub-hypergraph	k=5	252	k=7	120	k=5	3003	k=7	6435	I
	k=9	10	k=9	10	k=14	15	k=14	15	I
	k=7	120			k=7	6435	k=9	5005	II
					k=9	5005	k=11	1365	II
					k=11	1365	k=13	105	I
Generalized Triangle	l=4	840	l=5	1260	l=4	5460	l=5	15015	I
	l=5	2520	l=6	3150	l=5	♠ 30030	l=6	♠ 75075	II
Congestion	l=4	2520	l=5	5040	l=4	16380	l=5	♠ 60060	II
	l=5	5040	l=6	8400	l=5	♠ 60060	l=6	♠ 200200	II
Trivial		120		210		455		1365	I
$\sum_{e \in H_r^n} x_e \leq \binom{n}{r} - \binom{\lfloor \frac{n}{2} \rfloor}{r} - \binom{\lfloor \frac{n}{2} \rfloor}{r}$									I
$\sum_{e \in H_r^n} x_e \geq \binom{n-1}{r-1}$ is added to force a cut in bipartitioning.									I

♠: These constraints are added up to the limit of Cplex!

Table 5.1: The stages of computation.

designed three stages. The first stage contains easy-to-list constraints. A quick solution is obtained immediately after solving these constraints. In the second stage, there are lots of constraints that are checked whether they are satisfied by the solution found in the previous run. If an integer solution is not found at the end of stage II, we resort to branch and bound. The initial bound is obtained from a simple clustering using similarity coefficients followed by an interchange routine across the cut of clustering.

Two different Cplex routines are compared in this computational study. The first routine is a primal simplex routine. The second routine is based on an interior point method, called barrier routine. For the details of these routines, the interested reader is referred to [28].

In each problem set, the mean, the minimum and the maximum values for the CPU seconds used by the random problems are reported. These are net CPU values obtained from the "sys/times" routine. Furthermore, each test set are divided into two subcases in each stage according to the solution characteristics: integer and fractional. The CPU times are reported for separately each subcase together with the number of problems that are in the subcase. The fractional problems are fed into the next stage, and so on. In the third stage, branch and bound is performed by using the primal method only.

The results for the both problems are presented in Table 5.2. We have obtained integral maximum cut solutions of the 146 test problems out of 160 by using the inequalities listed in this study. The percentage integer solutions obtained before branch and bound is 91.25 which is quite high. At the end of the first stage, more than 80 percent the test problems have integer solutions. By adding the inequalities in the second stage, we have obtained 13 integer solutions out of 27 problems which is approximately a half. The results for the minimum cut problem are as follows. The number of integer solutions before the last stage is 122, yielding a percentage of 76.25. This amount is less than that of maximum cut problem. Therefore, we need more powerful valid inequalities investigated especially for solving the minimum hypergraph bipartitioning. The success ratio

STAGE	CASE	NO	PRIMAL CPLEX			BARRIER CPLEX		
			Mean	Min	Max	Mean	Min	Max
Problem: Max-Cut								
ONE	10,3 Integer	37	25.3198	17.5500	55.0167	618.5275	587.9334	716.0167
	10,3 Fractional	3	35.5111	26.1667	28.3833	631.6333	589.4334	653.8000
	10,4 Integer	30	81.3272	51.0167	139.7500	1182.1767	1061.6667	1409.8500
	10,4 Fractional	10	57.4150	44.0833	105.0667	1132.1072	1061.8667	1178.4667
	15,3 Integer	35	1249.8766	1039.6667	1596.1833	28197.6867	24305.1881	34196.7333
	15,3 Fractional	5	1116.2267	980.4000	1287.4500	34581.3275	26943.6667	39174.3333
	15,4 Integer	31	4840.8924	4217.1933	5314.1667	Not Enough Memory!		
	15,4 Fractional	9	3385.8876	2819.7333	3966.1200	Not Enough Memory!		
TWO	10,3 Integer	1	32.5998	32.5998	32.5998	1722.1667	1722.1667	1722.1667
	10,3 Fractional	2	30.6251	30.4834	30.7667	1690.7058	1514.1919	1867.2197
	10,4 Integer	4	68.2000	50.3500	100.7333	4714.1107	4279.6488	5481.1763
	10,4 Fractional	6	80.3639	52.7167	120.9167	4966.2364	4007.1111	5814.6782
	15,3 Integer	4	1509.6083	1431.1333	1645.6333	39967.5687	31988.4667	43109.4983
	15,3 Fractional	1	1430.4166	1430.4166	1430.4166	30987.5614	30987.5614	30987.5614
	15,4 Integer	4	8718.3333	8109.6700	9412.4411	Not Enough Memory!		
	15,4 Fractional	5	8964.6714	8411.9786	9733.4554	Not Enough Memory!		
THREE	10,3 Integer	2	105.8995	97.6667	114.1322	Primal Cplex is used!		
	10,4 Integer	6	214.1322	196.1967	227.6645	Primal Cplex is used!		
	15,3 Integer	1	2412.6667	2412.6667	2412.6667	Primal Cplex is used!		
	15,4 Integer	5	3117.1895	1227.6667	5414.1969	Primal Cplex is used!		
Problem: Min Cut								
ONE	10,3 Integer	25	23.7381	20.1823	28.5833	622.1871	583.4051	681.7293
	10,3 Fractional	15	22.0667	17.5167	30.4833	628.3202	577.9456	679.6387
	10,4 Integer	27	77.1254	44.2334	86.9864	1174.1971	1064.3718	1512.5464
	10,4 Fractional	13	61.1428	39.2916	87.1197	1166.8202	1060.6409	1473.6667
	15,3 Integer	25	987.1924	896.9422	1044.5501	29147.6504	16067.3333	36398.1142
	15,3 Fractional	15	953.8667	874.1369	1107.3600	25914.2363	12073.8667	39780.4343
	15,4 Integer	26	4514.7794	4109.1411	5096.1778	Not Enough Memory!		
	15,4 Fractional	14	4689.6882	3987.7486	4909.1881	Not Enough Memory!		
TWO	10,3 Integer	5	35.1494	31.0086	42.4492	1937.1990	1109.2524	2436.3186
	10,3 Fractional	10	38.1966	30.9871	47.1569	1956.8014	1341.6737	2634.4567
	10,4 Integer	4	71.0801	60.0704	87.0203	5410.4535	3907.2798	6988.0207
	10,4 Fractional	9	76.9743	67.1927	84.3476	5129.3607	4414.1867	5716.1167
	15,3 Integer	7	919.1324	881.4715	1096.8167	27616.5129	16372.4033	57241.1178
	15,3 Fractional	8	1106.8767	918.1335	1307.4921	27442.8262	19101.1540	44832.9451
	15,4 Integer	3	8419.1695	6518.2470	9711.2742	Not Enough Memory!		
	15,4 Fractional	11	7867.8766	7105.2460	8801.3967	Not Enough Memory!		
THREE	10,3 Integer	10	307.1766	37.1000	567.1922	Primal Cplex is used!		
	10,4 Integer	9	1918.7295	43.6667	3386.9197	Primal Cplex is used!		
	15,3 Integer	8	2526.0314	127.1763	5419.8167	Primal Cplex is used!		
	15,4 Integer	11	7843.4253	819.2161	11075.9150	Primal Cplex is used!		

Table 5.2: The results of the computational study on $P_C(H_n^r)$ for $n=10,15$ and

during the first stage is $\frac{103}{160}$, which is approximately 65 percent. Among the remaining problems, we have found 19 integer solutions out of 57 problems in stage IV, which is exactly one thirds. The CPU times remain almost unchanged, if the objective function is switched from maximization to the minimization.

The simplex method outperforms the barrier method in terms of speed. As the number of constraints is increased, the barrier method's CPU usage jumps up. As the number of variables is increased, its time scores get worse. The behavior of the primal simplex method against these changes is found to be relatively tolerable.

Chapter 6

CELL FORMATION PROBLEM

Cell formation i.e., placing machine groups of functionally dissimilar types together to enable the manufacture of a specific range of parts, is acknowledged to be the first and major stage in the design of Cellular Manufacturing systems. The initial decision made at this stage presides over all the other decisions involved in the design process.

Since it was asserted for the first time (Burbidge 1971) [19], the cell formation problem has grown into an area in which much research has been conducted. As the cell formation problem turns out to be \mathcal{NP} -Complete [4, 73], a large number of cell formation heuristics have been designed to obtain feasible solutions. Various taxonomies of the cell formation techniques have also been proposed [5, 59, 65, 96].

In this chapter, two new cell formation techniques are developed. One is based on the special case of the Boolean R-atic Polytope which is proved to be polynomially solvable in Section 4.6. The other technique is based on approximating the hypergraph with a graph and breaking this graph into pieces by successively solving maximum flow problems. These techniques together with six leading cell formation techniques are compared in terms of developed measures in different manufacturing environments so as to help establish guidelines to select a proper cell formation procedure for a specific situation. A similar study has

been reported by Miltenburg and Zhang [79].

Three ‘quick & clean’ efficiency indices to evaluate the cell formation solutions are suggested in the next section. The first index is the grouping efficiency which is focused on the combination of the magnitude of exceptional elements and inner-cell densities. The second is related to inner-cell work-load balances. The last index measures the under-utilizations of individual machines.

Finally, to provide a testing ground for evaluating and comparing design techniques, a problem generator is developed. The above mentioned techniques are evaluated in terms of randomly generated test problems under different scenarios. The effects on the proposed efficiency measures from the number of parts, shop densities, demand and depreciation cost variations, and manufacturing environments on the cell formation problems are investigated statistically for each technique.

6.1 Hypergraph Representation

Cell formation problem is defined via hypergraphs such that the machine types are represented by vertices of its associated hypergraph, and hyperedges represent parts; or vice versa. A hyperedge is incident to a node in the hypergraph representation if there is a routing relationship between the corresponding part and machine type pair. Moreover, various weights can be tagged in the hypergraph representation of the situation. Machine types differ from each other by their cost values. Parts are different in their unit profits. The major drawback of hypergraph representation is to neglect process sequences.

Cell formation problem can be defined as the minimum hypergraph free partitioning problem when the nodes of the associated hypergraph is machine types. If the number of cells is fixed and the lower and/or upper cell size limits are imposed, the definition is termed as multiple partitioning. The special case of multiple partitioning where there are only two cells to be formed is called bipartitioning. The additional restriction of balanced cell sizes defines bisection problem. Moreover, there is a variant of multiple partitioning problem in which

Part	Machine set (Machine fraction)	Profit	Part	Machine set (Machine fraction)	Profit
1	1(.36), 2(.34), 3(.77)	100	8	4(.24), 9(.74), 10(.63)	50
2	2(.76), 3(.15)	50	9	2(.24), 4(.42), 5(.32), 9(.52)	25
3	2(.16), 3(.16), 7(.78)	25	10	6(.09), 7(.46)	75
4	7(.45), 10(.46)	75	11	4(.18), 5(.40), 6(.44)	125
5	7(.73), 8(.60), 9(.96)	125	12	4(.26), 6(.29)	100
6	7(.52), 8(.66), 9(.54), 10(.77)	150	13	4(.12), 5(.20), 6(.24)	50
7	7(.30), 9(.34), 10(.46)	100	14	1(.57), 2(.70), 6(.15)	25

Table 6.1: Part information of the example problem.

Machine	Usage	Number	Cost	Machine	Usage	Number	Cost
1	0.93	1	33	6	1.21	2	143
2	2.20	3	80	7	3.24	4	91
3	1.08	2	111	8	1.26	2	77
4	1.22	2	100	9	3.10	4	50
5	0.92	1	200	10	2.32	3	71

Table 6.2: Machine type information of the example problem.

number of cells are not fixed, called free partitioning (cell formation) problem. The latter formulation is termed as separation problem if the nodes are parts and hyperedges represent machine types. Not surprisingly, the cell formation problem defined as free partitioning is \mathcal{NP} -Hard. In fact, multiple partitioning is \mathcal{NP} -Hard. Actually, the hypergraph bipartition problem belongs to \mathcal{NP} -Hard class [74].

The manufacturing situation tabulated in Table 6.1 is taken as an example throughout the discussions. We have ten machine types and fourteen parts. Machine type set together with machine fractions, and unit profit as relative difference are associated for each part. Machine fraction value for a machine-part pair is the percentage of annual usage of that machine for that part. Furthermore, the number of physical machines and relative machine differences (based on machine type 4) in annual depreciations, or buying prices, or salvage values, etc. are given in Table 6.2. The number of physical machines of a given type is calculated as the smallest integer that is greater than the total usage of that

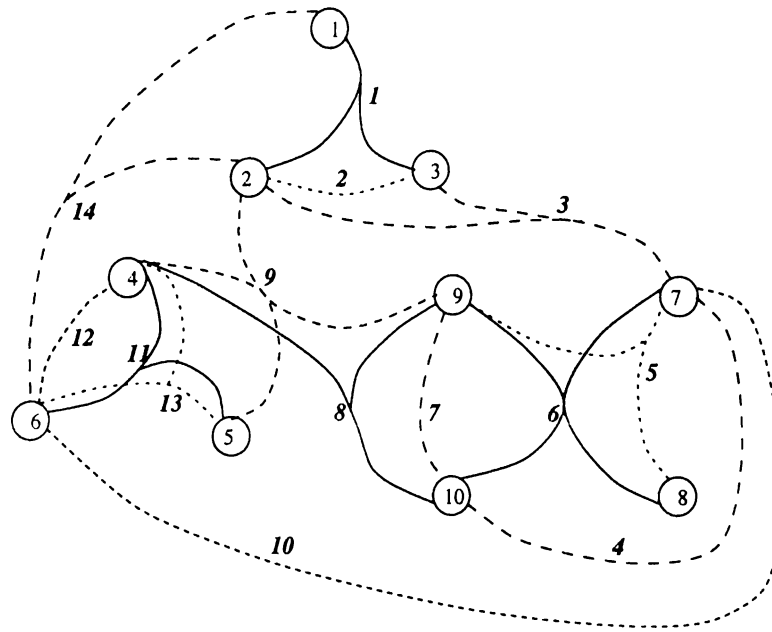


Figure 6.1: The associated hypergraph of the example cell formation situation.

machine type which is the sum of machine fractions over all parts. The associated hypergraph having vertices as machine types is illustrated in Figure 6.1.

6.2 A Generic Cell Formation Algorithm

The cell formation method proposed in this section is to approximate the hypergraph by graphs so that the cuts are less affected by the approximation. Then a Gomory-Hu cut tree of the graph approximation is obtained. Using this tree, the minimum cuts between all pair of vertices are calculated easily and partition tree is produced. Finally, the cell formation heuristic based on the partition tree is stated.

6.2.1 Hypergraph Approximation

The first step of the method used is approximating the hypergraph representation by a multi-graph in which there can be more than one edge among two vertices. The vertices of the hypergraph and the graph representation coincide. There

are two ways of representing hyperedges. One method is to use a clique of the end-points of a hyperedge. Hence, if two vertices are end-points of more than one hyperedge, there will be parallel edges connecting these vertices in the graph approximation. Consider Figure 6.2-a. The example hyperedge, 6, connects vertices 7, 8, 9, and 10. The clique approximation of hyperedge 6 is given in Figure 6.2-b. The second way to represent a hyperedge is to add a superficial node and use a *star* graph having center at this superficial node. A star graph is the graph where all edges connects outer nodes to the center node, like an asterix. The star approximation of hyperedge 6 is illustrated in Figure 6.2-c.

The most important issue in graph approximation is to assign suitable weights to graph edges in such a way that the value of a cut in the hypergraph is almost equal to the value of the cut in the graph approximation which is defined by the same vertices. Another issue is to interpret the approximation in cell formation terms. The proposed solutions for two representation methods are explained one by one.

After assuming that the weight of the hyperedge under study is one, a straightforward way of solving the above problem for clique approximation is to assign edge weights as less than one and to try to obtain unity as the value of the cut in the clique. Consequently, the weight of every edge in the clique is multiplied with the weight of the hyperedge to be approximated by the clique.

There are several attempts to assign weights to clique edges. Charney and Plato [23] suggested $\frac{2}{n-1}$, for a hyperedge having n incident vertices; whereas Hanan and Kurtzberg [51] offered $\frac{2}{n}$. In [35], Donath presented a proof that better results can be obtained by choosing the weight $\frac{4}{n^2 - n \bmod 2}$. Hadley [48]

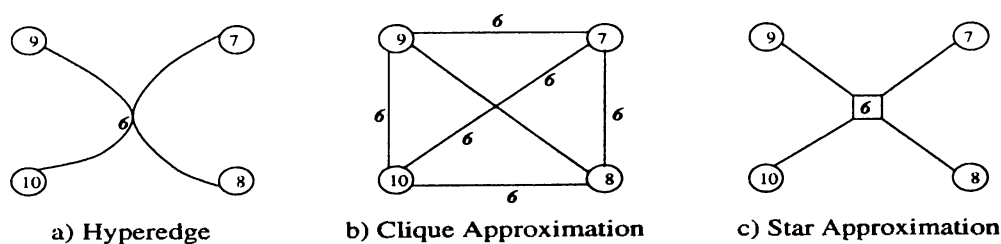


Figure 6.2: The two representations.

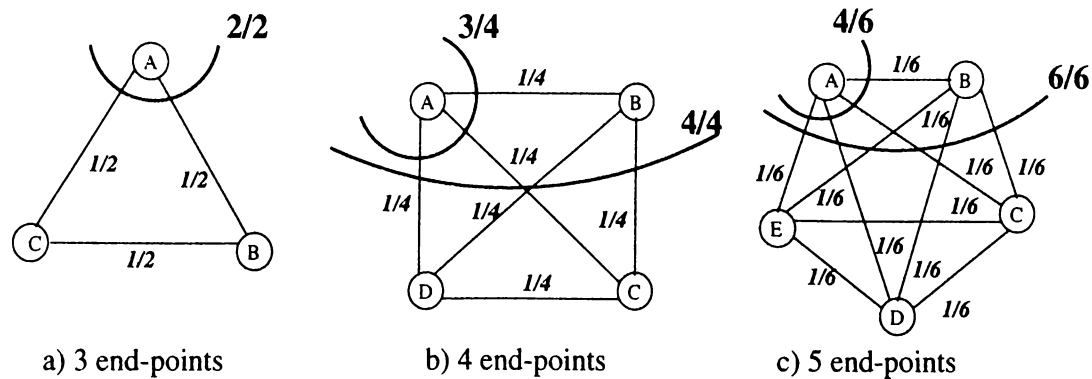


Figure 6.3: Donath - Hadley approximation of the hyperedges in various sizes.

proposed the weights $(\lceil \frac{n}{2} \rceil \times \lfloor \frac{n}{2} \rfloor)^{-1}$, where $\lceil t \rceil$ ($\lfloor t \rfloor$) is the smallest (largest) integer greater (less) than or equal to t . Hadley discussed the optimality of these weights of the clique edges in order to approximate unity as the value of any cut in the clique in various norms (l_1 , l_2 , and l_∞). Actually, Hadley and Donath figured the same weights. The clique approximation using their weights is illustrated for hyperedges having 3, 4, 5 vertices in Figure 6.3. In this figure, the possible cuts and their values are shown. Clearly, this approximation underestimates the cuts in hypergraphs.

The following method is used to approximate the hypergraph representing a cell formation situation: approximate each hyperedge by the clique using Donath-Hadley weights. Weight each clique (part) by its associated unit profit value. Replace all parallel edges belonging to different cliques by single edges having weights equal to the total weights of the parallel edges. The graph scaled by 4 for the clique approximation of the hypergraph given in Figure 6.1 is illustrated in Figure 6.4. Consider edge (7,9) having value of 600. It has a weight of $1/2 \cdot 125$ from part 5, $1/4 \cdot 150$ from part 6, and $1/2 \cdot 100$ for part 7, having a total value of $600/4$. Since the graph is scaled by 4, the weight of edge (7,9) in the figure is 600.

A different method is proposed for star representation of a cell formation situation. A more direct method of labeling the star edges is used. Each edge in a star approximation of a hyperedge describes a relationship between the corresponding

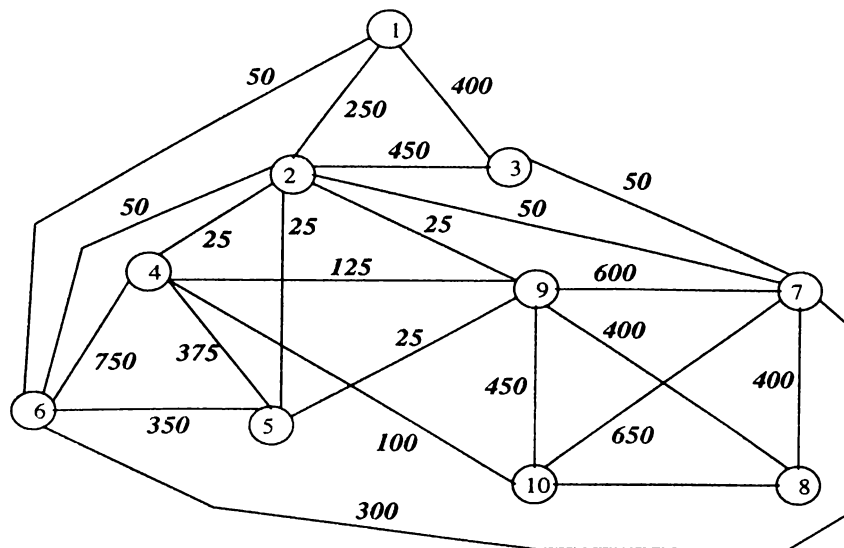


Figure 6.4: The clique approximation of the example hypergraph.

machine type – part pair. This relationship is simply the machine fraction due to the part under consideration. If a star graph is used for each part, then the graph approximation of the whole cell formation situation is actually a bipartite graph having edge weights as machine fractions. Considering the difference between machine types, all edges emanating from a node representing a machine type are weighed by its relative value in terms of annual depreciation, salvage value, initial cost, etc. The star approximation of our example cell formation problem is illustrated in Figure 6.5. The cost of the edge between machine type 3 and part 2 is 17 ($=0.16 \times 108$).

6.2.2 Gomory - Hu Cut Tree

A network flow based method based on the maximum flow - minimum cut theorem is used in this heuristic attempt to solve cell formation problem. The minimum cuts values for all machine type pairs is needed for stating the heuristic method proposed. A naive way of finding the cuts is to solve a maximum flow problem for every pair. This algorithm requires the solution of $\frac{n(n-1)}{2}$ maximum flow problems assuming there are n machine types. However if Gomory-Hu algorithm is applied

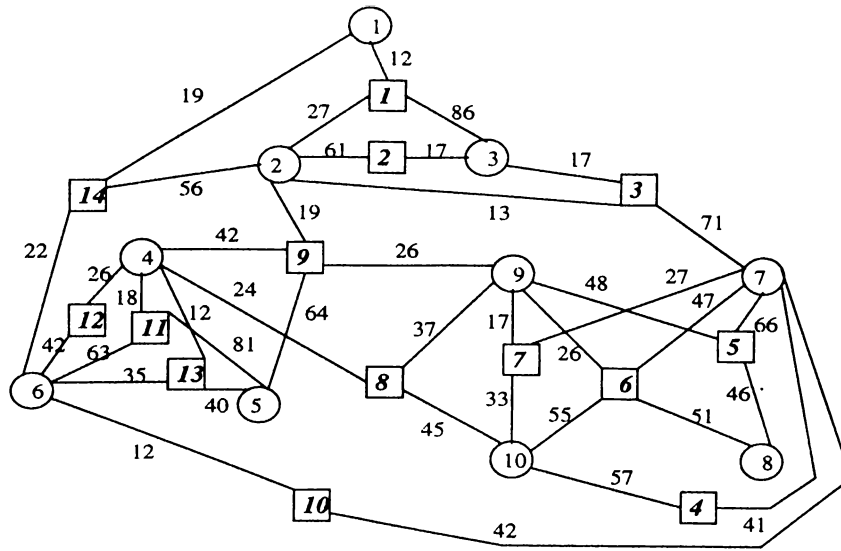


Figure 6.5: The star approximation of the example hypergraph.

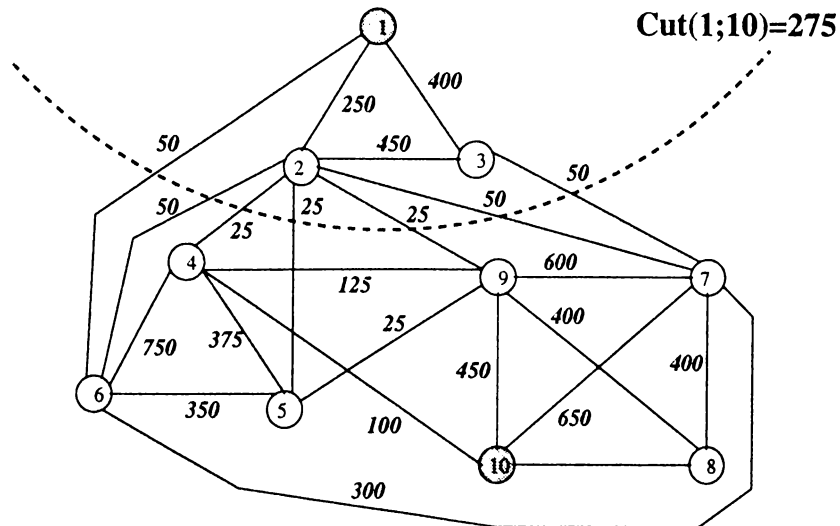


Figure 6.6: First iteration of Gomory-Hu algorithm.

[41] at most $(n-1)$ maximum flow calculations are required for any subset of the n machine types.

A brief description of Gomory-Hu algorithm is given on the clique approximation of the example problem. First select two terminal nodes arbitrarily, say 1 and 10, and do a maximum flow computation on the original graph. This gives

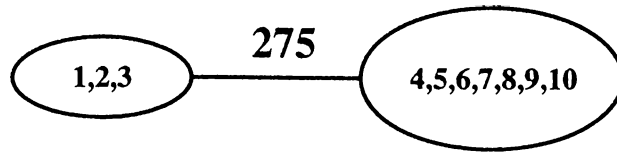


Figure 6.7: First link of tree diagram.

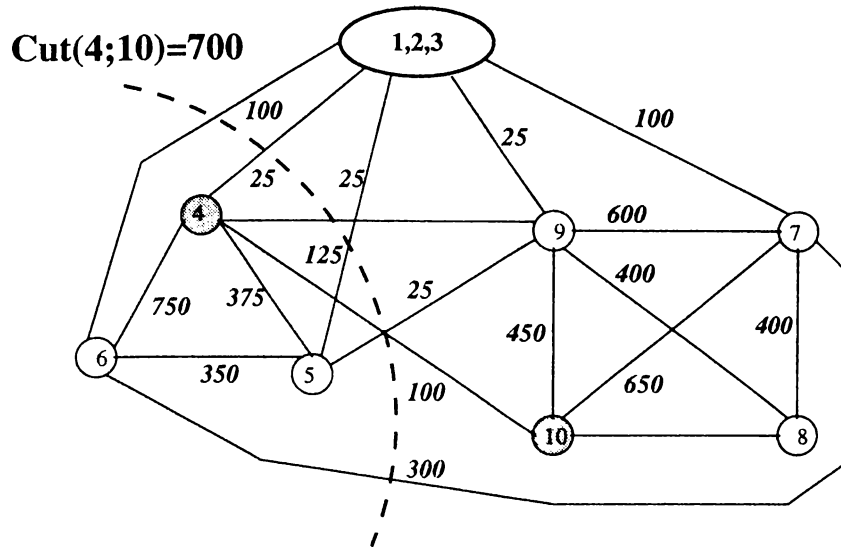


Figure 6.8: Second iteration of Gomory-Hu algorithm.

a minimum cut $C_{1,10}$ as indicated in Figure 6.6. This cut is symbolized by two circles connected by a link having the value of the cut as in Figure 6.7. This network is termed as *tree diagram*.

Second, from the tree diagram obtained so far, select any circle, say $(4,5,6,7,8,9,10)$, which contains more than two nodes, and do a maximum flow computation between the two terminal nodes $(4,10)$ on a graph derived from the original graph in which the nodes $(1,2,3)$ in the unselected circle in the tree diagram is shrunk into a single node. This derived graph is flow-equivalent to the original graph in finding the maximum flow between 4 and 10. This maximum flow computation gives another minimum cut $C_{4,10}$ as illustrated in Figure 6.8. This is also represented symbolically as in Figure 6.9. Note that $(4,5,6)$ is attached to $(7,8,9,10)$ since $(1,2,3)$ and $(7,8,9,10)$ are on the same side of $C_{4,10}$ with value 700.

After $(n-1=9)$ maximum flow computations, a tree diagram is obtained in

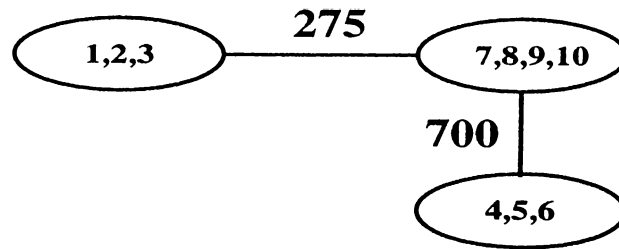


Figure 6.9: Second link of tree diagram.

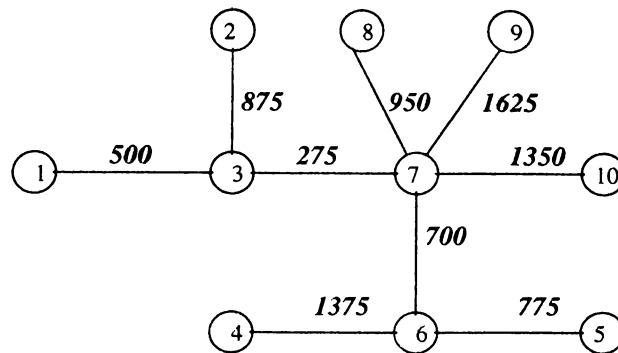


Figure 6.10: A Gomory-Hu cut tree of the clique approximation of the example problem.

which each vertex in the tree diagram is a node in the original graph as presented in Figure 6.10. Note that, the maximum flow computations are being done on graphs that are progressively simple than the original graph due to the condensation of nodes. The final form of the tree diagram is called a *Gomory-Hu* cut tree. The original graph is flow-equivalent to the Gomory-Hu cut tree.

A Gomory-Hu cut tree of the star representation of the example problem is given in Figure 6.11. In this cut tree, the costs of the edges incident to circle nodes representing parts can be ignored during bipartitioning the squared node set corresponding to machine types. However, their connection should be kept for part assignments to the cells defined by the partition. The clique approximation is inferior in this sense since it requires a part assignment scheme. A trivial scheme is to assign a certain part to the cell among the possible alternatives in which the part's workload is the most. Moreover, if there are m machine types

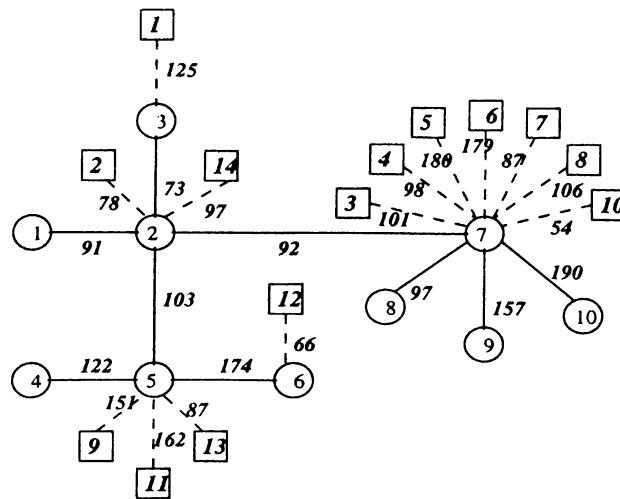


Figure 6.11: A Gomory-Hu cut tree of the star approximation of the example problem.

and p parts, the clique approximation's cut tree is obtained by $n-1$ maximum flow computations on a network of order m . On the other hand, the cut tree of star representation requires $m+p-1$ maximum flow computations on a network that have $m+p$ vertices. Therefore, star representation is inferior in terms of computational complexity.

6.2.3 Generic Algorithm

In this subsection, a generic algorithm is stated. The name of the algorithm is *Hypergraph Approximation - Cut Tree* (HAP-CUT). The algorithm HAP-CUT inputs the cell formation situation, the representation method, and three other parameters: a criterion to measure the quality of the solution, and upper limit on the sizes of parts in the bipartition. The limit can be such that the parts cannot be larger than a prespecified percentage of the size of the set to be bipartitioned.

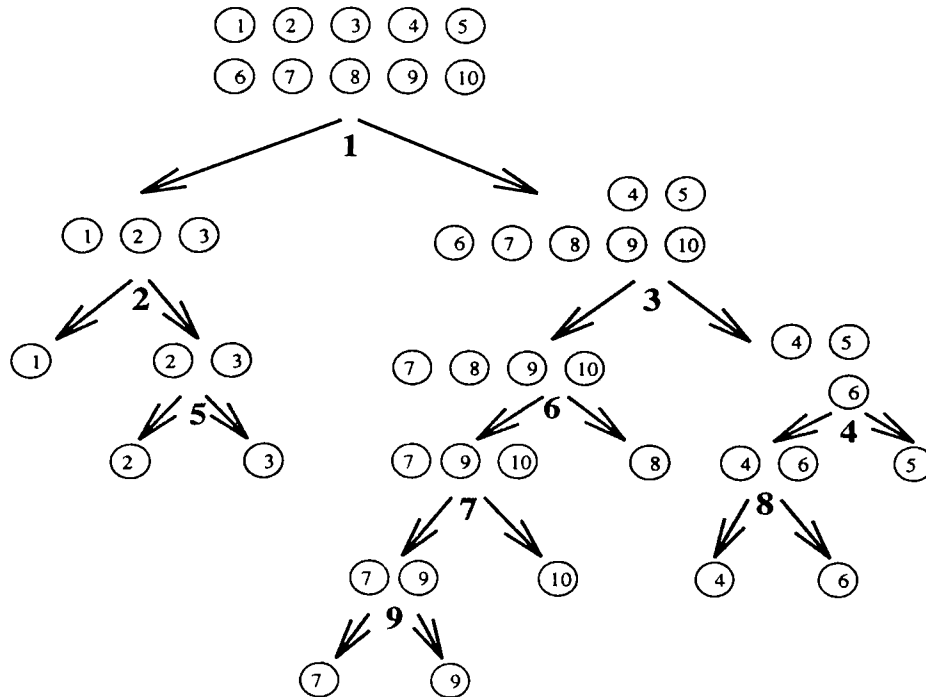


Figure 6.12: HAP-CUT{Example,clique,maximize number of partitions,80%}.

Two examples are given for illustration purposes. One is HAP-CUT{Table1, clique, maximize number of partitions, 80 %} which is given in Figure 6.12. In this example, the solution is 10 partitions of size one. Another example is HAP-CUT{Table1, star, any-criterion, 60 %} whose cut tree is given in Figure 6.11. No matter what an efficiency criterion is specified, the solution is of the form given in Figure 6.13.

The time complexity of HAP-CUT is determined by the second step, assuming calculation of efficiency values has a lower time complexity. If the algorithm by Melhorn *et al.* is used for determining the multi-terminal flows in the second step, the computational complexity of HAP-CUT is $O(\frac{n^4}{\log_2 n})$. In the case of clique approximation, $n = m$, whereas $n = m + p$ in star representation.

The clique approximation version of HAP-CUT was coded in Fortran on a SUN/Unix platform and several runs for 85 machine types and 1200 parts were made. The mean time value obtained from adding user time and system time elapsed on the behalf of user which is determined from the system function *time*

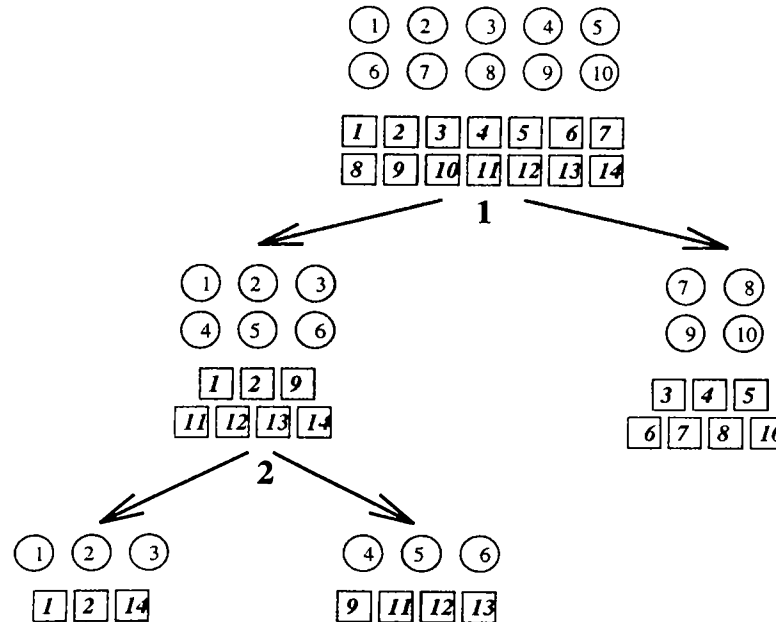


Figure 6.13: HAP-CUT{Example,star,any-criterion,60%}.

was 21.421 seconds, whereas the shortest run took 21.00 seconds and the longest one took 21.94 seconds. These run times seem to be attractive.

6.3 Sequential Identifying the Best Manufacturing Cell

The problem of identifying the best manufacturing cell is the grouping of the machines and parts into one cell such that the profit obtained from the cell is maximized. The cost of a cell over a period of time can roughly be measured with the total cost of the machines that form the cell. On the other hand, the gain of a cell over the same time period can be calculated as the total revenue obtained from the parts produced in the cell. Thus, the profit of a cell is the total revenue of the parts produced minus the total cost of its machines. Then the problem of identifying the best manufacturing cell is to form the cell with the maximum profit value.

There are two entity sets in this problem. The machine set M contains m

machine types each having costs c_i , $i = 1, \dots, m$. The set of parts P consists of p parts having revenue values d_j , $j = 1, \dots, p$. There is a routing relationship between these two sets. Let us define e_j as the set of machine types that are used in the fabrication of part j . The relation is from the part set to the subsets of the machine set. Let $k_j = |e_j|$ denote the number of distinct machines that part j visits.

Let S be a subset of machines, or equivalently a subset of nodes in the hypergraph representation. The set of parts that can be produced with this machine set, or equivalently the set of internal hyperedges with respect to S is denoted by $\gamma(S)$. The problem of identifying the best manufacturing cell is combinatorially formulated as below:

$$\max_{S \subseteq M} \sum_{e_j \in \gamma(S)} d_j - \sum_{i \in S} c_i.$$

Let x_i be a decision variable that indicates whether machine i is in the cell or not. Similarly, let y_j be a decision variable controlling whether part j can be produced in the cell or not. Consequently, the mathematical programming formulation of the problem is as follows:

$$\text{Max. } \sum_{j=1}^p d_j y_j - \sum_{i=1}^m c_i x_i$$

s.t.

$$\begin{aligned} y_j &= x_{i_1} \cdot x_{i_2} \cdot \dots \cdot x_{i_{k_j}} & j &= 1, \dots, p & e_j &= \{i_1, i_2, \dots, i_{k_j}\} \\ x_i &= \begin{cases} 1, & i \in S \\ 0, & \text{otherwise} \end{cases} & i &= 1, \dots, m \\ y_j &= \begin{cases} 1, & e_j \in \gamma(S) \\ 0, & \text{otherwise} \end{cases} \end{aligned}$$

This formulation is \mathcal{NP} -Hard. Please recall from Chapter 4 that the above nonlinear constraints $y_j = \prod_{i \in e_j} x_i$ can be replaced by linear constraints $0 \leq y_j \leq x_i \leq 1$, and $\sum x_i \leq (r-1) + y_j$ where $i \in e_j$. If all machine costs, c_i 's, and all part revenues d_j 's are nonnegative then the objective function does the job of the second set of constraints $\sum x_i \leq (r-1) + y_j$. This yields the formulation given

below:

$$\text{Maximize } \sum_{j=1}^p d_j y_j - \sum_{i=1}^m c_i x_i$$

subject to:

$$y_j \leq x_i \quad \forall i \in e_j, \forall j = 1, \dots, p$$

$$x_i \leq 1, \quad \forall i = 1, \dots, m$$

$$y_j \geq 0, \quad \forall j = 1, \dots, p$$

The problem of identifying the best cell is polynomially solvable by Corollary 4.2.

A new heuristic for the cell formation problem can be developed by sequentially solving the problem of identifying the best manufacturing cell. First, the best manufacturing cell is identified. After the machine requirements of the best cell is calculated, the required number of machines of each type is assigned to this cell. Next, the identification of the current best manufacturing cell is done by solving the problem on the remaining machines and parts. These iterations terminate when there is no profitable cell left. All of the remaining machines are grouped together forming the remainder cell. Any unassigned part can either be manufactured in the remainder cell or subcontracted or assigned to one of the cells formed.

Various Cplex 3.0 alternative Linear Programming routines can be used for solving the problem of identifying the best manufacturing cell. These are standard primal simplex, dual simplex, barrier method and a hybrid barrier module. A computational study is carried out for choosing the method to be used for cell formation. In this study, ten randomly generated problems are solved for each machine-part pair. The time statistics are given in Table 6.3. The barrier method is considered as the best among all. It statistically dominates all other methods in almost all of the cases, i.e., it has the lowest mean value and has the lowest variance. The barrier method is approximately 18 percent better than the second best method as the mean values are considered.

The new algorithm for cell formation is termed as *Sequential Identifying the Best Manufacturing Cell*, SIBC:

M	P	Primal		Dual		Barrier		Hybrid	
		Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.
25	50	0.2183	0.0138	0.2183	0.0189	0.1899	0.0185	0.2166	0.0258
30	60	0.3133	0.0339	0.3166	0.0307	0.2666	0.0223	0.3050	0.0298
35	70	0.4266	0.0317	0.4416	0.0260	0.3516	0.0203	0.4166	0.0316
40	80	0.5583	0.0367	0.5683	0.0397	0.4616	0.0258	0.5466	0.0439
45	90	0.7033	0.0276	0.7083	0.0281	0.5900	0.0249	0.6916	0.0436
50	100	0.8533	0.0331	0.8666	0.0387	0.7100	0.0238	0.8550	0.0441
55	110	1.0816	0.0556	1.0850	0.0456	0.8899	0.0302	1.0633	0.0697
60	120	1.2683	0.0630	1.2916	0.0533	1.0450	0.0380	1.2900	0.0955
65	130	1.4766	0.0650	1.5116	0.0767	1.2549	0.0826	1.5400	0.1252
70	140	1.7050	0.0875	1.7750	0.0624	1.4350	0.0844	1.7983	0.1440
75	150	1.9916	0.1272	2.0616	0.1229	1.6750	0.0696	2.1183	0.1571
80	160	2.3049	0.1154	2.3400	0.1302	1.9433	0.1227	2.1300	0.1390

Table 6.3: Time comparison of the four alternative methods in Cplex3.0.

Algorithm SIBC {cell formation problem, LP routine, threshold}

S0. $k=1$.

S1. Identify the current best manufacturing cell using an LP optimizer. Let z_k be the objective value and x^k, y^k be the current optimal solution.

S2. If $z_k \geq 0$ or $\frac{\sum_{j=1}^p d_j y_j}{\sum_{i=1}^m c_i x_i} \geq \text{threshold}$ then Form k^{th} cell.
otherwise Go to S5.

S3. Calculate the physical machine requirements of the current cell.
Remove the parts in the cell, i.e., remove $\forall j$ such that $y_j = 1$.
Remove the machine types that have no physical machines left.

S4. $k \leftarrow k + 1$.
Go to S1.

S5. Form the remainder cell.
Reassign the remaining parts to the cells $1, \dots, k$ and the remainder cell.

S6. Calculate efficiency measures.

In order to illustrate the execution of the above algorithm, our example problem is fed to the algorithm. The first iteration is presented in Figure 6.14. Machine costs and part revenues are shown in the figure. The Cplex solution states

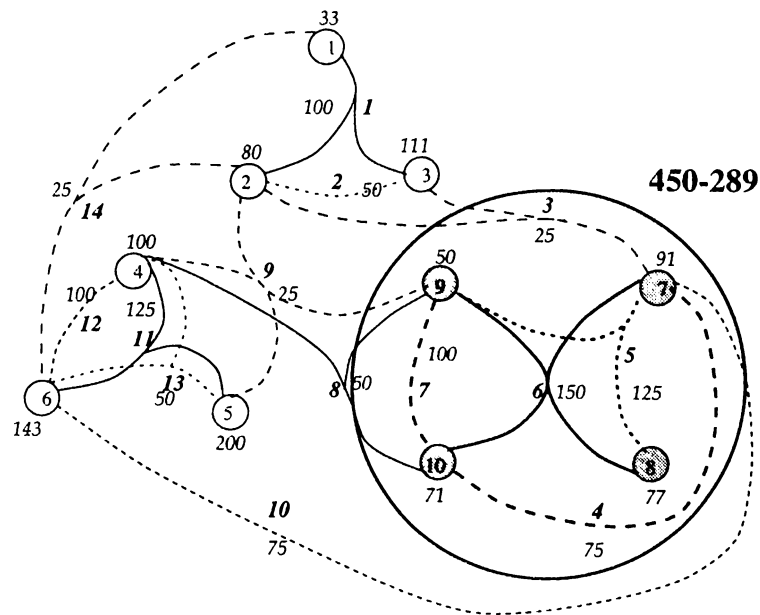


Figure 6.14: First iteration of SIBC{Example,netflo,2/3}.

the first cell is composed of machines 7, 8, 9, and 10. This cell is dedicated to the manufacture of parts 4, 5, 6, and 7. The profit of the cell is $450-289=161>0$. In this cell, there are 2 machines of type 7, 2 machines of type 8, 2 machines of type 9, and again 2 machines of type 10. The second iteration is illustrated in Figure 6.15. The LP solution says that machine types 1, 2, and 3 are grouped together with a cost of 224 units, and parts 1 and 2 are produced in this cell. Since the revenue-cost ratio is slightly greater than $2/3$, this cell is formed with a single machine of type 1, 2 machines of type 2 and a single machine of type 3. The remainder cell consists of machines 2, 3, 4, 5, 6, 7, 9, and 10. The parts 3, 8, 9, 10, 11, 12, 13 are assigned to be produced in the remainder cell. Part 14 is either assigned to the first cell or to the remainder cell.

6.4 Efficiency Measures

The evaluation of cell formation solutions is a vital issue in the design of Cellular Manufacturing systems. Although a number of techniques have already been

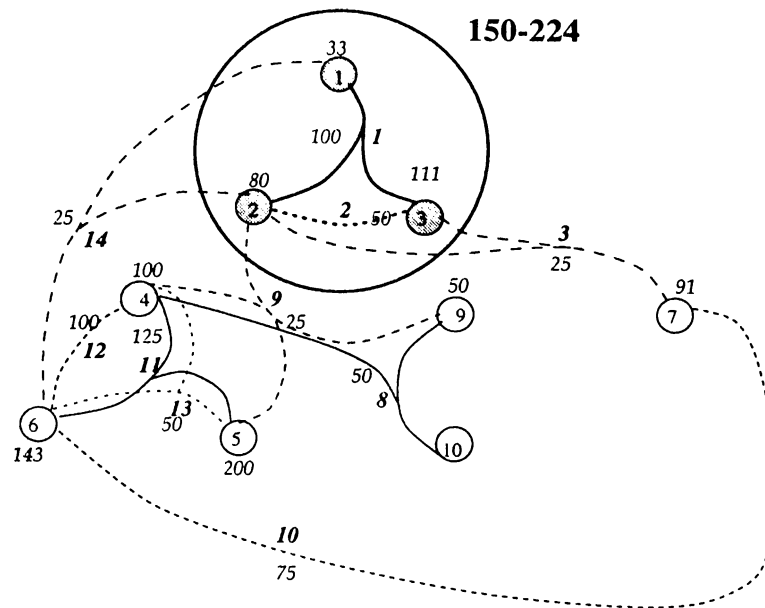


Figure 6.15: Second iteration of SIBC{Example,netflo,2/3}.

developed, the evaluation of cell formation solutions has remained somewhat qualitative [4, 20, 21, 96, 97]. Some of the commonly used quantitative efficiency measures are the number of inter- and inner-cell moves, the number and cost of duplicated equipment, the number of parts removed from the system, and machine utilizations [3, 5, 20, 21, 69, 70, 71, 79, 85, 86, 92, 94]. In particular, cell formation techniques have usually been compared in relation to the number of exceptional elements generated in the solutions [5, 20, 79, 94, 96].

A modelling tool for cell formation is the machine-part incidence matrix, the rows of which correspond to machine types and the columns to parts. Each element of the incidence matrix is 'one' if there exists a routing relation between the associated column and row, otherwise it is 'zero'. The objective of the cell formation problem is to obtain a block-diagonal structure in the incidence matrix. The resulting diagonal blocks represent the manufacturing cells. Desirable cell formation solutions are the ones in which all parts complete all of their operations in their assigned cell. For such solutions, there is no inter-cell movement of parts. *Exceptional elements* are the entries of the incidence matrix that do not belong

to any diagonal block preventing the solution from being a desirable one. They are actually connections inbetween the end-points of the hyperedges in the cut set.

It is also possible to use work-load matrices in solving cell formation problems as they contain considerably more information than incidence matrices. Each entry of a work-load matrix represents the percentage of machine capacity allocated to the corresponding operation. The sum of the elements of a specific row of a work-load matrix indicates the number of machines desired of that type. Work-load matrix enables user to deal with individual machines instead of machine types. Hence, more than one cell can have the same machine type. Furthermore, if the workload of an exceptional element is high enough, a machine of the corresponding type will be inserted into the corresponding part's cell. Thus, the use of work-load matrices provides an opportunity to eliminate some of the exceptional elements [59].

The efficiency of each cell formation solution can be measured from its work-load matrix. Three efficiency indices with assumed values between zero and one are suggested for evaluating the solutions. The first measure is the modified grouping efficiency which penalizes exceptional elements and considers inner-cell densities. This measure is an extended version of what Chandrasekharan and Rajagopalan reported [20, 21] whereas the remaining two are developed within the scope of this study. The second measure pertains to the inner-cell load balances. The third measure focuses on under-utilizations of individual machines. Discussion of these efficiency measures requires some notation and definitions:

i : machine type index ($i = 1, \dots, T$),

j : part index ($j = 1, \dots, P$),

k : cell index ($k = 1, \dots, K$),

$CM(k)$: index set of machine types that are assigned to cell k ,

$CP(k)$: index set of parts that are assigned to cell k ,

$AC(j)$: index of cell to which part j is assigned,

AV_j : annual production volume of part j {units},

$ST_{i,j}$: total standard times of operations of part j on machine type i {time-units/unit},

C_i : annual availability of machine type i {time-units/machine},

$WL_{i,j}$: annual work-load on machine type i induced by part j {machine-fraction},

$$WL_{i,j} \doteq \frac{AV_j \times ST_{i,j}}{C_i}$$

$TU_{k,i}$: total usage of machines of type i in cell k {machine-fraction},

$$TU_{k,i} \doteq \sum_{j \in CP(k)} WL_{i,j}$$

$N_{k,i}$: number of machines of type i in cell k {machines},

$$N_{k,i} \doteq \lceil TU_{k,i} \rceil$$

S_k : number of different machine types in cell k ,

OC_i : annual operating cost (including depreciation) of a machine of type i {\$/machine},

$TWLC_j$: total work-load cost of part j {\\$},

$$TWLC_j \doteq \sum_{i=1}^T WL_{i,j} \times OC_i$$

WCC_j : work-load cost of part j in its assigned cell {\\$},

$$WCC_j \doteq \sum_{i \in CM(AC(j))} WL_{i,j} \times OC_i$$

$WLCE_j$: work-load cost of exceptional elements belonging to part j {\\$},

$$WLC E_j \doteq TWLC_j - WCC_j$$

$FP_{i,j}$: field potential value of clustering both part j and machine type i into the same cell $\{\$^2\}$. It is an artificial weight given to each entry of the work-load matrix which represents the solution. The associated weight is the multiplication of column and row weights. Here it is assumed that machine types differ from each other in terms of total annual operating costs whereas parts are differentiated by means of total work-load costs:

$$FP_{i,j} \doteq TWLC_j \times OC_i \times N_{AC(j),i}$$

$AP_{i,j}$: assignment potential of clustering both part j and machine type i into the same cell. It is the associated field potential value incurred only if both machine type and part are assigned to the same cell.

$$AP_{i,j} \doteq \begin{cases} FP_{i,j} & \text{if } WL_{i,j} > 0, \\ 0 & \text{otherwise.} \end{cases}$$

$MWL_{k,i}$: mean percent workload on machines of type i assigned to cell k $\{\%\}$,

$$MWL_{k,i} \doteq \frac{100}{N_{k,i}} \times \sum_{j \in CP(k)} WL_{i,j}$$

MCL_k : mean percent cell-load in cell k $\{\%\}$,

$$MCL_k \doteq \frac{1}{S_k} \times \sum_{i \in CM(k)} MWL_{k,i}$$

$UU_{k,i}$: total under-utilization of machine type i in cell k {machine-fraction},

$$UU_{k,i} \doteq N_{k,i} - TU_{k,i}$$

Each element $WL_{i,j}$ of the work-load matrix is nothing but a machine fraction value used in facilities planning. It has a nonnegative real value representing the number of machines of type i needed for carrying out the related annual total manufacturing operations of part j . Machine types are different in terms of costs. It is unrealistic to treat a cheap machine and an expensive one similarly. Specifically, an exceptional element due to an assembly bench can be eliminated simply by duplicating it. However, this is certainly not the case with CNCs. Furthermore, two different machine types of the same initial cost may have different operating expenses. Therefore, rows of the work-load matrix are differentiated from each other in terms of total annual operating cost of all machines of the corresponding type. Similarly, each part is different considering product volumes and manufacturing expenses. In this analysis, we solely consider machine operating costs, including depreciation and direct labor, in manufacturing expenses and ignore direct material costs. But these terms can easily be incorporated into cost expressions. Hence, each column of the matrix has different ranks equal to the total work-load cost of the corresponding part.

Modified grouping efficiency is a combined measure made up of two parts. The first part measures the inter-cell work-load created by exceptional elements. Inter-cell flow efficiency, μ_1 , is defined as the complementary normalized cost of all exceptional elements:

$$\mu_1 \doteq 1 - \frac{\sum_{j=1}^P WLC E_j}{\sum_{j=1}^P TWLC_j} \quad (6.1)$$

The second part of the modified grouping efficiency is a weighed estimate of the inner-cell densities. Each block-diagonal entry in the work-load matrix is weighed by multiplying the column and row weights. Consequently, a potential field is defined for each diagonal block representing a manufacturing cell. The estimated density of a cell is the normalized total potential in this field. Therefore,

inner-cell efficiency, μ_2 , of any cell formation solution is defined as:

$$\mu_2 = \frac{\sum_{j=1}^P \sum_{i \in CM(AC(j))} AP_{i,j}}{\sum_{j=1}^P \sum_{i \in CM(AC(j))} FP_{i,j}} \quad (6.2)$$

There are two extreme cell formation situations. One is the cell system which contains small, dense cells and a lot of exceptional elements. This situation which maximizes inner-cell density measure μ_2 is penalized by inter-cell flow efficiency μ_1 . The other is the cell system with large, sparse cells without exceptional elements. This situation which maximizes μ_1 is penalized by μ_2 . Hence, inter-cell flow efficiency μ_1 and inner-cell density μ_2 are inversely proportional such that each of which penalizes one extreme whereas the other improves.

Grouping efficiency, μ , is defined as the convex combination of inter-cell flow and inner-cell efficiencies:

$$\mu = \alpha \times \mu_1 + (1 - \alpha) \times \mu_2, \quad \alpha \in [0, 1] \quad (6.3)$$

The parameter α can be interpreted as an indication of whether inner-cell efficiencies or inter-cell flows are more important to the decision maker. A large value of α gives more weight to exceptional elements. As α approaches unity, there emerges a tendency to eliminate all exceptional elements and the trivial cell formation solution turns out to be a single big cell, that is a job shop system. On the other hand, a very small α value indicates that inner-cell efficiency is more important than inter-cell flow efficiency, which makes a solution with high number of small-sized dense cells acquire the best value among all cell formation alternatives. That is the reason why moderate values of α are suggested to calculate the modified grouping efficiency.

Work-load balance measure, β , shows the degree of machine load balance in each cell. It is an important measure if other resources in a cell like personnel are shared. On the other hand, balanced cells have better performances especially in JIT systems. If all machines in each cell are evenly loaded, then the work-load balance index takes a value very close to one. This measure considers the squared machine work-load deviations from the mean cell work-load. If there are more than one individual machine of the same type in a cell, the squared

deviation is adjusted accordingly. Furthermore, the squared deviation due to a machine type is weighed by the associated operation cost. The weighed squared deviations are first calculated for all machine types in each cell, then they are added and normalized. Hence this efficiency measure is defined as the normalized sum of square of the weighed deviations between the mean cell load and individual machine loads:

$$\beta \doteq 1 - \sqrt{\frac{\sum_{k=1}^K \sum_{i \in CM(k)} (MWL_{k,i} - MCL_k)^2 \times N_{k,i} \times OC_i}{10000 \times \sum_{k=1}^K \sum_{i \in CM(k)} N_{k,i} \times OC_i}} \quad (6.4)$$

The third efficiency measure, $\bar{\gamma}$, shows the relative weighed *under-utilization* levels of the individual machines. Individual machine under-utilizations are multiplied by the associated annual operating costs and normalized to find the case dependent under-utilization index:

$$\gamma \doteq \frac{\sum_{k=1}^K \sum_{i \in CM(k)} UU_{k,i} \times OC_i}{\sum_{k=1}^K \sum_{i \in CM(k)} OC_i} \quad (6.5)$$

This index includes the initial under-utilizations of machines before cell formation. The real under-utilization index should indicate the amount of relative under-utilizations created due to cell formation. The last efficiency index is calculated as the complement of the relative utilization of the cells scaled by maximum utilization:

$$\bar{\gamma} \doteq 1 - \frac{1 - \gamma}{1 - \gamma_{JS}}, \quad (6.6)$$

where γ_{JS} is the initial under-utilizations of the machines in the original job shop. Actually, $\bar{\gamma}$ indicates the ratio between the reduction in average utilization due to Cellular Manufacturing (i.e., $\gamma - \gamma_{JS}$) divided by the original average utilization level (i.e., $1 - \gamma_{JS}$).

The solution can be freed of exceptional elements by machine duplication and addition. In the case of machine duplication, the number of individual machines of type i assigned to cell k , $N_{k,i}$, is increased by one. This increases the field potential value of all parts assigned to cell k along row i . Consequently, zero entries in row i are penalized more in the denominator of (6.2), decreasing μ_2 . When machine type i is added into the cell k , a new row is created. The zero elements in

this row decrease μ_2 . On the other hand, the presence of exceptional elements creates inter-cell work-loads and clearly decreases μ_1 . Both machine duplication or addition and creation of inter-cell flows do affect the mean percent work-loads. However, the influence on β is too complex to investigate analytically. In the case of machine addition due to an exceptional element belonging to part j , the numerator of (6.5) is increased by $OC_i \times ([WL_{i,j}] - WL_{i,j})$ and the denominator by OC_i . It may decrease as well as increase the under-utilization index $\bar{\gamma}$. If bottleneck machine i is duplicated then the under-utilization of that machine type is increased leading to an overall increase in γ , hence in $\bar{\gamma}$. If we keep the exceptional elements and create inter-cell work-loads, the numerator of (6.5) will be increased. Therefore, $\bar{\gamma}$ will be increased as well.

$$\left[\begin{array}{cccccc} + & + & + & + & + & + \\ + & + & + & + & + & + \\ + & + & + & + & + & + \\ + & + & + & + & + & + \\ + & + & + & + & + & + \end{array} \right] \quad \left[\begin{array}{cccc} + & + & + & \\ + & + & + & + \\ + & + & + & + \\ + & + & + & + \end{array} \right] \quad \left[\begin{array}{ccc|c} + & + & + & + \\ + & + & + & \\ \hline & & & + & + & + \\ + & & & + & + & + \\ & & & + & + & + \end{array} \right]$$

a) $K = 1$
 $\mu_1 = 1, \mu_2 = 1,$
 $\beta = 1$
 $\gamma = 1, \bar{\gamma} = 0$

b) $K = 1$
 $\mu_1 = 1, \mu_2 = 1/2,$
 $\beta = 1$
 $\gamma = 1/2, \bar{\gamma} = 0$

c) $K = 2$
 $\mu_1 = 15/17, \mu_2 = 1,$
 $\beta = 1$
 $\gamma = 1/2, \bar{\gamma} = 2/17$

$$\left[\begin{array}{ccc|c} + & + & + & \\ + & + & + & \\ \hline & & & + & + & + \\ & & & + & + & + \\ & & & + & + & + \end{array} \right] \quad \left[\begin{array}{cc|c} + & + & \\ + & + & + \\ \hline & & + & + & + \\ & & + & + & + \\ & & & + & + \end{array} \right] \quad \left[\begin{array}{ccc|c} + & + & + & + \\ + & + & + & \\ \hline & & & + & + & + \\ + & & & + & + & + \\ & & & + & + & + \end{array} \right]$$

d) $K = 2$
 $\mu_1 = 1, \mu_2 = 1,$
 $\beta = 1$
 $\gamma = 1/2, \bar{\gamma} = 0$

e) $K = 2$
 $\mu_1 = 1, \mu_2 = 13/15,$
 $\beta = 0.917$
 $\gamma = 17/30, \bar{\gamma} = 0$

f) $K = 2$
 $\mu_1 = 13/15, \mu_2 = 13/15,$
 $\beta = 0.921$
 $\gamma = 17/30, \bar{\gamma} = 2/15$

Figure 6.16: Some examples of efficiency values.

The scores of the three efficiency indices represent the quality of the cell

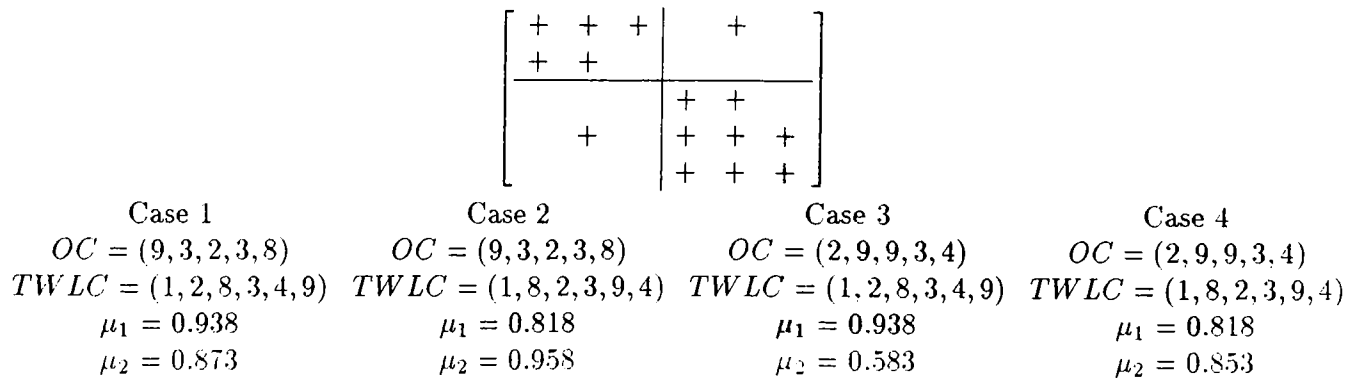


Figure 6.17: Effect of part and machine type weights on grouping efficiency values.

formation solution on a quantitative scale. A number of examples showing the effectiveness of the indices are given in Figures 6.16, 6.17, and 6.18. The cases in Figure 6.16 represent solutions to six different but simple cell formation situations where parts, machines, and operations are assumed to be the same. In the example, it is also presupposed that each machine has a capacity of six operations. In the first two cases, the cell formation solution is a single cell, the original job shop. For the rest, there are two cells in the solutions: cell-one contains the top two machine types and the leftmost three parts, whereas cell-two is composed of the bottom three machine types and is dedicated to fabricate the rightmost three parts. In cases a,b,d and e, there are no exceptional elements. However, both cases c and f have two exceptional elements in their solutions. The cells formed in cases a, c and d are full-dense cells, whereas the other cases have solutions with relatively sparse cells. The effect of differences in machine types and parts on grouping efficiency values is investigated in Figure 6.17. which contains four cases in which all operations are the same. The only difference is due to part and machine type weights, resulting in different grouping efficiency values. In the last example, the operations are different in the two cases but the machine type weights and the cell formation solution are the same. The effect on work-load balance and under-utilization measures is presented in Figure 6.18.

30	.25	.25	.25	.25	.05	.05	.05	.85
20	.40	.40			.40	.40		
40					.40	.40		
60		.25			.25	.25	.25	.85
50					.30	.30	.30	.30
<i>OC</i>								

Case i	Case ii
$\beta = 0.997$	$\beta = 0.656$
$\gamma = 0.168$	$\gamma = 0.468$
$\gamma_{JS} = 0.085$	$\gamma_{JS} = 0.085$
$\bar{\gamma} = 0.090$	$\bar{\gamma} = 0.489$

Figure 6.18: Effect of work-loads on balance and under-utilization measures.

The above explained efficiency indices are quite effective in evaluating the cell formation solutions as can be seen from the examples. All three efficiency indices are more powerful when applied to real life situations. These measures are simple and rough enough to use in the initial phases of the design process. The data for the calculation of the efficiency measures are present when a manufacturing system is to be designed. The standard times and operation sequences, rough estimates of annual demand figures and machine costs constitute the input to the initial phases of designing Cellular Manufacturing systems. Our efficiency indices make use of these data to determine the relative weights of parts, machine types and operations. Although originally developed for a Cellular Manufacturing environment, these indices can well be applied to other manufacturing environments.

6.5 Other Cell Formation Techniques Evaluated

Some of the cell formation techniques cited in the literature have been developed by academic researchers while others have emerged as a result of practical applications. Moreover, computer implementation of earlier techniques is usually rather painstaking as they are not mathematically oriented. At any rate, most of the techniques proposed for the cell formation problem are of little value for they ignore performance criteria, which are important in achieving satisfactory solutions. What is more, most procedures for manufacturing cells generate different solutions to the same cell formation problem depending on the form of input.

A subset of six analytical cell formation techniques which require routing information between machine types and parts is selected for a detailed analysis. Although these techniques consider neither possible routing alternatives nor operation sequences, the resulting solutions are independent of any special block-diagonal structure embedded in the input data. As a result, each technique generates a unique solution to the same problem fed in different input formats. Moreover, they are computationally efficient. If a solution with a perfect block-diagonal structure is found possible, then all of these techniques will generate this ideal cell formation solution with absolutely no exceptional elements.

The techniques selected for further analysis and comparison are lattice-theoretic *combinatorial grouping* (COMBGR), *modified rank order clustering* (MODROC), *machine-component cell formation* (MACE), and *within-cell utilization based clustering* (WUBC), *cost analysis algorithm* (CAA), and *zero-one data - ideal-seed algorithm for clustering* (ZODIAC).

Of these, COMBGR and MACE consider only the machine grouping problem. COMBGR uses set inclusion and joint set union methods whereas MACE employs similarity coefficients. Subsequent to the identification of machine clusters, parts are assigned to the cells in both techniques. However, part and machine assignments are considered simultaneously and/or subsequently in MODROC, WUBC, CAA, and ZODIAC. MODROC is a reordering method applied to the

machine-part incidence matrix. WUBC and CAA are searching algorithms on the graph generated by parts and machines as vertices and routing relationships as edges. ZODIAC is the only seed clustering technique reported in the literature.

COMBGR, MACE and MODROC, on the other hand, are hierarchical techniques. Initially candidate cells are generated and subsequently merged into larger cells. One common feature of these techniques is that the candidate cell decision of the first stage directly affects the final solution. If a machine-part pair is clustered initially in a particular cell, the assignment in the final solution will be the same cell. Other techniques are non-hierarchical in that a machine-part pair previously grouped could possibly be reassigned to a different cell. A summary of the features of these techniques is described next. For more detailed analysis and critique of these techniques, see [59].

COMBGR is a lattice-theoretic hierarchical grouping algorithm developed by Purcheck [82, 85, 86]. The basic advantage of COMBGR is that it generates cell formation solutions without any exceptional elements. However, proposed cells are relatively large so they cause more or less the same drawbacks encountered in job shops. This algorithm divides parts into two classes— hosts and guests. Hosts are the parts whose machine set is not contained in the respective machine sets of any other part. Hosts constitute the minimal independent set in terms of routing relations. Initially, cells are identified in such a way that each host represents a candidate cell. In other words, a candidate cell is constructed by clustering all machines used by the corresponding host. Candidate cells are merged successively until the original job shop is obtained. Each merge iteration creates an alternative cell formation solution.

MACE is another hierarchical, machine-grouping cell formation technique [94]. It uses measures of similarity showing the degree to which a set of parts can be processed on a pair of machines. MACE groups machines only if their similarity coefficient are greater than a prespecified value. During the first stage, the technique generates initial machine clusters, each representing a candidate cell. Consequently, the candidate cells are successively joined by using similarity coefficients. The last stage of MACE involves part assignments to the final cells.

MODROC is based on the original Rank Order Clustering (ROC) technique [64]. The ROC algorithm treats each row or column of the machine-part incidence matrix as a binary word. Integer equivalents of binary words are calculated and rows and columns are reordered successively in descending order of integer equivalents. A ROC iteration consists of a row reordering followed by a column reordering. Those iterations are terminated when no change is encountered. This algorithm was altered [65] by utilizing a new data structure and a new sorting mechanism. Yet another modification was offered later by Chandrasekharan and Rajagopalan [21]. They used King's iterations twice to obtain an incidence matrix containing a rectangular block of 'ones' at its top-left corner. This rectangular block represents a candidate cell. The corresponding columns of the candidate cell are all eliminated from the incidence matrix and the procedure resumes. After all candidate cells are formed, they are merged successively until the final solution is attained. A similarity coefficient method is applied to assist this merging process.

WUBC is a graph searching cell formation technique [5] by Ballakur and Steudel. WUBC induces a breadth-first search on the graph generated by the routing relationships between parts and machine types. A *key* machine type is selected as the root in the search and all parts routed through the key machine type are examined. These parts are either admitted to the cell generated by the key machine type or remain in their previously assigned cells. The parts that are not previously assigned are automatically included in the cell when examined for the first time. Consequently, all machine types related to the admitted parts are examined in this process. Machine types are added to the cell if their within-cell work-loads due to the parts already assigned exceed a prespecified level. Upon completion of each search, the required number of machines of each type are allocated to the cell based on the within-cell utilizations. Next, another search is initiated after selecting a new key machine type among the remaining machines. Finally, a *remainder cell* is formed by bringing all left-over machines together. All the previously assigned parts may be reassigned to the remainder cell provided that it is not empty. Conversely, unassigned parts, if any, are included in the remainder cell.

Like WUBC, CAA (Kusiak and Chow) [70] is a graph searching cell formation technique. However, it focuses on parts rather than machines in defining the root during the search process. CAA initiates a breadth-first search on the machine-part graph. Each search identifies a different cell. During the search, an admit/reject decision is made for all parts except for the root. A rejection eliminates the part under consideration from the analysis. A part is admitted to the cell unless it increases the number of machine types above a prespecified value. The search continues until there are no parts to be assigned to the cell. The next cell is constructed by taking another key part as the root. This process ceases when all the parts have been either assigned to a cell or rejected from the analysis.

ZODIAC is a seed clustering cell formation technique [20, 22] by Chandrasekharan and Rajagopalan. The parts and machine types are treated independently in the initial phase. Rows of the machine-part incidence matrix represent machine types in binary vector format. Similarly, the binary vector for a specific part can be obtained from the corresponding column. Parts and machine types are clustered separately by means of seeds, where a seed is a binary vector. Part and machine clusters are then assigned to each other by the use of similarity coefficients. Consequently, each assignment produces a cell.

In addition to the evaluation of these techniques, various modifications and extensions to the original algorithms are made in order to bring them into the same state so that a sound comparative analysis becomes possible. To illustrate, most of the selected techniques are designed to operate on the incidence matrix, which means they do not recognize individual machines of the same type. These techniques are altered to enjoy the advantage of using work-load and cost information. Other modifications and extensions are as follows:

COMBGR's set partitioning scheme is modified to improve the quality and number of alternative cell formation solutions. COMBGR joins each candidate cell with at least one other cell, and this puts an artificial condition on the candidate cell in each merging iteration. Besides, it is sometimes beneficial to keep a candidate cell untouched. Part assignments are not considered in the original

COMBGR. Hence, we propose a part assignment scheme for COMBGR which is optimal in terms of grouping efficiency when all parts, machines and operations are identical. COMBGR performs merging iterations sequentially until the original shop is obtained. In the meantime, the inner-cell efficiency value of the alternative solutions is getting worse since the proposed cells are getting larger. There is no need to iterate further if the machine requirements for all types are met. This can be considered as a sensible stopping criterion. Another stopping criterion we suggest is the minimum machine-difference between the candidate cells to be merged. If this value exceeds a prespecified threshold, the merging of candidate cells is interrupted. This threshold value should be determined by taking the trade-off between extra investment and inner-cell densities into account.

After initial candidate cells are identified, MODROC merges one pair of candidate cells at a time. Therefore, MODROC's cell formations during the final iterations usually contain one large cell and a number of small ones. Naturally, the big cell attracts nearby small cells, which reduces the quality of cell formation since the resulting large cell decreases the inner-cell density. The following merging scheme is proposed to overcome this drawback and to reduce total number of merging iterations. All pairs of cells with similarity coefficient values greater than a specified value are picked in each merging iteration so that more than one cell-pair merging take place. The threshold for similarity coefficients can be determined as a prespecified percentage of the maximum similarity value of the current pairs. If the number of independent pairs of candidate cells exceeds another prespecified value such as a percentage of number of current candidate cells, the threshold is increased for the current iteration. Each merging iteration generates an alternative cell formation. As in the case of COMBGR, we offer the grouping efficiency measure as the sole criterion for picking the best cell formation solution alternative pertaining to the machine availabilities.

The part assignment scheme suggested by MACE leads to a large number of exceptional elements. Therefore, a superior part assignment scheme using work-load cost fractions is proposed. The work-load cost fraction of a part is the percentage of its total work-load cost that is allocated to its assigned cell.

For each part, the cell with the highest work-load cost fraction is selected for assignment. Two threshold values for similarity coefficients are introduced in merging machines. After the pair with maximum similarity is picked, two passes are made to configure the candidate cell. If the similarity values between a candidate machine and the selected pair are higher than the first threshold value, the candidate machine is joined to the cell identified by the selected pair. This operation is carried out for all the machines other than the selected pair. During the second pass, candidates with similarities to all machine types in the current composition of the cell exceeding the second threshold value are joined. The next pair is chosen among the remaining candidate cells and the two passes are carried out again. The MACE technique can alternatively use three different similarity coefficients whose individual effects are explained in the above mentioned study. The original version of MACE refers to the machines of the cells without any part assignments as “blocking machines”, nevertheless, it lacks a convenient method to handle them. Hence, such a method is also suggested.

When a part is reassigned by WUBC, work-loads of the related machine types in the part's previous cell are affected. A decrease in the work-load of a machine type might give rise to a decrease in the number of machines of that type, even to the removal of the type. This influences the cell-parts still having loads on the removed machines. If such a part has operations in another cell, it will naturally be reassigned. The new part assignment could lead to new work-load decreases. Therefore, WUBC is altered accordingly. Moreover, alternative measures for selecting either the key machine type or part assignments are evaluated. The machine type with the highest total work-load is found to be the best candidate for the the key machine type and the cell in which a part has the maximum work-load percentage is found to be the best cell to assign. The same study reveals that 0.60 is the best value for the cell admission factor.

As for CAA, an alternative rule to select the key part is suggested. The part with the highest total work-load cost is offered as the key part. CAA expands cells only by examining the costs of allowable non-key parts which do not increase the cell size above an upper limit. This method usually prevents similar parts from

joining the cell. Therefore, the following method is introduced. First, allowable non-key parts are examined for extra machine requirements. If the number of machines needed for a non-key part is less than the cell admission factor, then that part is considered as a candidate. The cell admission factor is defined as the ratio of the maximum number of extra machines needed to the current cell size. Secondly, the set of machine requirements for each candidate part is examined to see whether it contains the machine requirement set of other candidates. The cost of such candidates should be taken into account whenever a non-key part is selected for cell admission. After each cell is formed, the required number of individual machines of each type is calculated. If there are any remaining machines, then their corresponding types might be considered in the formation of future cells to prevent some parts from being rejected totally. Rejected parts are suggested to be reassigned to the already formed cells. For the assignment, the cells with the maximum work-load are proposed. Furthermore, the effects of the cell admission factor and the cell size upper limits are investigated. Results indicate that a cell admission factor of 0.20 is satisfactory, and the upper limit on the cell size should be of the minimum value which is the number of 'ones' in the most dense column of the incidence matrix.

ZODIAC chooses an arbitrary representative seed from each group, which may fail to represent the corresponding cluster. The most dense binary vector in each cluster is offered as the first representative seed. The remaining representative seeds can be determined in such a way that they will be distant from all current seeds. First, the candidates with the maximum distance from all the seeds become representative seeds. The maximum distance is controlled by machine difference factor, and this factor is decreased by a threshold percentage for the next set of representative seeds. Moreover, the steps of ZODIAC are resynchronized to improve the quality of solutions and to accelerate the algorithm.

Furthermore, the selected techniques make use of fine tuning variables such as threshold values. With the help of the efficiency indices described in the previous section, the alternatives are evaluated and the best combinations of fine tuning values are identified. Yet the reader should refer to the original study [59] for a

detailed analysis of the extensions, modifications and the improvements offered. The modified algorithms used in the analysis are listed below:

Algorithm {COMBGR}

- S-1. Input part routing codes, number of machines in the job shop, and maximum machine-difference limit;
- S-2. Compute sizes and code significances of parts,
Sort parts by size in decreasing order,
Order parts of the same size by code significance in descending order;
- S-3. Find hosts and guests, construct hospitality and flexibility relationships,
Identify initial candidate cells characterized by hosts,
Assign parts;
- S-4. Calculate size of minimal machine-differences between cells,
If minimal machine-difference size $>$ maximum machine-difference limit, jump to S-6,
Compute set combination sizes, forward and inverse relations,
Assign priorities;
- S-5. Form super-hosts,
Calculate total machine requirements,
If total machine requirements \leq total number of machines, jump to S-6,
Replace hosts by superhosts,
Return back to S-4;
- S-6. Output the solution,
Calculate efficiency measures;

end {COMBGR}.

Algorithm {MODROC}

- S-1. Input incidence matrix, machines in the job shop, lower limit on similarity coefficient, upper limit on number of independent parts, and aspiration level α ;
- S-2. Make two ROC iterations on the incidence matrix;
- S-3. Identify the largest top-left block of 'ones',
Determine the candidate cell,
Slice the corresponding columns in the incidence matrix,
If the resultant incidence matrix is not empty, return back to S-2;
- S-4. If total machine requirement \leq total number of machines, save the solution,
If number of cells is equal to one, go to S-6,
Generate similarity coefficient matrix,
Choose independent pairs of cells having higher similarities than the lower limit,
If number of independent pairs is zero, save the solution and go to S-6;
- S-5. Merge cells,
Join part families,
Return back to S-4;

- S-6. Based on α , choose the solution with the highest grouping efficiency value among all cell formation proposals,
Output the solution,
Calculate efficiency measures;

end { MODROC }.

Algorithm {WUBC}

- S-1. Input work-load matrix, cell admission factor, cell size upper limit, number of machines of each type, rule for key machine type selection, rule for part assignments;
- S-2. Select key machine type according to the input rule,
If there is no key machine type, then go to S-10,
Insert key machine type into the FCFS queue,
Add key machine type into cell;
- S-3. Examine all parts routed through the key machine type,
If the examined part is not already assigned, then assign the part;
If the examined part has a higher part assignment value in this cell, then mark the part;
- S-4. Evaluate all non-key machine types in the routings of marked or assigned parts,
If non-key type is neither admitted nor rejected and its $WLF \geq CAF$, then admit the non-key type, otherwise reject;
- S-5. Insert all single machine admitted non-key types into the FCFS queue,
Delete the top machine type from the FCFS queue,
If FCFS queue is not empty, then set the new key as the top element of the queue and go to S-3;
- S-6. If there is at least one machine type other than the key in the cell, go to S-7,
Erase marks on parts,
Release machines of the key type in this cell,
Prevent this type from being a key in further iterations,
Go to S-2;
- S-7. Compute WCU of all admitted machine types due to marked parts,
List admitted machine types in decreasing order of WCU values,
Assign admitted machine types in this order until CSUL is reached;
- S-8. Examine all marked parts,
Assign a marked part if it has a higher part assignment value in this cell;
- S-9. If there is no part assigned to the cell, then discard the cell,
Otherwise, for assigned marked parts, update the work-loads of the corresponding machine types in previous cells, release them if necessary,
Erase marks on parts,
Go to S-2;
- S-10. Add all left-over machines into the remainder cell,
Examine all parts for possible reassignment to the remainder cell;
- S-11. If there is no part assignment to the remainder cell, then go to S-12,
For reassigned parts, update work-loads of the corresponding machine types in previous cells,
If there is a machine release, then go to S-10;

- S-12. Output solution,
Calculate efficiency measures;

end {WUBC}.

Algorithm {CAA}

- S-1. Input work-load matrix (WL), cell admission factor (CAF), total work-load costs for parts (TWLC), and number of machines of each type,
Compute CSUL;
- S-2. Select the key part,
Assign the key part to the cell,
Add all machine types related to the key part into the cell;
- S-3. Find candidate parts,
If there is no candidate part, then go to S-5,
Investigate set inclusion relations between extra machine requirement of candidate parts,
Update cost of candidate parts;
- S-4. Find the candidate part having the maximum cost,
Expand the cell,
If CSUL is not reached, then go to S-3;
- S-5. Calculate number of machines for each type in the cell,
If there are left-over machines go to S-2;
- S-6. Assign rejected parts;
- S-7. Output solution,
Calculate efficiency measures;

end {CAA}.

Algorithm {MACE}

- S-1. Input similarity coefficient type, threshold value, number of machines in the job shop;
- S-2. Compute $NCC_{i,j}$, TNC_i , TFC_i ,
Calculate similarity coefficients of the selected type;
- S-3. Select the machine pair with the maximum similarity,
Examine the closest machines,
Form a candidate cell;
- S-4. Repeat S-3 until no more machine type is left;
- S-5. Compute inter-cell flows,
Replace machines by candidate cells,
Calculate similarity coefficients between candidate cells ($SCTF_{k,l}$),
Repeat S-3 until no more candidate cells are left;
- S-6. Assign parts,
Check the existence of blocking machines,
If there exists blocking machines, relocate them;

- S-7. Output the solution,
Calculate efficiency measures;

end {MACE}.

Algorithm {ZODIAC}

- S-1. Input machine-part incidence matrix, weighting factor q , threshold value for representative seeds,
Calculate maximum allowable number of cells, K^* ,
Set $K \leftarrow K^*$;
- S-2. Choose K artificial seeds for columns,
Cluster columns,
Choose representative seeds for columns,
Cluster columns;
- S-3. Find number of non-null column clusters, K_C ,
Modify $K \leftarrow K_C$,
Repeat S-2 for rows,
Find number of non-null row clusters, K_R ;
- S-4. Modify $K \leftarrow \text{Min}\{K_R, K_C\}$,
If $K_R \neq K_C$, then go to S-2,
Reorder rows and columns in the order of cluster membership;
- S-5. Compute similarity coefficients,
Allocate part clusters to machine clusters,
Reorder columns according to the new order of clusters;
- S-6. Compute clustering efficiency ξ , and relative efficiency ξ_R ,
If $\xi_R \cong 1$, then go to S-11,
Set $\hat{\xi} \leftarrow \xi$;
- S-7. Generate ideal seeds for machine clusters,
Cluster rows,
Modify $K \leftarrow K_R$,
Generate ideal seeds for part clusters,
Cluster columns,
Modify $K \leftarrow \text{Min}\{K_R, K_C\}$,
Generate representative seeds for part clusters,
Cluster columns,
Modify $K \leftarrow \text{Min}\{K_R, K_C\}$,
Generate ideal seeds for machine clusters,
Cluster rows;
- S-8. If $K_R \neq K_C$, then go to S-9,
Generate ideal seeds for part clusters,
Cluster columns,
If $K_R \neq K_C$, then go to S-7,
Go to S-10;
- S-9. Replace columns by rows,
Repeat S-7,
If $K_R \neq K_C$, then go to S-7;

- S-10. Reorder columns according to the new order of clusters,
 Compute ξ and ξ_R ,
 If $\xi_R \cong 1$, then go to S-11,
 If $\xi < \hat{\xi}$, then revert to the earlier grouping and go to S-11,
 Replace $\hat{\xi} \leftarrow \xi$,
 Liquidate the smallest block (optional),
 Go to S-7;
- S-11. Output solution,
 Calculate efficiency measures;
- end* {ZODIAC}.

6.6 Problem Generator

In order to provide a critical analysis of the cell formation techniques, the evaluation should be based on common test problems representing real life situations rather than specific, small-sized test problems. A random problem generator is used to evaluate and compare the existing cell formation techniques. The problem generator initially produces a machine-part incidence matrix and then generates operation sequences and standard times for each and every part. Thereafter, annual operating costs, availabilities of machine types, and annual demands of parts, which determine the work-load matrix, are created. The number of individual machines of each type and total work-load cost of each part are computed from the work-load matrix. Finally, the efficiency measures for the generated shop are calculated.

The incidence matrix indicates the size and the density of the original shop, and the place of the generated shop in the *manufacturing spectrum*. The manufacturing spectrum has two boundaries: ideal job shop and ideal Cellular Manufacturing shop. In an ideal job shop, all parts do use all machines. On the other hand, ideal Cellular Manufacturing shop involves mutually separated and therefore independent cells. Four parameters are required to construct a machine-part incidence matrix. The first two, the number of parts and the number of machine types in the system, are *size* parameters showing the size of the shop to be generated. The other two are *shape* parameters. One is the density which is actually the ratio of 'ones' in the incidence matrix to the total area. The last

parameter is clumpiness which identifies the location of the generated shop on the manufacturing spectrum. The clumpiness parameter shows the degree of possible block-diagonalization in the machine-part incidence matrix. This parameter always takes positive values. An ideal job shop can be realized by specifying the density parameter as one. Ideal Cellular Manufacturing systems can be generated by assigning low values such as 0.10 to the density parameter, and high values such as 1000 to the clumpiness parameter.

A detailed analysis of the clumpiness parameter requires the definition of inner-cell density and off-diagonal density. These definitions are used only in explaining the random problem generation module, and they are not general definitions that apply to cell formation problems. Let,

T : number of machine types (number of rows of incidence matrix),

P : number of parts (number of columns of incidence matrix),

k : number of imaginary cells (number of possible diagonal blocks in incidence matrix),

c : clumpiness parameter,

d : overall density of the shop (density of incidence matrix),

d_I : inner-cell density of imaginary cells (density of block-diagonal area in incidence matrix),

d_O : density of exceptional elements (density of off-diagonal area in incidence matrix),

\bar{t} : average number of different machine types per cell,

\bar{p} : average number of different parts per cell.

The number of 'ones' in the incidence matrix, E , can be calculated in the following two ways:

$$E = T \times P \times d,$$

$$E = (k \times \bar{t} \times \bar{p}) \times d_I + ((T \times P) - (k \times \bar{t} \times \bar{p})) \times d_O.$$

If full block diagonal machine part incidence matrix is square, the overall density d is exactly the inverse of the number of diagonal blocks k . With this particular

c	d	d_O	d_I	c	d	d_O	d_I
1	0.10	0.1000	0.1000	6	0.10	0.0166	0.8500
	0.15	0.1500	0.1500		0.15	0.0250	0.8583
	0.20	0.2000	0.2000		0.20	0.0333	0.8687
2	0.10	0.0500	0.5500	7	0.10	0.0143	0.8714
	0.15	0.0750	0.5750		0.15	0.0214	0.8786
	0.20	0.1000	0.6000		0.20	0.0286	0.8857
3	0.10	0.0333	0.7000	8	0.10	0.0125	0.8875
	0.15	0.0500	0.7167		0.15	0.0188	0.8938
	0.20	0.0667	0.7333		0.20	0.0250	0.9000
4	0.10	0.0250	0.7750	9	0.10	0.0111	0.9000
	0.15	0.0375	0.7875		0.15	0.0167	0.9056
	0.20	0.0500	0.8000		0.20	0.0222	0.9111
5	0.10	0.0200	0.8200	10	0.10	0.0100	0.9100
	0.15	0.0300	0.8300		0.15	0.0150	0.9150
	0.20	0.0400	0.8400		0.20	0.0200	0.9200

Table 6.4: Effect of shape parameters on inner-cell and off-diagonal densities.

situation in mind, if we assume that the number of imaginary cells is inversely proportional to the overall density i.e., $k \doteq \frac{1}{d}$, from the above equations we have

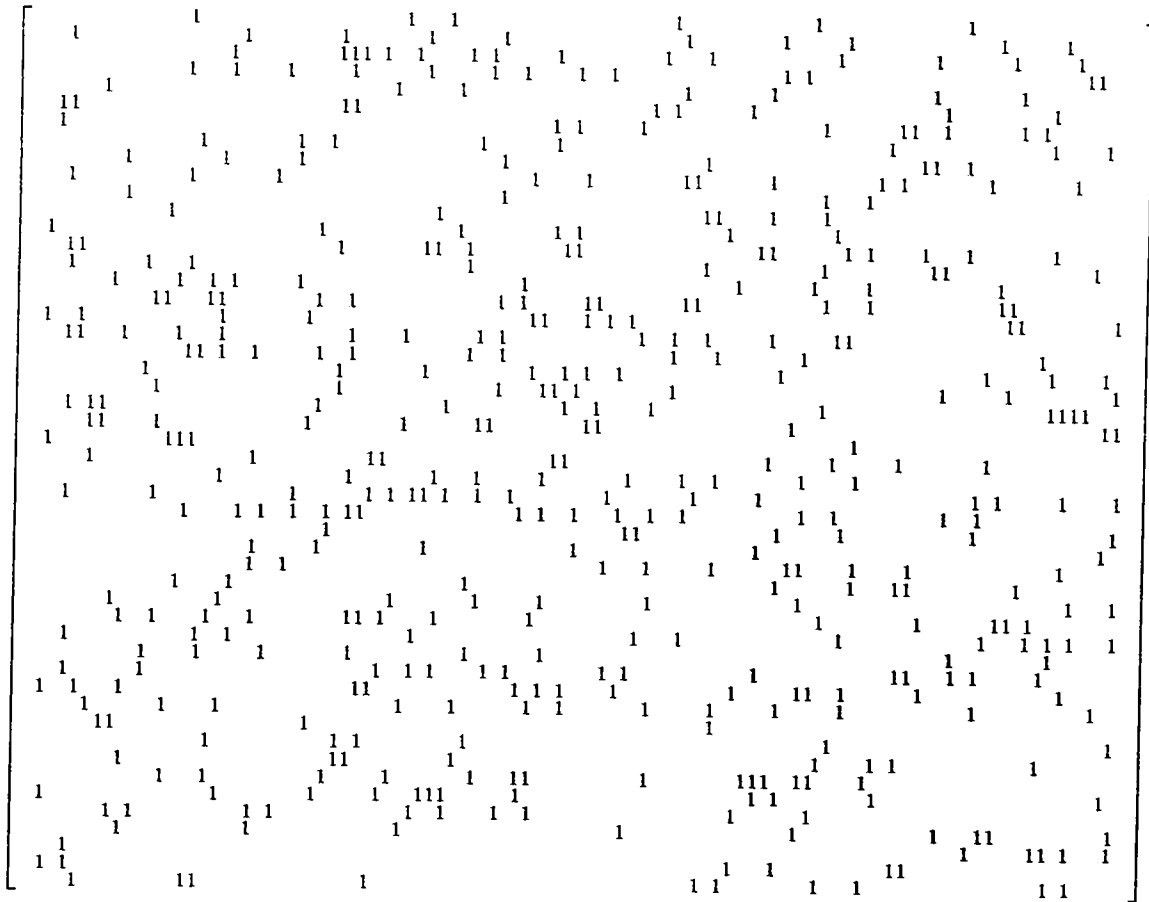
$$d = d \times d_I + (1 - d) \times d_O,$$

which can be solved for d_O and d_I by introducing a new parameter c as:

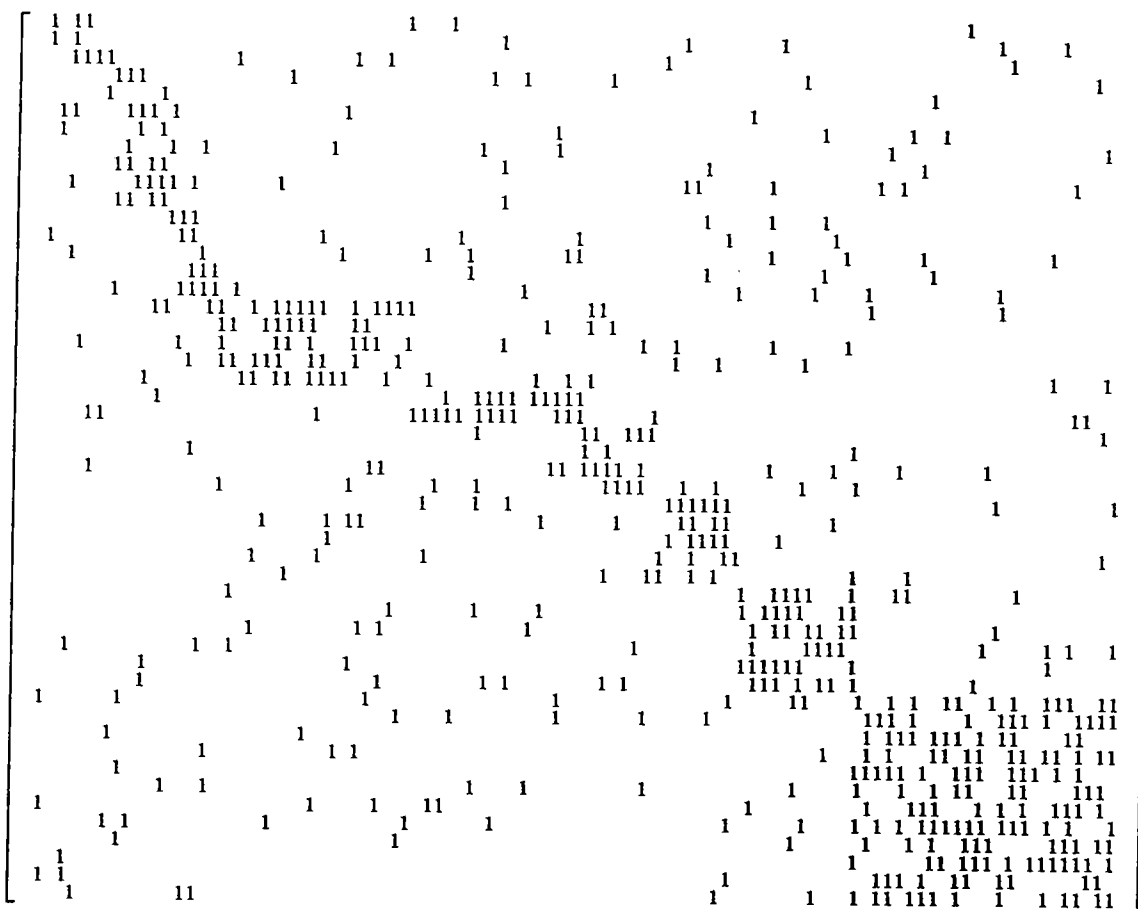
$$d_O = \frac{d}{c}, \quad d_I = 1 - \frac{1-d}{c}.$$

Thus, the clumpiness parameter and the overall density give rise to inner-cell and off-diagonal densities. The effect of the shape parameters on the densities is illustrated in Table 6.4. It can be observed from the table that the effect of the clumpiness parameter on both of the individual densities is more significant than that of the overall density. Some examples of the generated incidence matrices in relation to various clumpiness values are given in Figure 6.19.

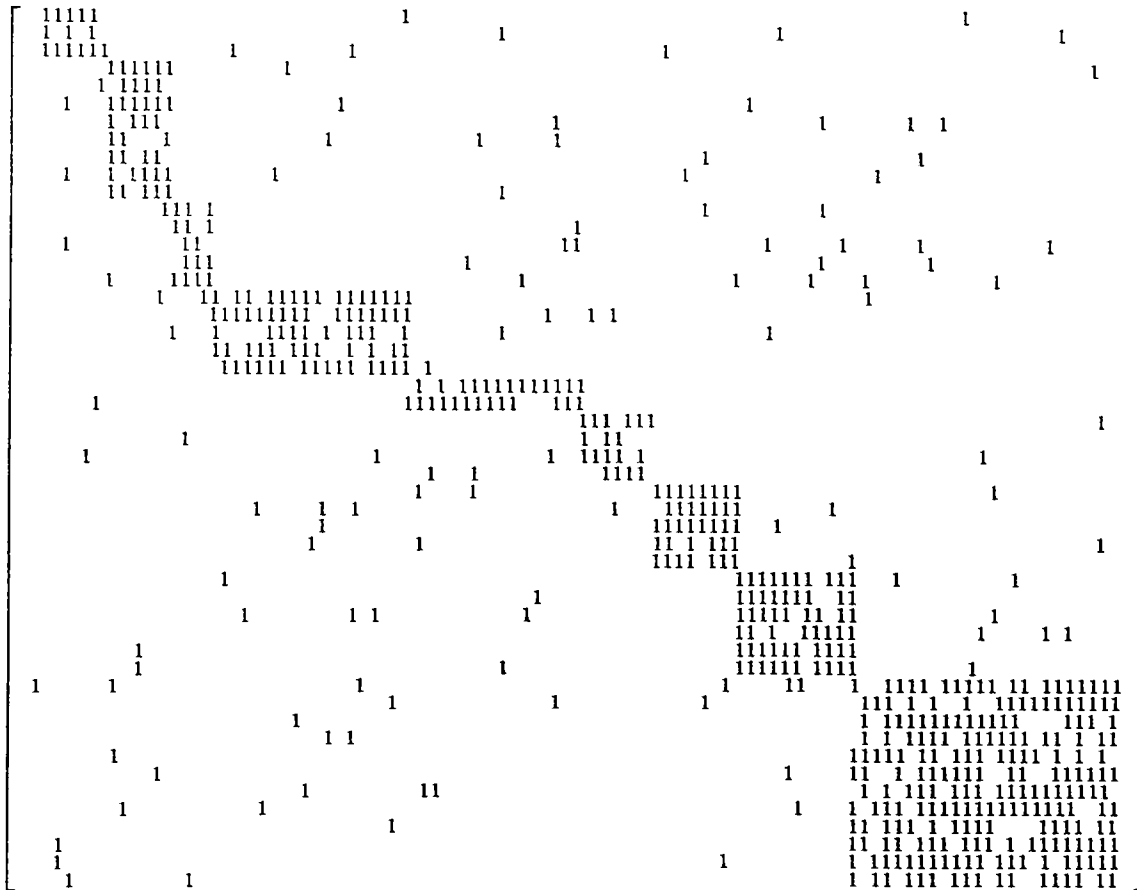
Once density and clumpiness values and the sizes of the problem are set, the incidence matrix can well be generated according to inner-cell and off-diagonal



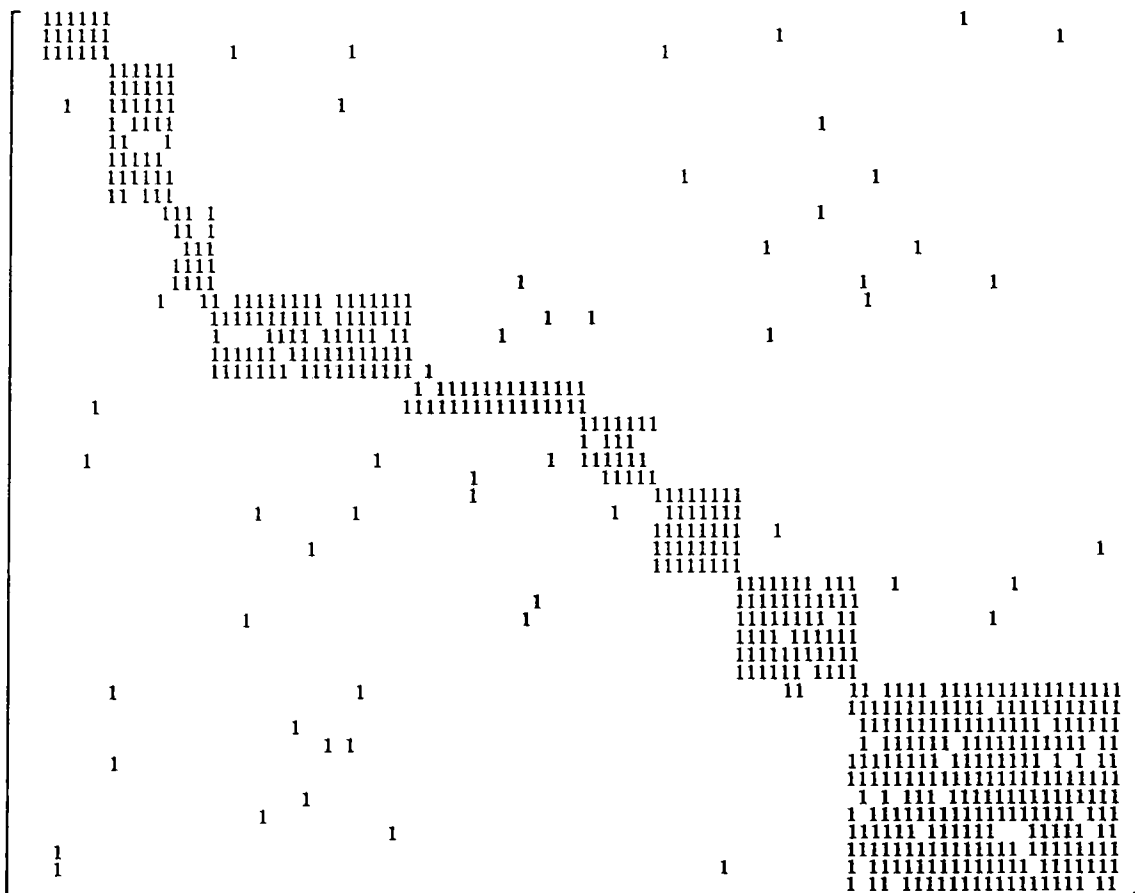
a) Clumpiness=1



b) Clumpiness=2



c) Clumpiness=4



d) Clumpiness=9

Figure 6.19: Effect of clumpiness on generated incidence matrices.

densities. After determining the number of imaginary cells k , the average number of different machine types ($\bar{t} = T \cdot d$) and parts ($\bar{p} = P \cdot d$) in each imaginary cell is calculated. Then the imaginary diagonal blocks are generated in random with sizes around these average values without changing T and P limits. For each entry of the incidence matrix, a random number between 0 and 1 is generated, and if it is less than the inner-cell (or off-diagonal) density then the element will be 1; if not, it will be 0. Finally, the rows and columns of the incidence matrix are permuted randomly to eliminate a trivial cell formation solution.

Subsequent to the incidence matrix, a corresponding work-load matrix is formed. At this stage, standard times and annual demands of parts are generated. In the meantime, annual available capacities of machine types are produced. Afterwards, the work-load matrix is computed. Finally, the number of machines of each type and total work-load costs of each part are calculated from the work-load matrix.

6.7 Experimental Design

In the thesis, the performance of the selected cell formation techniques are tested under randomly generated conditions. In order to reduce the size of the experiment, some input parameters of the generator are fixed. The varying factors in the experiment are the number of parts, density of the shop, clumpiness, demand of parts, and machine operating costs.

A summary of the factors and their levels in the experimentation is presented in Table 6.5. Working of the techniques under different problem sizes are analyzed by setting the number of parts to two levels. Both of the shape parameters which affect the type of the shop being generated are altered at different levels. The selected densities are based on the results of example problems and specific implementations in the literature. Densities between 0.10 and 0.20 are found to represent different scenarios adequately. Small values for the clumpiness parameter change the shape of incidence matrices substantially. However, the greater the clumpiness value, the smaller the change in the shape. Therefore, the levels

FACTORS		VALUES		#
<i>Size Parameters</i>	Number of machine types	$T = 50$		1
	Number of parts	$P = 100, 150$		2
<i>Shape Parameters</i>	Density	$d = .10, .15, .20$		3
	Clumpiness	$c = 1, 2, 4, 9$		4
<i>Distribution Parameters</i>	Annual part demands	High variance	$U[50; 4050]$	2
		Low variance	$U[1050; 3050]$	
	Annual machine operating cost	High variance	$U[100; 100100]$	2
		Low variance	$U[25050; 75050]$	
<i>Total number of selected PF/MG-F techniques</i>				8

Table 6.5: Factors of the experiment.

ALGORITHM	FACTOR	VALUE
<i>COMBGR</i>	Maximum machine difference limit	7
<i>MODROC</i>	Lower limit on similarities	0.75
	Upper limit on independent pairs	5
<i>MACE</i>	Threshold values	0.10
	Job shop like	<i>SCTF</i>
	Intermediate	<i>PSC</i>
	Ideal CM like	<i>SC</i>
<i>ZODIAC</i>	Weighting factor	0.50
	Threshold value	7
<i>WUBC</i>	Cell admission factor	0.60
	Cell size upper limit	50
	Key machine selection rule	A4
	Part assignment rule	B2
<i>CAA</i>	Cell admission factor	0.20
	Extra factor on cell size limit	0
<i>HAPCUT</i>	Approximation type	Clique
	Threshold value	0.75
<i>SIBC</i>	Cplex routine	Barrier
	Threshold value	0.80

Table 6.6: The best fine tuning values of cell formation techniques.

considered are 1, 2, 4, and 9. The effect of these values on a specific incidence matrix is illustrated in Figure 6.19. Finally, both demand and operating cost factors are analyzed at two levels, high variance and low variance.

Each of the eight selected techniques requires specific parameter settings. The best fine-tuned values of these parameters are illustrated in Table 6.6. In the

evaluation process, each technique is analyzed under 96 different scenarios. Ten different statistically independent cell formation problems are generated for each combination of the factors. Hence, the number of runs in the experimentation amounts to 7680:

$$10 \times 8 \times 2 \times 3 \times 4 \times 2 \times 2 = 7680.$$

6.8 Results and Discussion

For each combination of the factors, the work-load balance index, the under-utilization index, and the modified grouping efficiency index, which is determined by inter-cell flow efficiency and inner-cell density, are computed. The analyses of variance (ANOVA) for all efficiency measures are listed in Tables 6.7, 6.8, 6.9, and 6.10. The significance level for all tests is 0.05. In the tables, the statistically significant factors are shown by boldface letters. ANOVA tables for each measure are analyzed as follows:

The ANOVA analysis for inter-cell flow is presented in Table 6.7. According to the \mathcal{F} test, the main effects of all the factors except for the operating cost are considerably significant. The main effects of the algorithms and clumpiness together with their cross effect are the most significant. Furthermore, almost all of the 2-way, 3-way and 4-way interaction of the algorithms, the number of parts, density and clumpiness are statistically significant.

The same conclusions can be drawn for inner-cell density as can be seen from Table 6.8. The main effects of algorithms, number of parts, density and clumpiness are significant together with their cross effects. One should notice the small error sum of squares in the two parts of the modified grouping measure. This indicates that the factors and their interactions explain 96.63 % of the variability in inter-cell flow, and 97.17 % in inner-cell density measure.

According to the ANOVA analysis given in Table 6.9, the main effects of all the factors except for the operating cost on the work-load balance measure are considerably significant. The main effects of the algorithms, the number of parts

Source	Sum of Square	Dof	Mean Square	F Value	$F_{dof, \infty}$
x	1296211.325	7	185173.046	13750.691	2.01
y	7658.687	1	7658.687	568.723	3.84
z	24.029	1	24.029	1.784	3.84
t	107.768	1	107.768	8.003	3.84
u	2920.325	2	1460.162	108.430	3.00
v	256134.565	3	85378.188	6340.065	2.60
xy	18763.025	15	1250.868	92.888	1.67
xz	2032.187	15	135.479	10.060	1.67
xt	377.386	15	25.159	1.868	1.67
xu	129320.044	23	5622.611	417.527	1.53
xv	5129826.222	31	165478.265	12288.185	1.45
yz	51.682	3	17.227	1.279	2.60
yt	31.268	3	10.423	0.774	2.60
yu	2428.057	5	485.611	36.061	2.21
yv	483.837	7	69.120	5.133	2.01
zt	4.688	3	1.563	0.116	2.60
zu	16.537	5	3.307	0.246	2.21
zv	141.311	7	20.187	1.499	2.01
tu	19.860	5	3.972	0.295	2.21
tv	37.860	7	5.409	0.402	2.01
uv	25371.433	11	2306.494	171.277	1.79
xyz	91.019	31	2.936	0.218	1.45
xyt	295.797	31	9.542	0.709	1.45
xyu	12622.086	47	268.555	19.943	1.37
xyv	11467.137	63	182.018	13.516	1.32
xzt	39.052	31	1.260	0.094	1.45
xzu	245.783	47	5.229	0.388	1.37
xzv	627.080	63	9.954	0.739	1.32
xtu	193.169	47	4.110	0.305	1.37
xtv	321.381	63	5.101	0.379	1.32
xuv	165912.384	95	1746.446	129.689	1.27
yzt	6.863	7	0.980	0.073	2.01
yzu	4.774	11	0.434	0.032	1.79
yzv	30.636	15	2.042	0.152	1.67
ytu	32.228	11	2.930	0.218	1.79
ytv	23.357	15	1.557	0.116	1.67
yuv	5687.348	23	247.276	18.362	1.53
ztu	5.664	11	0.515	0.038	1.79
ztv	34.576	15	2.305	0.171	1.67
zuv	23.408	23	1.018	0.076	1.53
tuv	47.456	23	2.063	0.153	1.53
xyzt	21.737	63	0.345	0.026	1.32
xyzu	248.431	95	2.615	0.194	1.27
xyzv	197.855	127	1.558	0.116	1.22
xytu	196.743	95	2.071	0.154	1.27
xytv	357.438	127	2.814	0.209	1.22
xyuv	38085.928	191	199.403	14.807	1.15
xztu	219.002	95	2.305	0.171	1.27
xztv	173.171	127	1.364	0.101	1.22
xzuv	283.577	191	1.485	0.110	1.15
xtuv	398.604	191	2.087	0.155	1.15
yztu	37.053	23	1.611	0.120	1.53
yztv	42.523	31	1.372	0.102	1.45
yzuv	120.228	47	2.558	0.190	1.37
ytuv	19.225	47	0.409	0.030	1.37
ztuv	92.456	47	1.967	0.146	1.37
xyztu	100.235	191	0.525	0.039	1.15
xyztv	146.674	255	0.575	0.043	1.10
xyzuv	473.997	383	1.238	0.092	1.07
xytuv	602.671	383	1.574	0.117	1.07
xztuv	435.105	383	1.136	0.084	1.07
yztuv	107.696	95	1.134	0.084	1.27
xyztuv	705.782	767	0.971	0.072	1.05
ERROR	93066.661	6911	13.466	1.000	1.01

x: Algorithms y: Number of Parts z: Operating Cost
t: Demand u: Density v: Clumpiness

Table 6.7: Analysis of Variance table for inter-cell flow.

Source	Sum of Square	Dof	Mean Square	F Value	$F_{dof, \infty}$
x	2942538.124	7	420362.589	21550.822	2.01
y	1860.267	1	1860.267	95.371	3.84
z	60.140	1	60.140	3.083	3.84
t	46.932	1	46.932	2.406	3.84
u	40892.650	2	20446.325	1048.226	3.00
v	467271.209	3	155757.070	7985.232	2.60
xy	1776.568	15	118.438	6.072	1.67
xz	1673.157	15	111.544	5.719	1.67
xt	1211.896	15	80.793	4.142	1.67
xu	115172.338	23	5007.493	256.720	1.53
xv	7969878.644	31	257092.859	13180.438	1.45
yz	83.424	3	27.808	1.426	2.60
yt	100.851	3	33.617	1.723	2.60
yu	1076.548	5	215.310	11.038	2.21
yv	2267.108	7	323.873	16.604	2.01
zt	11.709	3	3.903	0.200	2.60
zu	156.524	5	31.305	1.605	2.21
zv	140.930	7	20.133	1.032	2.01
tu	56.253	5	11.251	0.577	2.21
tv	347.566	7	49.652	2.546	2.01
uv	18628.770	11	1693.525	86.822	1.79
xyz	434.293	31	14.009	0.718	1.45
xyt	331.819	31	10.704	0.549	1.45
xyu	16088.389	47	342.306	17.549	1.37
xyv	18383.902	63	291.808	14.960	1.32
xzt	446.530	31	14.404	0.738	1.45
xzu	427.204	47	9.089	0.466	1.37
xzv	1212.707	63	19.249	0.987	1.32
xtu	946.475	47	20.138	1.032	1.37
xtv	1511.570	63	23.993	1.230	1.32
xuv	130354.701	95	1372.155	70.347	1.27
yzt	8.806	7	1.258	0.064	2.01
yzu	273.372	11	24.852	1.274	1.79
yzv	259.624	15	17.308	0.887	1.67
ytu	30.187	11	2.744	0.141	1.79
ytv	148.274	15	9.885	0.507	1.67
yuv	1179.196	23	51.269	2.628	1.53
ztu	257.600	11	23.418	1.201	1.79
ztv	221.201	15	14.747	0.756	1.67
zuv	296.790	23	12.904	0.662	1.53
tuv	267.927	23	11.649	0.597	1.53
xyzt	202.226	63	3.210	0.165	1.32
xyzu	942.196	95	9.918	0.508	1.27
xyzv	827.816	127	6.518	0.334	1.22
xytu	702.836	95	7.398	0.379	1.27
xytv	1075.566	127	8.469	0.434	1.22
xyuv	37612.908	191	196.926	10.096	1.15
xztu	737.450	95	7.763	0.398	1.27
xztv	945.676	127	7.446	0.382	1.22
xzuv	2432.935	191	12.738	0.653	1.15
xtuv	1760.887	191	9.219	0.473	1.15
yztu	18.365	23	0.798	0.041	1.53
yztv	76.109	31	2.455	0.126	1.45
yzuv	159.378	47	3.391	0.174	1.37
ytuv	388.249	47	8.261	0.423	1.37
ztuv	355.967	47	7.574	0.388	1.37
xyztu	677.528	191	3.547	0.182	1.15
xyztv	1283.046	255	5.032	0.258	1.10
xyzuv	2035.406	383	5.314	0.272	1.07
xytuv	2006.940	383	5.240	0.269	1.07
xztuv	1764.208	383	4.606	0.236	1.07
yztuv	616.940	95	6.494	0.333	1.27
xyztuv	2102.701	767	2.892	0.148	1.05
ERROR	134803.478	6911	19.506	1.000	1.01

x: Algorithms y: Number of Parts z: Operating Cost
t: Demand u: Density v: Clumpiness

Table 6.8: Analysis of Variance table for inner-cell density.

Source	Sum of Square	Dof	Mean Square	F Value	$F_{dof, \infty}$
x	195173.775	7	27881.968	3281.587	2.01
y	5166.960	1	5166.960	608.129	3.84
z	13.405	1	13.405	1.578	3.84
t	175.397	1	175.397	20.643	3.84
u	4558.743	2	2279.372	268.272	3.00
v	322.186	3	107.395	12.640	2.60
xy	2247.515	15	149.834	17.635	1.67
xz	586.653	15	39.110	4.603	1.67
xt	137.128	15	9.142	1.076	1.67
xu	10701.985	23	465.304	54.764	1.53
xv	785507.692	31	25338.958	2982.286	1.45
yz	46.759	3	15.586	1.834	2.60
yt	46.834	3	15.611	1.837	2.60
yu	41.464	5	8.293	0.976	2.21
yv	550.066	7	78.581	9.249	2.01
zt	22.897	3	7.632	0.898	2.60
zu	191.819	5	38.364	4.515	2.21
zv	62.834	7	8.976	1.056	2.01
tu	61.332	5	12.266	1.444	2.21
tv	224.256	7	32.037	3.771	2.01
uv	438.815	11	39.892	4.695	1.79
xyz	501.567	31	16.180	1.904	1.45
xyt	159.967	31	5.160	0.607	1.45
xyu	2059.063	47	43.810	5.156	1.37
xyv	3224.510	63	51.183	6.024	1.32
xzt	341.387	31	11.012	1.296	1.45
xzu	656.363	47	13.965	1.644	1.37
xzv	830.414	63	13.181	1.551	1.32
xtu	218.130	47	4.641	0.546	1.37
xtv	1370.404	63	21.752	2.560	1.32
xuv	6979.617	95	73.470	8.647	1.27
yzt	37.852	7	5.407	0.636	2.01
yzu	138.777	11	12.616	1.485	1.79
yzv	145.181	15	9.679	1.139	1.67
ytu	84.750	11	7.705	0.907	1.79
ytv	73.974	15	4.932	0.580	1.67
yuv	280.851	23	12.211	1.437	1.53
ztu	35.097	11	3.191	0.376	1.79
ztv	114.590	15	7.639	0.899	1.67
zuv	95.327	23	4.145	0.488	1.53
tuv	357.447	23	15.541	1.829	1.53
xyzt	270.999	63	4.302	0.506	1.32
xyzu	961.001	95	10.116	1.191	1.27
xyzv	641.922	127	5.055	0.595	1.22
xytu	335.418	95	3.531	0.416	1.27
xytv	1417.793	127	11.164	1.314	1.22
xyuv	2853.200	191	14.938	1.758	1.15
xztu	317.445	95	3.342	0.393	1.27
xztv	1476.052	127	11.622	1.368	1.22
xzuv	933.455	191	4.887	0.575	1.15
xtuv	1591.241	191	8.331	0.981	1.15
yztu	28.447	23	1.237	0.146	1.53
yztv	58.882	31	1.899	0.224	1.45
yzuv	155.623	47	3.311	0.390	1.37
ytuv	132.388	47	2.817	0.332	1.37
ztuv	964.644	47	20.524	2.416	1.37
xyztu	326.978	191	1.712	0.201	1.15
xyztv	258.856	255	1.015	0.119	1.10
xyzuv	763.974	383	1.995	0.235	1.07
xytuv	1049.152	383	2.739	0.322	1.07
xztuv	7262.543	383	18.962	2.232	1.07
yztuv	245.045	95	2.579	0.304	1.27
xyztuv	1763.350	767	2.426	0.285	1.05
ERROR	58719.230	6911	8.496	1.000	1.01

x: Algorithms y: Number of Parts z: Operating Cost
t: Demand u: Density v: Clumpiness

Table 6.9: Analysis of Variance table for work-load balance.

Source	Sum of Square	Dof	Mean Square	F Value	$F_{dof, \infty}$
x	330512.338	7	47216.048	6120.899	2.01
y	26440.470	1	26440.470	3427.637	3.84
z	647.499	1	647.499	83.939	3.84
t	184.364	1	184.364	23.900	3.84
u	8080.348	2	4040.174	523.752	3.00
v	41721.215	3	13907.072	1802.857	2.60
xy	25997.978	15	1733.199	224.685	1.67
xz	997.552	15	66.503	8.621	1.67
xt	711.559	15	47.437	6.150	1.67
xu	42211.504	23	1835.283	237.919	1.53
xv	750380.379	31	24205.819	3137.945	1.45
yz	21.869	3	7.290	0.945	2.60
yt	0.632	3	0.211	0.027	2.60
yu	3900.875	5	780.175	101.139	2.21
yv	1444.030	7	206.290	26.743	2.01
zt	65.599	3	21.866	2.835	2.60
zu	29.518	5	5.904	0.765	2.21
zv	429.998	7	61.428	7.963	2.01
tu	705.039	5	141.008	18.280	2.21
tv	386.802	7	55.257	7.163	2.01
uv	10066.521	11	915.138	118.635	1.79
xyz	762.764	31	24.605	3.190	1.45
xyt	695.404	31	22.432	2.908	1.45
xyu	13280.047	47	282.554	36.629	1.37
xyv	13216.581	63	209.787	27.196	1.32
xzt	175.900	31	5.674	0.736	1.45
xzu	299.943	47	6.382	0.827	1.37
xzv	1539.209	63	24.432	3.167	1.32
xtu	1709.953	47	36.382	4.716	1.37
xtv	1548.315	63	24.576	3.186	1.32
xuv	41187.286	95	433.550	56.204	1.27
yzt	5.758	7	0.823	0.107	2.01
yzu	37.599	11	3.418	0.443	1.79
yzv	686.742	15	45.783	5.935	1.67
ytu	239.044	11	21.731	2.817	1.79
ytv	306.227	15	20.415	2.647	1.67
yuv	3551.337	23	154.406	20.017	1.53
ztu	6.988	11	0.635	0.082	1.79
ztv	123.242	15	8.216	1.065	1.67
zuv	262.048	23	11.393	1.477	1.53
tuv	1826.725	23	79.423	10.296	1.53
xyzt	179.114	63	2.843	0.369	1.32
xyzu	675.066	95	7.106	0.921	1.27
xyzv	1741.333	127	13.711	1.777	1.22
xytu	1343.768	95	14.145	1.834	1.27
xytv	2407.259	127	18.955	2.457	1.22
xyuv	37595.838	191	196.837	25.517	1.15
xztu	1039.980	95	10.947	1.419	1.27
xztv	854.700	127	6.730	0.872	1.22
xzuv	2352.984	191	12.319	1.597	1.15
xtuv	4076.265	191	21.342	2.767	1.15
yztu	0.313	23	0.014	0.002	1.53
yztv	43.690	31	1.409	0.183	1.45
yzuv	576.931	47	12.275	1.591	1.37
ytuv	968.751	47	20.612	2.672	1.37
ztuv	177.651	47	3.780	0.490	1.37
xyztu	805.909	191	4.219	0.547	1.15
xyztv	662.554	255	2.598	0.337	1.10
xyzuv	3821.731	383	9.978	1.294	1.07
xytuv	3433.558	383	8.965	1.162	1.07
xztuv	1474.428	383	3.850	0.499	1.07
yztuv	171.425	95	1.804	0.234	1.27
xyztuv	2054.902	767	2.827	0.366	1.05
ERROR	53310.810	6911	7.714	1.000	1.01

x: Algorithms y: Number of Parts z: Operating Cost
t: Demand u: Density v: Clumpiness

Table 6.10: Analysis of Variance table for under-utilizations.

and density are the most significant ones. Another sources of variation are due to the cross effect of the algorithms and clumpiness, and due to the cross effect of algorithms and density. The other significant factors detected by means of the \mathcal{F} test have relatively small effects on work-load balances. The factors and their indicated interactions explain only 82.74 % of the variability in the randomly generated problems.

The ANOVA analysis for under-utilization is presented in Table 6.10. The majority of the rows are found to be statistically significant. All of the 1-way, 2-way, 3-way and even 4-way interactions of the algorithms, the number of parts, density and clumpiness have a considerable variation effect on the under-utilization measures. The other significant factors detected by means of the \mathcal{F} test have relatively small effects on under-utilizations. The factors and associated interactions explain 94.07 % of the variability in under-utilization values.

The results of pairwise comparisons among the factor levels are obtained by using Sheffée-type simultaneous 95 % confidence intervals. Since the number of parts, machine operating costs and demand factors have only two levels, the existing sources of variations are clearly due to these two. Table 6.11 presents the estimated mean differences $\hat{L}_{i,j}$ for the factor levels i and j . The 95 % confidence interval for these mean differences can be obtained from $\hat{L}_{i,j} \pm \mathcal{E}$, where \mathcal{E} stands for the error term and is given in the parenthesis for each factor. As an example, the mean inter-cell flow index difference between COMBGR and MODROC is considered. From the table, the 95% confidence interval for this difference is 26.9375 ± 0.6283 which is (26.3092, 27.5658). Since the confidence interval does not contain zero, a statistical difference is observed between the inter-cell flow efficiency means of COMBGR and MODROC. The other factors at each efficiency measure can be interpreted similarly. From this analysis, it is observed that only few pairwise comparisons prove insignificant and these are indicated in boldface letters in the table.

As a summary, the main sources of variation in all of the efficiency measures are due to the algorithms, clumpiness, the number of parts, density and their interactions. The ANOVA tables indicate that the levels of these factors and

their interactions do have different means. However, they do not identify the levels of the factors which are significantly different than the others. In order to illustrate such differences, the mean values of these levels are plotted in Figures 6.20, 6.21, and 6.22.

A summary of the important findings for each technique is given below:

1. CAA is the most insensitive technique regardless of the factors. Only a slight increase in the grouping efficiency value is observed as we increase clumpiness.
2. Because COMBGR generates solutions with no exceptional elements, machine duplications occur and this leads to:
 - 100% inter-cell flow efficiency;
 - low inner-cell density measures;
 - low work-load balance values;
 - high machine under-utilization values.
3. MACE creates
 - large number of small sized cells in low clumpiness values;
 - grouping efficiency values that are insensitive to alpha changes in the case of high clumpiness values;
 - good work-load balances;
 - high under-utilization figures in low clumpiness values;
4. It has been observed that MODROC solutions depend heavily on the first two ROC iterations. The presence of exceptional elements affects final merging iterations leading to solutions with:
 - low grouping efficiency values for low clumpiness and moderate values for high clumpiness values;

- satisfactory work-load balance and under-utilization scores, but not the best;
 - sensitivity to an introduction of new parts into manufacturing environment.
5. WUBC behaves like COMBGR except that it allows the existence of exceptional elements. WUBC provides solutions with:
- close grouping efficiency values for low clumpiness and worse for high clumpiness values;
 - higher work-load balance values due to clustering based on work-loads; better machine utilization measures.
6. Since ZODIAC is designed to generate a perfect block-diagonal structure, the technique does not perform well for low clumpiness values. For high clumpiness values, the solutions generated by ZODIAC have:
- robust grouping efficiency values in changing α values;
 - best grouping efficiency values;
 - high work-load balance measures;
 - low machine under-utilizations.
7. HAPCUT has flat efficiency plots which shows a good balance of the inner-cell density and the inter-cell flow efficiency. HAPCUT
- performs relatively well in terms of the modified grouping efficiency when the density is high and α has moderate values. When the clumpiness increases, the plot shifts upwards.
 - its modified efficiency scores are almost insensitive to density changes, and it is robust in changes in the number of parts.
 - has reasonable work-load balance scores. When the number of parts is increased, the variability of the work-load balance is increased.

- has relatively high under-utilization values. They are increased as clumpiness increases.
8. SIBC has low inner-cell density values as compared to inter-cell flow values. As clumpiness increases, the slope its efficiency plot reduces. SIBC
- has a good performance in terms of the modified grouping efficiency in low clumpiness values especially in cases clumpiness is equal to 2.
 - has lower inter-cell flow values when the number of parts is increased.
 - has better work-load balance values when the number of parts is increased.
 - has lower under-utilization values when the clumpiness is increased. As the number of parts is increased, the under-utilization scores are worse off. If the density decreases, the under-utilization scores are better off. increased.

The best cell formation technique suggested for each combination of the shape parameters in terms of the grouping efficiency measure are presented in Table 6.12. The best technique for each clumpiness-density pair is extracted from the associated plot. The techniques with results very close to the best score are also included in the related entry in the table. Similarly, the best cell formation techniques in terms of the work-load balance measure and the under-utilization values are given in Table 6.13.

In relation to the grouping efficiency, MACE seems to be the best technique for low clumpiness values if inner-cell densities are considered more critical than inter-cell flow efficiencies. As clumpiness increased a bit, HAPCUT joins MACE. COMBGR and WUBC are the best when inter-cell flows gain importance. As clumpiness increases, ZODIAC dominates the other techniques, even at moderate to high α values. Nevertheless, COMBGR is still the best alternative provided that the presence of exceptional elements are totally undesirable. If the two extremes are in balance i.e., inner-cell densities and inter-cell flows are equally important, ZODIAC and SIBC should be preferred. Hence, MACE, COMBGR,

and WUBC are the best for low clumpiness whereas ZODIAC and HAPCUT and SIBC are the best for high clumpiness values.

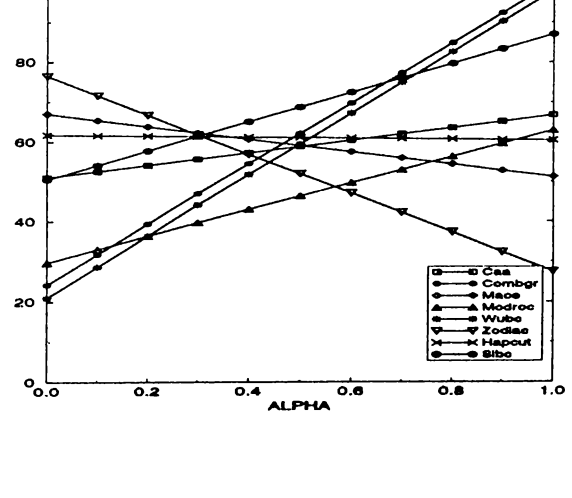
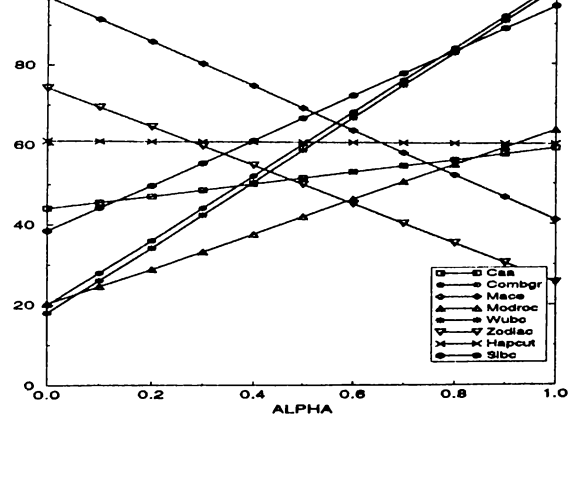
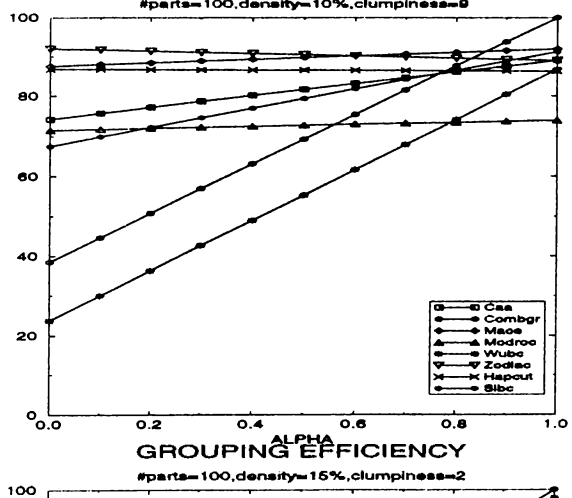
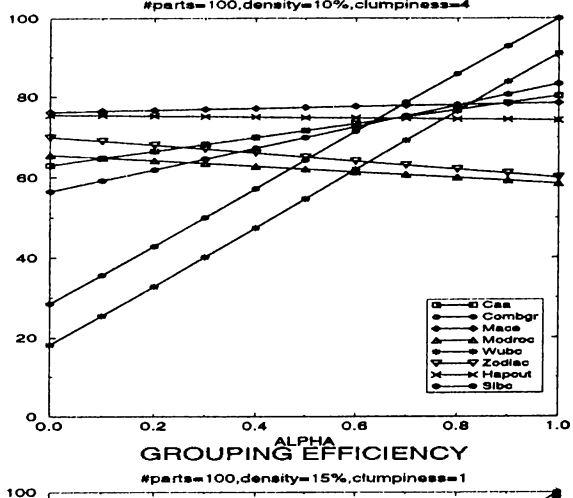
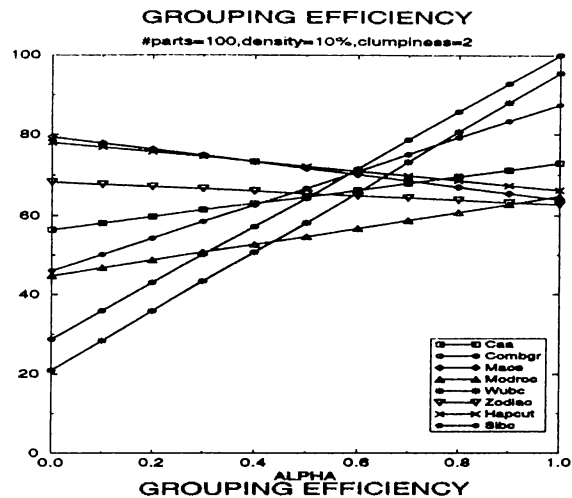
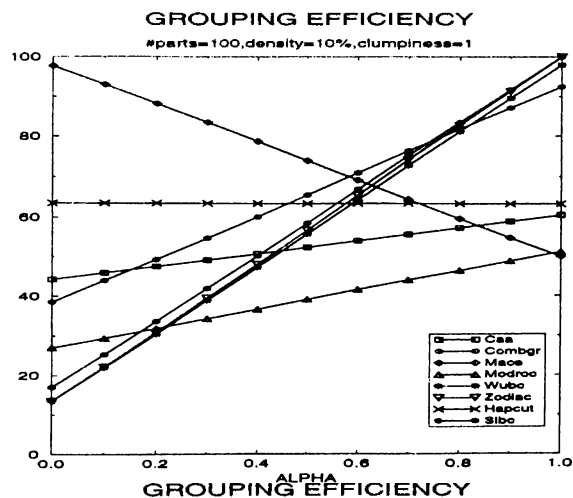
The cell formation solutions generated by COMBGR and CAA are inferior in terms of work-load balances. The other four techniques except HAPCUT and SIBC, however, perform reasonably well. In almost all of the cases, WUBC performs best in terms of the under-utilization. In addition to WUBC, MODROC and ZODIAC generate good cell formation solutions in terms of under-utilization measure. For high clumpiness values, MACE appears to have a good performance.

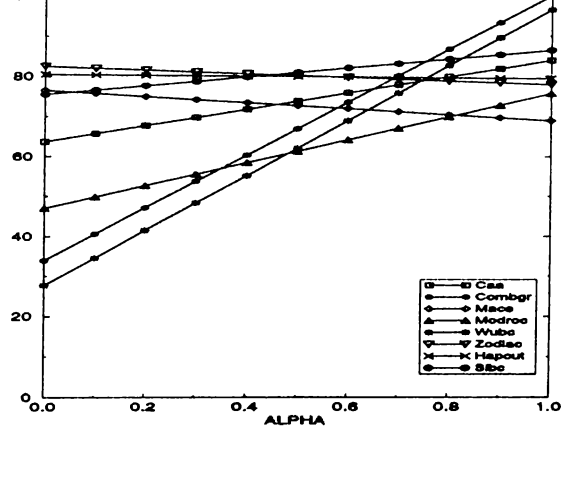
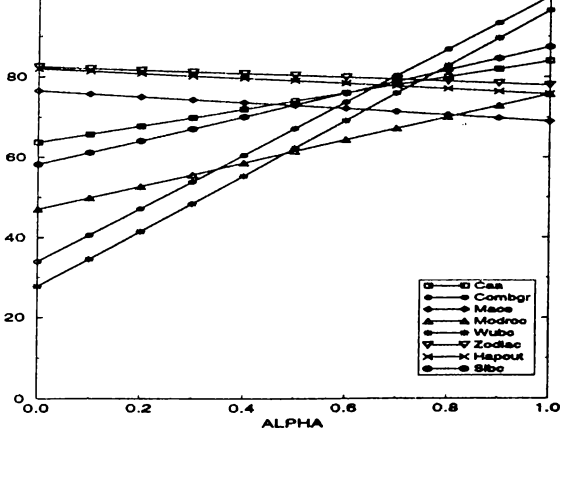
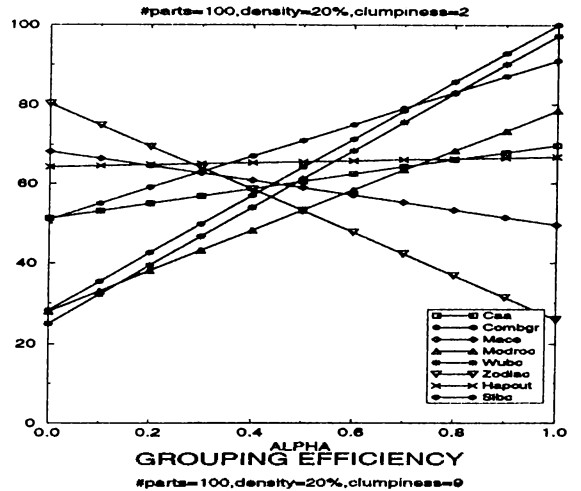
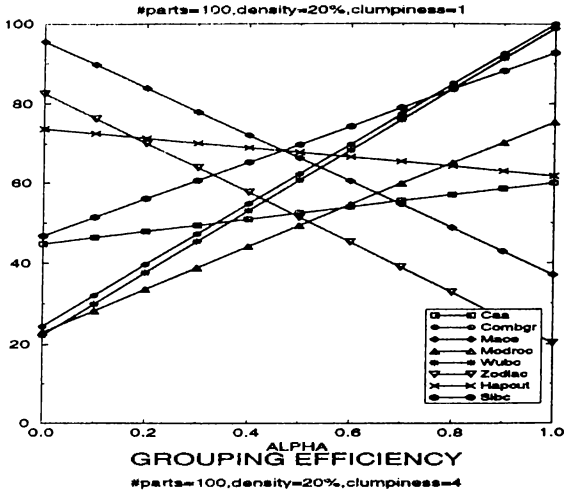
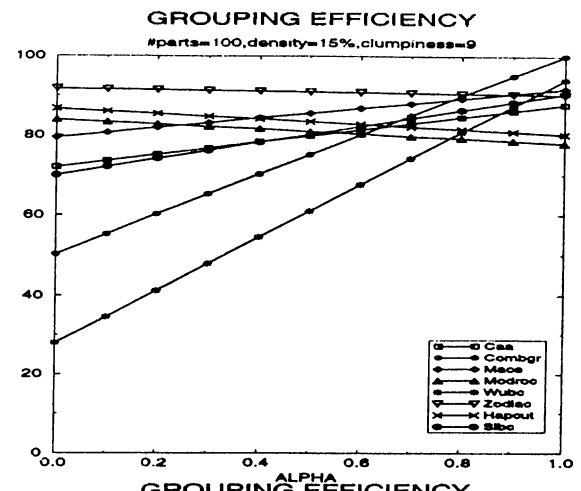
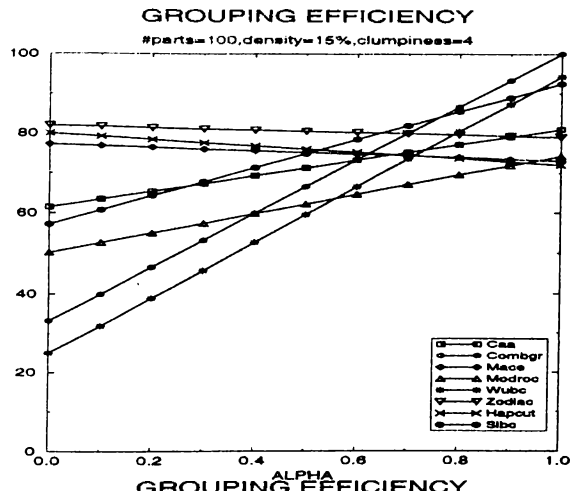
	Inter-cell flow	Inner-cell density	Work-load balance	Under-utilization
ALGORITHM	(0.6283)	(0.7561)	(0.4991)	(0.4755)
COMBGRvsMODROC	100.000-73.062= 26.938	31.417-48.000=-16.584	72.864-86.438=-13.573	21.729- 8.125= 13.604
COMBGRvsWUBC	100.000-96.010= 3.990	31.417-22.896= 8.521	72.864-86.417=-13.552	21.729- 6.073= 15.656
COMBGRvsCAA	100.000-76.260= 23.740	31.417-58.438=-27.021	72.864-78.458=- 5.594	21.729-15.021= 6.708
COMBGRvsMACE	100.000-64.385= 35.615	31.417-73.510=-42.094	72.864-86.646= -13.781	21.729-16.292= 5.438
COMBGRvsZODIAC	100.000-64.938= 35.062	31.417-81.240=-49.823	72.864-85.458= -12.594	21.729-14.635= 7.094
COMBGRvsHAPCUT	100.000-70.771= 29.229	31.417-74.750=-43.333	72.864-76.938= - 4.073	21.729-26.833= - 5.104
COMBGRvsSIBC	100.000-88.937= 11.063	31.417-55.271=-23.854	72.864-78.677= - 5.813	21.729-21.594= 0.135
MODROCvsWUBC	73.062-96.010=-22.948	48.000-22.896= 25.105	86.438-86.417= 0.021	8.125- 6.073= 2.052
MODROCvsCAA	73.062-76.260=- 3.198	48.000-58.438=-10.437	86.438-78.458= 7.979	8.125-15.021= - 6.896
MODROCvsMACE	73.062-64.385= 8.677	48.000-73.510=-25.510	86.438-86.646= - 0.208	8.125-16.292= - 8.167
MODROCvsZODIAC	73.062-64.938= 8.125	48.000-81.240=-33.239	86.438-85.458= 0.979	8.125-14.635= - 6.510
MODROCvsHAPCUT	73.062-70.771= 2.292	48.000-74.750=-26.750	86.438-76.938= 9.500	8.125-26.833= -18.708
MODROCvsSIBC	73.062-88.937=-15.875	48.000-55.271=- 7.270	86.438-78.677= 7.760	8.125-21.594= -13.469
WUBCvsCAA	96.010-76.260= 19.750	22.896-58.438=-35.542	86.417-78.458= 7.958	6.073-15.021= - 8.948
WUBCvsMACE	96.010-64.385= 31.625	22.896-73.510=-50.615	86.417-86.646= - 0.229	6.073-16.292= -10.219
WUBCvsZODIAC	96.010-64.938= 31.072	22.896-81.240=-58.344	86.417-85.458= 0.689	6.073-14.635=- 8.562
WUBCvsHAPCUT	96.010-70.771= 25.239	22.896-74.750=-51.914	86.417-76.938= 9.209	6.073-26.833=-20.760
WUBCvsSIBC	96.010-88.937= 7.073	22.896-55.271=-32.435	86.417-78.677= 7.470	6.073-21.594=-15.521
CAAvsMACE	76.260-64.385= 11.875	58.438-73.510=-15.072	78.458-86.646=- 8.188	15.021-16.292=- 1.271
CAAvsZODIAC	76.260-64.938= 11.332	58.438-81.240=-22.802	78.458-85.458= -7.000	15.021-14.635= 0.386
CAAvsHAPCUT	76.260-70.771= 5.489	58.438-74.750=-16.312	78.458-76.938= 1.520	15.021-26.833=-11.812
CAAvsSIBC	76.260-88.937=-12.677	58.438-55.271= 3.167	78.458-78.677=- 0.219	15.021-21.594=- 6.573
MACEvsZODIAC	64.385-64.938=- 0.553	73.510-81.240= - 7.730	86.646-85.458= 1.188	16.292-14.635= 1.657
MACEvsHAPCUT	64.385-70.771= - 6.386	73.510-74.750=- 1.240	86.646-76.938= 9.708	16.292-26.833=-10.541
MACEvsSIBC	64.385-88.937=-24.552	73.510-55.271= 18.239	86.646-78.677= 7.969	16.292-21.594=- 5.302
ZODIACvsHAPCUT	64.938-70.771=- 5.833	81.240-74.750= 6.490	85.458-76.938= 8.520	14.635-26.833=-12.198
ZODIACvsSIBC	64.938-88.937=-29.999	81.240-55.271= 25.969	85.458-78.677= 6.871	14.635-21.594=- 6.959
HAPCUTvsSIBC	70.771-88.937=-18.166	74.750-55.271= 19.479	76.938-78.677=- 1.739	26.833-21.594= 5.239
# PARTS	♠	♠	♠	♠
100 vs 150	78.297- 80.294= -1.997	55.198 -56.182= -0.984	80.667 -82.307= 1.640	18.143 -14.432= 3.711
OPER. COST	♠	♠	♣	♠
Low vs High	79.351 -79.240= 0.111	55.779 -55.602= 0.177	81.445 -81.529= -0.084	15.997 -16.578= -0.581
DEMAND	♣	♣	♠	♠
Low vs High	79.177 -79.414= 0.237	55.612 -55.768= -0.156	81.336 -81.638= -0.302	16.443 -16.133= 0.310
DENSITY	(0.3847)	(0.4630)	(0.3056)	(0.2912)
10 vs 15	80.129-79.102= 1.027	52.859-55.699=-2.840	80.422-81.820=-1.398	17.738-15.574= 2.164
10 vs 20	80.129-78.656= 1.473	52.859-58.512=-5.652	80.422-82.219=-1.797	17.738-15.551= 2.188
15 vs 20	79.102-78.656= 0.445	55.699-58.512=-2.812	81.820-82.219=-0.399	15.574-15.551= 0.023
CLUMPINESS	(0.4443)	(0.5347)	(0.3529)	(0.3362)
1 vs 2	72.401-72.750=- 0.349	45.630-47.469=- 1.839	81.740-81.396= 0.344	19.677-17.776= 1.901
1 vs 4	72.401-82.406=-10.005	45.630-59.396=-13.766	81.740-81.255= 0.485	19.677-14.677= 5.000
1 vs 9	72.401-89.625=-17.224	45.630-70.266=-24.636	81.740-81.557= 0.183	19.677-13.021= 6.656
2 vs 4	72.750-82.406=- 9.656	47.469-59.396=-11.927	81.396-81.255= 0.141	17.776-14.677= 3.099
2 vs 9	72.750-89.625=-16.875	47.469-70.266=-22.797	81.396-81.557=- 0.161	17.776-13.021= 4.755
4 vs 9	82.406-89.625=- 7.219	59.396-70.266=-10.870	81.255-81.557= - 0.302	14.677-13.021= 1.656

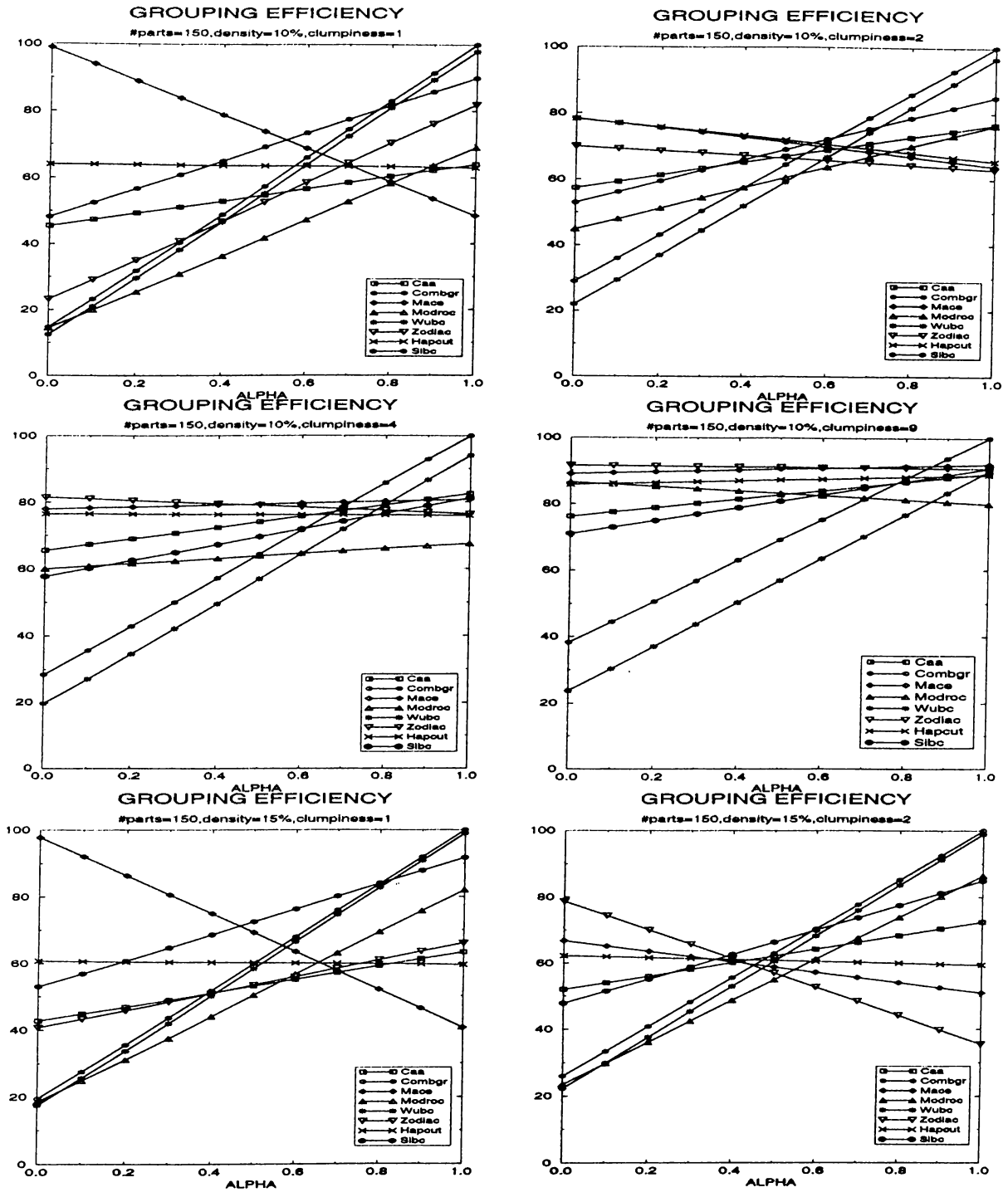
♠: The mean difference of the two factor levels is significant.

♣: The mean difference of the two factor levels is insignificant.

Table 6.11: Estimated mean differences for the factor levels.







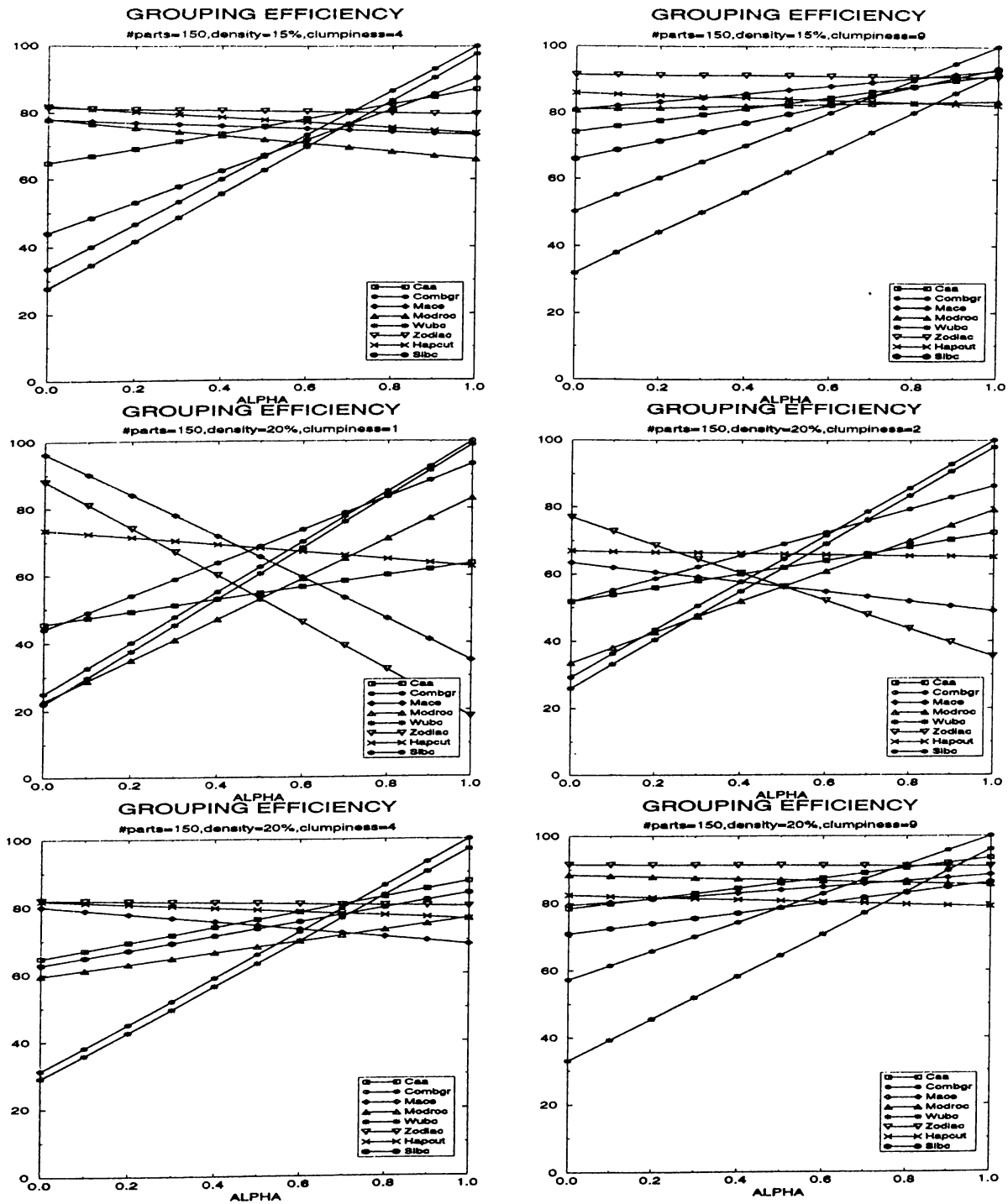
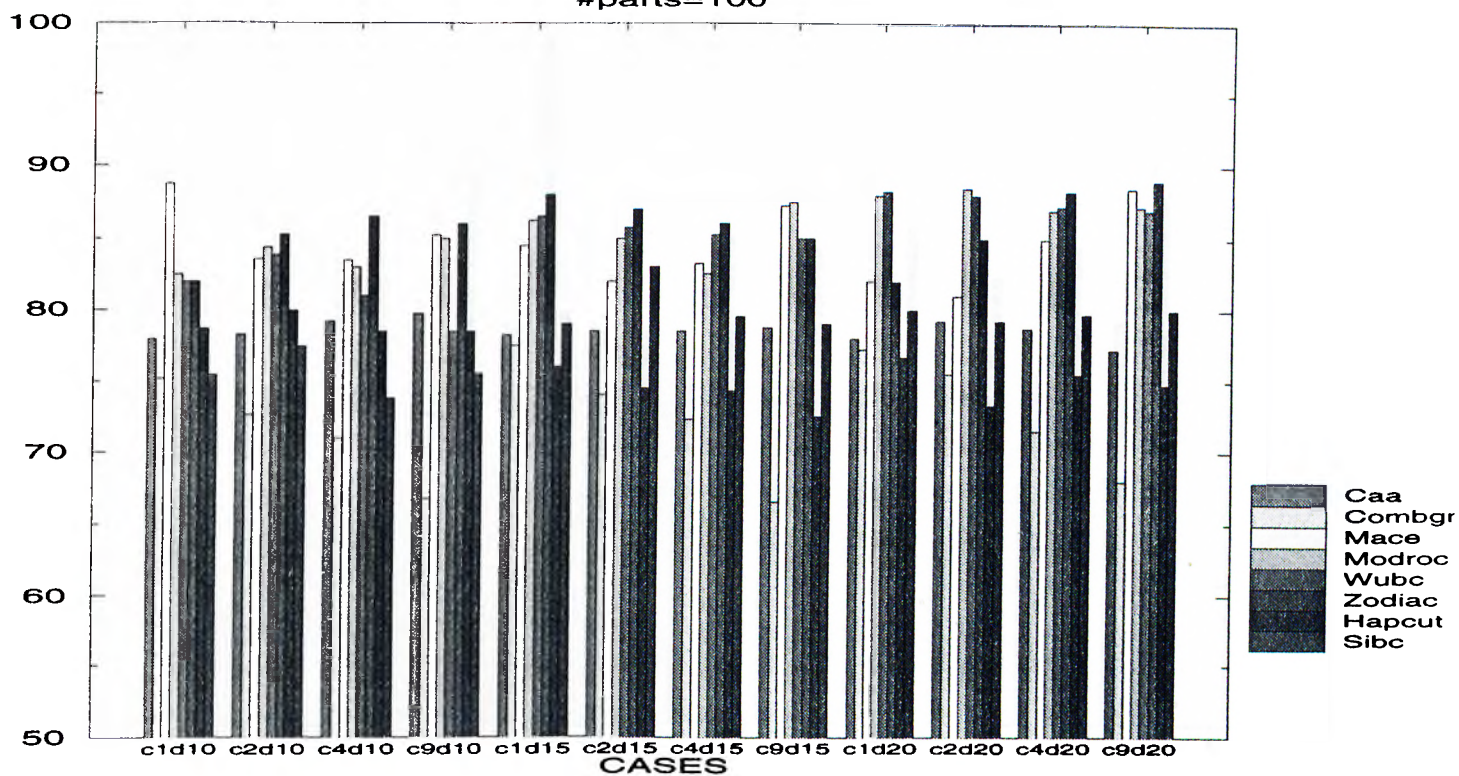


Figure 6.20: Effects clumpiness, density, number of parts and algorithms on grouping efficiencies.

WORKLOAD BALANCE

#parts=100



WORKLOAD BALANCE

#parts=150

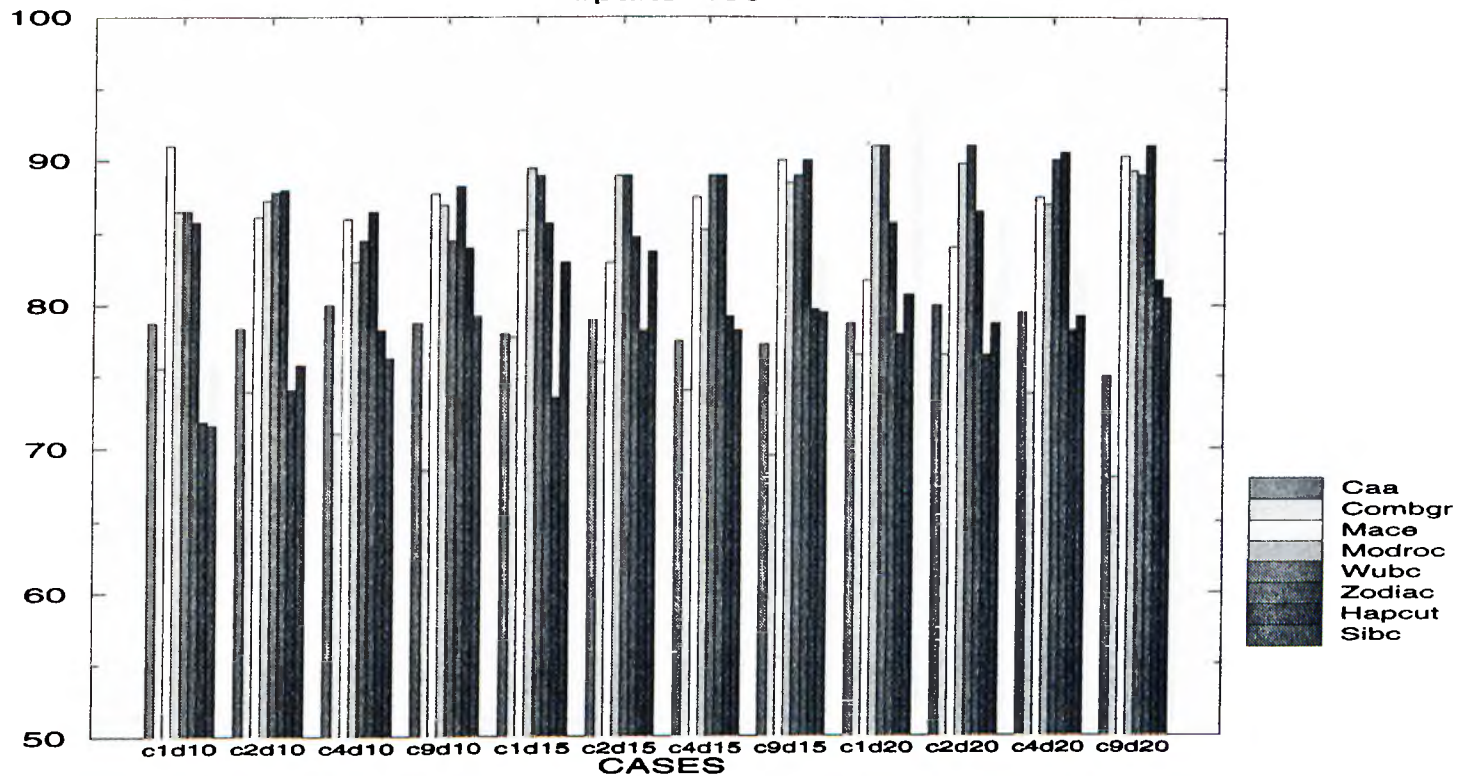
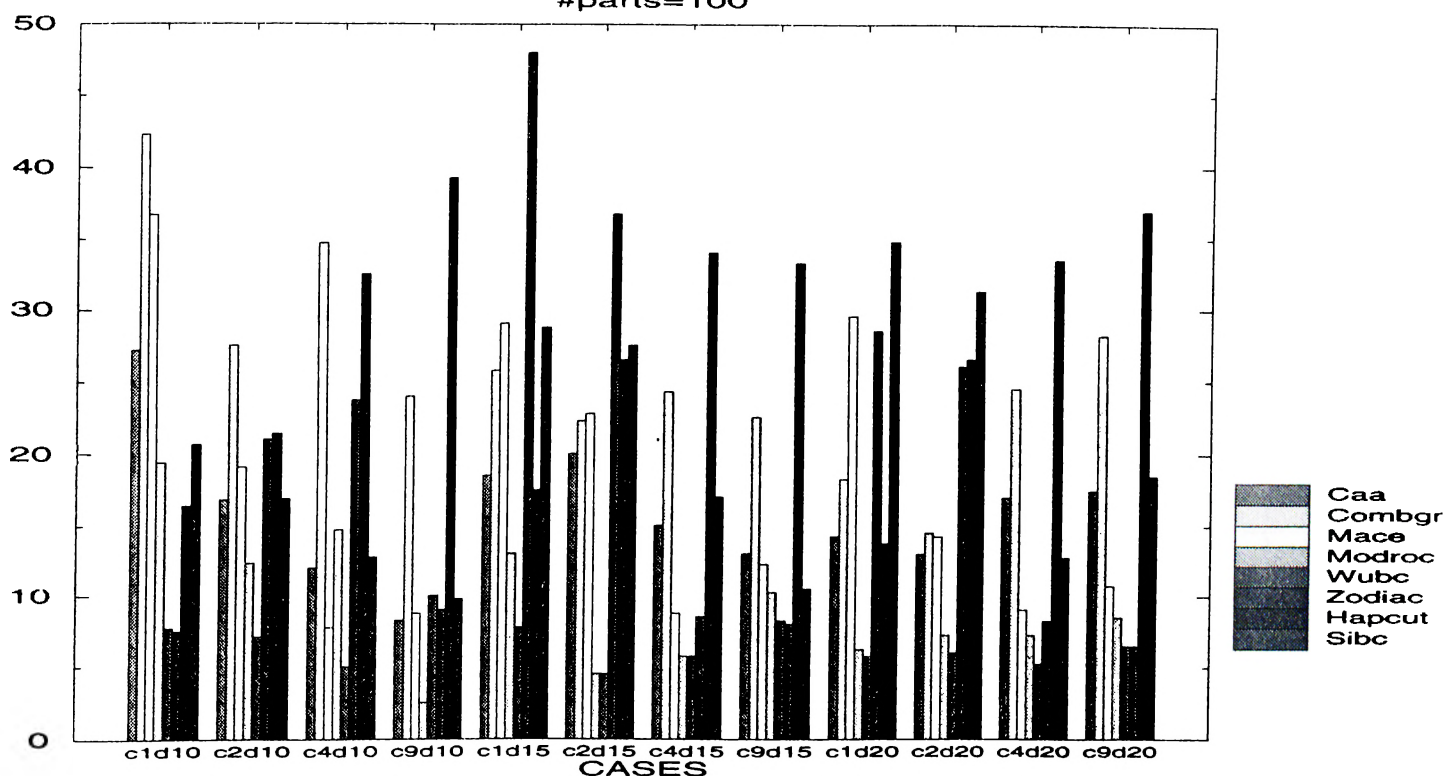


Figure 6.21: Effects clumpiness, density, number of parts and algorithms on work-load balances.

UNDER UTILIZATION

#parts=100



UNDER UTILIZATION

#parts=150

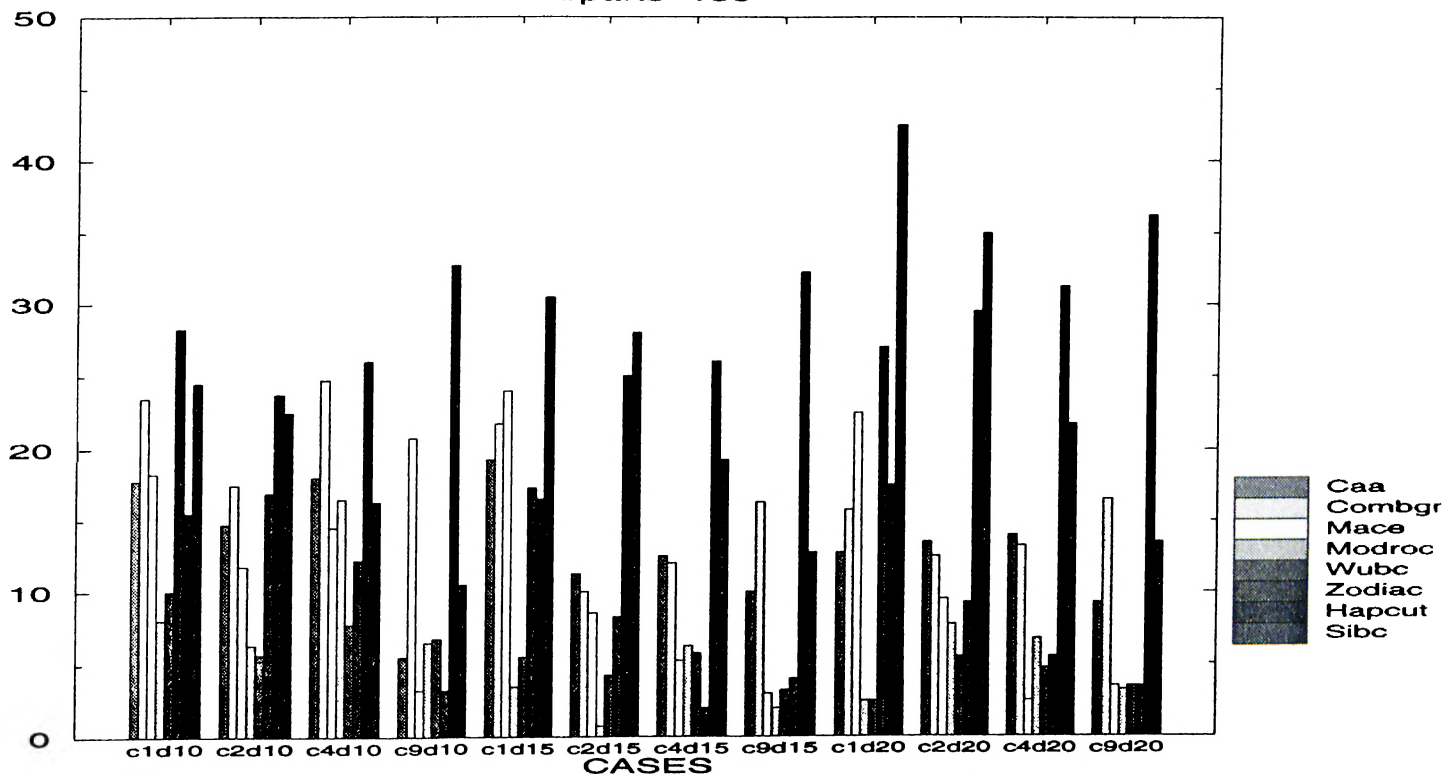


Figure 6.22: Effects clumpiness, density, number of parts and algorithms on under-utilizations.

Number of parts = 100												
CL.	DENS.	GROUPING EFFICIENCY = $f(\alpha)$										
(c)	(d)	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
1	0.10	C	C	C	C	C	C	H,C	H,B,F	B,F,E	B,F	B,F
	0.15	C	C	C	C	C	C,H	H	H,B,E	B,H,E	B,E	B,E
	0.20	C	C	C	C	C	H,G,C	H	H,B,E	B,E,H	B,E	B,E
2	0.10	C,G	C,G	C,G	C,G	G,C	G,C	B,H,G,C	B	B	B	B
	0.15	F	F	F	F,C,H,G	H	H	H	B,H,E	B,E	B,E	B,E
	0.20	F	F	F	G,F,H	H,G	H	H	H,B	B	B	B
4	0.10	C,G	C,G	C,G	C,G	C,G	C	C	B,C	B	B	B
	0.15	F	F	F	F	F	F	F,H	H,F,B	B,H	B	B
	0.20	F,G	F,G	F,G	F,G	F,G	F,G	F,G	B,F,A,H	B	B	B
9	0.10	F	F	F	F,C	F,C	F,C	C,F	C,F	C,F	B	B
	0.15	F	F	F	F	F	F	F	F	F,C,B	B	B
	0.20	F,G	F,G	F,G	F,G,H	F,H,G	H,F,G	H,F,G	H	B,H,E	B	B
Number of parts = 150												
CL.	DENS.	GROUPING EFFICIENCY = $f(\alpha)$										
(c)	(d)	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
1	0.10	C	C	C	C	C	C	C	H	H	B,H,E	B,E
	0.15	C	C	C	C	C	H	H	H	H,B,E	B,E	B,E
	0.20	C	C	C	C	C	H,G	H	H,B,E	B,E,H	B,E	B,E
2	0.10	G,C	G,C	G,C	G,C	G,C	G,C	H,B,G,C	B	B	B	B
	0.15	F	F	F	F	H,F,C,G	H	B,H,E	B,E	B,E	B,E	B,E
	0.20	F	F	F,G	G,F	G,H	H	H,B,E	B,E,H	B,E	B,E	B,E
4	0.10	F	F	F,C	C,F	C,F	C,F	C,F	C,F,B,A	B	B	B
	0.15	G,F	F,G	F,G	F,G	F	F	F,A	F,A,B	B	B	B
	0.20	C,F,G	F,G	F,G	F,G	F	F	F	B,F,A	B	B	B
9	0.10	F	F	F	F,C	F,C	F,C	C,F	C,F	C,F	B,C	B
	0.15	F	F	F	F	F	F	F	F	F,C,B	B	B
	0.20	F	F	F	F	F	F	F	B,F,A	B	B	B

A: CAA B: COMBGR C: MACE D: MODROC
E: WUBC F: ZODIAC G: HAPCUT H: SIBC

Table 6.12: Best cell formation techniques in terms of grouping efficiencies.

CL. (c)	DENS. (d)	# parts = 100		# parts = 150	
		WORK-LOAD BALANCE	UNDER UTILIZATION	WORK-LOAD BALANCE	UNDER UTILIZATION
1	0.10	C	F,E	C	D
	0.15	F,E,D	E	D,E	D,E
	0.20	D,E	D,E	D,E	D,E
2	0.10	F,D,E,C	E	F,E,D,C	E,D
	0.15	F,E,D,H	D,E	D,E,F,H	D
	0.20	D,E	E,D	E,D	E
4	0.10	F	E,C	F,C	E
	0.15	F,E	D,E	F,E,C	F
	0.20	F,E,D	E	F,E	C
9	0.10	F,D,C	D	F,C,D	F,C
	0.15	D,C	F,E	F,C,E,D	D,C,E
	0.20	F,C,D,E	F,E	F,C,D,E	D,E,F,C

A: CAA B: COMBGR C: MACE D: MODROC
E: WUBC F: ZODIAC G: HAPCUT H: SIBC

Table 6.13: Best cell formation techniques in terms of work-load balance and under-utilizations.

Chapter 7

CONCLUSIONS

This chapter provides an account of the contributions of this dissertation research and discusses the directions for future research.

7.1 Contributions

The primary concern of this study was to design methods for obtaining optimal solutions to the hypergraph partitioning problems using polyhedral combinatorics. Hypergraph partitioning arises from practical problems such as circuit partitioning in VLSI design and cell formation in Cellular Manufacturing systems. The summary of the findings in this dissertation work is given below.

Hypergraph partitioning problems were defined and a review of state-of-the-art in partitioning techniques was provided. This reveals that polyhedral techniques are successful in graph partitioning and they remain untouched in the case of hypergraph partitioning.

Two polytopes were defined on r -uniform hypergraphs: the first for solving the problem of identifying the best subhypergraph, and the second for solving hypergraph bipartitioning and max-cut problems. Moreover, any manufacturing system can be represented by hypergraphs.

The Boolean R -atic polytope $RP(H_n^r)$ is the convex hull of the incidence vectors of vertex induced subhypergraphs in a complete r -uniform hypergraph.

Each hypergraph can be transformed into an r -uniform hypergraph in such a way that the end-points of all hyperedges are increased to (r) terminal nodes by adding at most $(r-1)$ pseudo vertices. The vertex induced subhypergraphs are invariant under this transformation if all of the pseudo nodes are kept among the selected vertices in the r -uniform hypergraph. This transformation enables us to operate on r -uniform hypergraphs where each hyperedge connects r vertices. A number of facet defining inequalities were found and two families of valid inequalities were investigated. The problem of identifying the best manufacturing cell is a polynomially solvable instance of the problem defined. The experimental study showed that the inequalities presented throughout the thesis are enough to deal with real applications.

The Complete R -uniform Hypergraph Cut polytope $P_C(H_n^r)$ is the convex hull of the incidence vectors of cuts in a complete r -uniform hypergraph H_n^r . R -uniformity is necessary to have simplicity. Any hypergraph can be theoretically transformed to a complete r -uniform hypergraph by adding pseudo nodes and extra r -uniform hyperedges. However, this transformation increases the order of the original hypergraph. This polytope was proved to be full dimensional. Various families of valid inequalities were proposed and examined for both maximum cut problems and bipartitioning problems. The results indicated that the inequalities are effective especially in the case of solving maximum cut problems in r -uniform hypergraphs.

Any manufacturing system consisting of machines and parts and their routing relationships can be represented as a hypergraph. Based on this abstract structure, a formal definition to the cell formation problem was given as free partitioning. Two cell formation techniques were proposed by means of this formal definition.

The first cell formation method attempts to solve underlying hypergraph partitioning problem by approximating the hypergraph with a graph. Various approximation schemes were investigated. Moreover, a search algorithm operating on the approximated graph was developed. The algorithm makes use of a new kind of similarity coefficient that is based on the values of the minimum cuts

between a pair of machine types in the graph. The algorithm incorporates a well known multiterminal flow algorithm to determine the similarity coefficient values. The algorithm was designed to be as flexible as possible. It is possible to obtain wide range of cell formation alternatives by changing size limits. It is also possible to add any kind of efficiency criterion to direct the algorithm. The algorithm is quite simple and easy to code if a standard maximum flow code is linked. Furthermore, its low time complexity enables it to be used in real life situations.

Another cell formation technique was developed by sequentially solving the problem of identifying the best manufacturing cell. First, the best manufacturing cell is identified. After the machine requirements of the best cell is calculated, the required number of machines of each type is assigned to this cell. Next, the identification of the current best manufacturing cell is done by solving the problem on the remaining machines and parts. These iterations terminate when there is no profitable cell left. All of the remaining machines are grouped together forming the remainder cell. Any unassigned part can either be manufactured in the remainder cell or subcontracted.

The two new cell formation techniques were compared with the six well-known algorithms. Some basic guidelines were provided for the evaluation and the selection of a cell formation technique under different situations. All the techniques were implemented and tested via large problems representing the real life situations. The results were statistically analyzed. The algorithms, number of parts in the system, shop density and characteristics of the manufacturing environment were found to be effective factors in terms of the efficiency measures developed. The best cell formation techniques in terms of each measure were listed according to the mean results of the effective factors. It is hoped that our results will warn the potential user of the weaknesses pertaining to a particular technique. Our comparative analysis indicated that the efficiency indices which are easy to calculate provided that the work-load matrix is available, are quite powerful in evaluating a cell formation solution.

7.2 Future Research

Based on the results on the two polytopes of r -uniform hypergraphs, future research can be conducted on the cut polytope of general hypergraphs. For instance, every facet defining inequality of $P_C(H_n^r)$ also defines a facet of general hypergraph cut polytope $P_C(H)$, if H is any subhypergraph of H_n^r containing the supporting hypergraph of the inequality, or if H is any hypergraph containing H_n^r .

There is a wide research area for polyhedral combinatorics applications on other hypergraph partitioning problems. A hypergraph equicut polytope can be defined to analyze bisection problem in hypergraphs. A hypergraph k -cut polytope can be defined for multiple hypergraph partitionings. Finally, a hypergraph free cut polytope can be defined to study free partitionings in hypergraphs. The findings can be implemented to attack real life problems encountered in VLSI design and manufacturing systems.

Karmarkar's projective algorithm has increased researchers' interest to the interior point methods. Karmarkar [60] combined well known nonlinear optimization techniques such as penalty and barrier methods, projective transformation and devised a polynomial algorithm for linear programming. Interior point methods for integer programming have been initiated by the pioneering work due to Hillier [52] in 1969. In the first stage of Hillier's algorithm, a feasible integer interior point of the polytope of equality constraints is obtained. If one such interior integer point is found (not guaranteed), a line search is conducted by rounding off the points on the line segment going from this interior integer point to the optimal solution of linear programming relaxation. Karmarkar *et al.* developed and experimented a general interior point approach to 0-1 integer programming feasibility problem [57, 58, 61]. A sequence of interior points are generated such that each consecutive point reduces the value of a non-convex potential function. The results obtained were promising. Designing integer interior point based algorithms for hypergraph partitioning problems is another possible research direction.

The most recent tendency in very large-scale linear programming is to combine interior point and simplex methods [12, 50, 75]. Comparative studies in both methods state that interior point methods approach fast to the optimal solution during the first iterations. However, a considerable amount of time is spent to be in the neighborhood of the optimal solution so that the interior point method is stopped. On the other hand, simplex based methods converge slowly to the optimal extreme point in the beginning iterations. Simplex methods hit the optimal solution very fast as soon as they approach to the optimal solution. This lead to the development of hybrid procedures where interior point methods are applied first for few iterations, followed by the application of the simplex method. The difficulty of a hybrid method is the identification of the initial basis of the simplex method. Each application reported a way to handle this difficulty. Developing hybrid procedures for hypergraph partitioning problems are among the possible research directions to extend this dissertation research.

Research on the cell formation problem is far from complete. The new cell formation techniques are needed. It is worth to establish a comprehensive testing basis for new cell formation algorithms, and to pick up the most appropriate cell formation technique to employ in a particular instance. Furthermore, the efficiency measures suggested to evaluate the techniques, and the problem generation require even more exquisite analysis so as to study the cell formation problem from the widest possible perspective implementing a multi-criteria approach, and measure the sensitivity of a technique for different exogenous variables and provide feasible boundaries accordingly. It is quite important to identify the most promising cell formation approaches for future developments.

Appendix A

In the appendix, the conventions used in graph theory, polyhedral combinatorics and linear programming are presented. The details can be found in [81, 84, 87, 99] for linear programming and polyhedral theory, and in [11, 80] for graph and hypergraph theory. We also use in this dissertation well known notions of complexity theory about which an introduction can be found in [39].

A.1 Graph Theory

Let us define a *graph* $G=(V, E)$ of *order* n where $V = \{1, \dots, n\}$ and $ij \in E \iff w_{ij} \neq 0$; the *weight* w_{ij} is associated to the edge ij . If $H=(W, F)$ is a graph with $W \subseteq V$ and $F \subseteq E$, then H is called a *subgraph* of G . $V(F)$ denotes the set of nodes of G that occur at least once as an endnode of an edge in $F \subseteq E$. Similarly, $E(W)$ denotes the set of all edges of G with both endnodes in $W \subseteq V$. If $S, T \subseteq V$ and $S \cap T = \emptyset$ then $(S : T) = \{uv \in E : u \in S, v \in T\}$ denotes the set of edges with one endnode in S and the other in T . A *cut* $\delta(S)$, where $S \subseteq V$ is the set of edges having only one end in S ; i.e., $\delta(S) = \{ij \in E : i \in S, j \in V \setminus S\}$. We write $\delta(v)$, instead of $\delta(\{v\})$, for $v \in V$ and call $\delta(v)$ the *star* of v . The cardinality of the star of a node v is termed as its *degree*. If v is a node of a graph G , then $G \setminus v$ denotes the subgraph of G obtained by removing node v and all edges incident to v from G . Similarly, $G + v$ denotes the graph obtained by adding a new vertex v and edges between all vertices in G and the new vertex v .

A *path* p in G is a sequence of edges e_1, e_2, \dots, e_k such that $e_1 = v_0v_1, e_2 =$

$v_1v_2, \dots, e_k = v_{k-1}v_k$ and such that $v_i \neq v_j$ for $i \neq j$. The nodes v_0 and v_k are the endnodes of p . If $e_{k+1} = v_kv_0 \in E$, then the sequence is $e_1, e_2, \dots, e_k, e_{k+1}$ is called a *cycle*. If p is a cycle, and uv is an edge of $E \setminus p$ with $u, v \in V(p)$, then uv is called a *chord* of p . A cycle with three edges is called a *triangle*. A graph is called a *bicycle k -wheel* if G consists of a cycle of length k and two nodes that are adjacent to each other and to every node in the cycle. An example is illustrated in Figure 3.1.

A graph G is called *complete* if every two different nodes of G are linked by an edge. The complete graph with n nodes is denoted by K_n . A *clique* is a subgraph of a graph that is complete. A clique is not necessarily a maximal complete subgraph. A set A of edges in a graph $G=(V, E)$ is called a *clique partitioning* of G if there is a partition of $V=W_1 \cup W_2 \cup \dots \cup W_p$ upon the removal of A such that the subgraph induced by W_i is a clique for $i=1, \dots, p$. In case G is complete, every partition of the node set of G induces a clique partitioning.

A graph is called *bipartite* if its node set can be partitioned into two nonempty, disjoint sets V_1 and V_2 such that no two nodes in V_1 and no two nodes in V_2 are linked by an edge. If $|V_1|=p, |V_2|=q$ and G is a maximal bipartite graph, it is denoted by $K_{p,q}$.

A graph G is *contractible* to \hat{G} if \hat{G} can be obtained from G by a sequence of elementary contractions, in which a pair of adjacent vertices is identified by one of them and all other adjacencies between vertices are preserved. Multiple edges arising from the identification are replaced by single edges.

A.2 Linear and Integer Programming

A vector $x \in \mathbb{R}^n$ is called a *linear combination* of the vectors $x^1, \dots, x^k \in \mathbb{R}^n$ if for some $\lambda \in \mathbb{R}^k$, $x = \sum_{i=1}^k \lambda_i x^i$. If additionally $\sum_{i=1}^k \lambda_i = 1$, then x is called *affine combination*. Moreover, if $\lambda \geq 0$, then x is *convex combination*. Given a set $S \subseteq \mathbb{R}^n$, $S \neq \emptyset$, we denote by $\text{conv}(S)$ the *convex hull* of S , i.e., the set of all vectors in \mathbb{R}^n that are convex combination of the vectors in S . A set $S \subseteq \mathbb{R}^n$ is called *linearly independent* (respectively, *affinely independent*) if there exists

no subset $\{x^1, \dots, x_k\}$ of S and $\lambda \in \mathbb{R}^k$, $\lambda \neq \mathbf{0}$ (respectively, $\sum_{i=1}^k \lambda_i = 1$) such that $\sum_{i=1}^k \lambda_i x^i = 0$. The *dimension* of a set $S \subseteq \mathbb{R}^n$, denoted by $\dim(S)$, is the cardinality of a maximal affinely independent subset of S minus one.

A set $P \subseteq \mathbb{R}^n$ is called a *polyhedron* if P is the solution set of linear inequalities, i.e., $P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$. A bounded polyhedron is called a *polytope*. If $a \in \mathbb{R}^n \setminus \{0\}$ and $\alpha \in \mathbb{R}$, then the polyhedron $\{x \in \mathbb{R}^n \mid ax \leq \alpha\}$ is called a *halfspace*, and the polyhedron $\{x \in \mathbb{R}^n \mid ax = \alpha\}$ is called a *hyperplane*. If the dimension of $P \subseteq \mathbb{R}^n$ is n we say that the polyhedron is *full dimensional*.

Let $P \subseteq \mathbb{R}^n$ be a polyhedron. An inequality $ax \leq \alpha$ is called *valid* for P if $P \subseteq \{x \in \mathbb{R}^n \mid ax \leq \alpha\}$. We denote by $EQ(P, ax \leq \alpha) \doteq \{x \in P \mid ax = \alpha\}$. A set $F \subseteq P$ is called *face* of P , if there exists an inequality $ax \leq \alpha$ which is valid for P and $F = EQ(P, ax \leq \alpha)$. We say also that F is the *face defined (induced)* by $ax \leq \alpha$. A face F is called *nontrivial* if $F \neq \emptyset$ and $F \neq P$. A *facet* is a maximal nontrivial face. If the face induced by the inequality $ax \leq \alpha$ is a facet, we say that $ax \leq \alpha$ is *facet defining*.

The problem of minimizing (or maximizing) a linear function cx over a polyhedron P is called *Linear Programming problem*, or simply *Linear Program*. Linear Programs are often written in the form

$$\begin{array}{ll} \text{Max } cx & \\ \text{s.t. } Ax \leq b & \end{array} \quad \text{or} \quad \begin{array}{ll} \text{Min } cx & \\ \text{s.t. } Ax \leq b & \end{array}$$

The function cx is called *objective function*. A vector x^* is called the *maximal* (respectively, *minimal*) solution if $Ax^* \leq b$ and $cx \leq cx^*$ (respectively, $cx \geq cx^*$) for all $x \in \mathbb{R}^n$ with $Ax \leq b$. There is an integer version of Linear Programming Problem, defined as

$$\begin{array}{ll} \text{Max } cx & \\ \text{s.t. } Ax \leq b & \\ & x \text{ integral} \end{array} \quad \text{or} \quad \begin{array}{ll} \text{Min } cx & \\ \text{s.t. } Ax \leq b & \\ & x \text{ integral} \end{array}$$

and is called *Linear Integer Programming Problem*. The Linear Program obtained from the problem above by dropping the integrality constraint is called *LP Relaxation*.

Bibliography

- [1] M. R. Anderberg. *Cluster Analysis for Applications*. Monographs and Textbooks on Probability and Mathematical Statistics. Academic Press, Inc., New York, 1973.
- [2] G. Antenucci, S. Nicoloso, and B. Simeone. Optimal hypergraph partitioning in VLSI circuit layout. Technical Report R291, Istituto Di Analisi Dei Sistemi Ed Informatica, 1990.
- [3] R. G. Askin and S. P. Subramanaian. A cost-based heuristic for group technology configuration. *International Journal of Production Research*, 25(1):101–113, 1987.
- [4] A. Ballakur. *An Investigation of Part Family / Machine Group Formation in Designing a Cellular Manufacturing System*. PhD thesis, University of Wisconsin–Madison, 1985.
- [5] A. Ballakur and H. J. Steudel. A within-cell utilization based heuristics for designing cellular manufacturing systems. *International Journal of Production Research*, 25(5):639–665, 1987.
- [6] F. Barahona. The max-cut problem in graphs not contractible to K_5 . *Operations Research Letters*, 2(3):107–111, 1982.
- [7] F. Barahona, M. Grötschel, M. Junger, and G. Reinelt. An application of combinatorial optimization to statistical physics and circuit layout design. *Operations Research*, 36(3):493–513, 1988.

- [8] F. Barahona and A. R. Mahjoub. On the cut polytope. *Mathematical Programming*, 36(5):157–173, 1986.
- [9] E. R. Barnes. An algorithm for partitioning the nodes of a graph. *SIAM Journal on Algebraic and Discrete Methods*, 3(4):541–550, December 1982.
- [10] E. R. Barnes. Partitioning the nodes of a graph. *Manuscript*, 1985.
- [11] C. Berge. *Hypergraphs*. North-Holland, 1989.
- [12] R. E. Bixby, J. W. Gregory, I. J. Lustig, R. E. Masten, and D. F. Shanno. Very large-scale linear programming: A case study in combining interior point and simplex methods. *Operations Research*, 40(5):885–897, 1992.
- [13] R. B. Boppana. Eigenvalues and graph bisection: An average case analysis. In *Proc. of the 28th Annual Symposium on Foundations in Computer Science*, pages 280–285. IEEE, 1987.
- [14] E. Boros, Y. Crama, and P. L. Hammer. Upper bounds for quadratic 0–1 maximization problems. Technical Report RRR 14-89, RUTCOR, New Jersey 08903, May 1989.
- [15] E. Boros and P. L. Hammer. On clustering problems with connected optima in Euclidean spaces. *Discrete Mathematics*, 75:81–88, 1989.
- [16] E. Boros and P. L. Hammer. Cut-polytopes, Boolean Quadratic polytope and nonnegative quadratic pseudo-Boolean functions. Technical Report RRR 24-90, RUTCOR, New Jersey 08903, May 1990.
- [17] E. Boros and P. L. Hammer. The max-cut problem and quadratic 0–1 optimization: Polyhedral aspects, relaxations and bounds. Technical Report RRR 25-91, RUTCOR, New Jersey 08903, May 1991.
- [18] T. N. Bui, S. Chauduri, F. T. Leighton, and M. Sipsur. Graph bisection algorithms with good average case behaviour. *Combinatorica*, 7(2):171–191, 1987.

- [19] J. L. Burbidge. Production flow analysis. *The Production Engineer*, 50(4):139–152, 1971.
- [20] M. P. Chandrasekharan and R. Rajagopalan. An ideal seed non-hierarchical clustering algorithm for cellular manufacturing. *International Journal of Production Research*, 24(2):451–464, 1986.
- [21] M. P. Chandrasekharan and R. Rajagopalan. Modroc: An extension of rank order clustering for group technology. *International Journal of Production Research*, 24(5):1224–1233, 1986.
- [22] M. P. Chandrasekharan and R. Rajagopalan. Zodiac- an algorithm for concurrent formation of part-families and machine-cells. *International Journal of Production Research*, 25(6):835–850, 1987.
- [23] H. R. Charney and D. L. Plato. Efficient partitioning of components. In *Proc. of the 5th Annual Design Automation Workshop*, pages 16.0 – 16.21, 1968.
- [24] S. Chopra and M. R. Rao. Facets of the k-partition polytope. Technical report, June 1989. manuscript, submitted.
- [25] M. Conforti, M. R. Rao, and A. Sassano. The equipartition polytope: part 1. *Mathematical Programming*, 49:49–70, 1990.
- [26] M. Conforti, M. R. Rao, and A. Sassano. The equipartition polytope: part 2. *Mathematical Programming*, 49:71–91, 1990.
- [27] G. Cornuejols, D. Naddef, and W. R. Pulleyblank. The travelling salesman problem in graphs with 3-edge cutsets. *Journal of ACM*, 32:382–410, 1985.
- [28] CPLEX Optimization Inc. *Using the Cplex callable library and Cplex Mixed Integer Library*, 1994.
- [29] M. Deza. On the Hamming geometry of unitary cubes. *Doklady Akademii Nauk SSR*, 134:1037–1040, 1960. English translation in: *Soviet Physics Doklady*, 5, (1961), 940–943.

- [30] M. Deza, M. Grötschel, and M. Laurent. Clique–web facets for multicut polytopes. Technical Report 186, Institut für Mathematik, Universität Augsburg, Universitätsstr. 8 D-8900, Augsburg, September 1989.
- [31] M. Deza and M. Laurent. The cut cone iii: on the role of triangle facets. Technical Report 90655 - OR, Forschungsinstitut für Diskrete Mathematik Institut für ökonometrie und Operations Research, Nassestrasse 2, D-5300, Bonn, July 1990.
- [32] M. Deza and M. Laurent. The cut cone: simplicial faces and variety of realizations. Technical Report 90671 - OR, Forschungsinstitut für Diskrete Mathematik Institut für ökonometrie und Operations Research, Nassestrasse 2, D-5300, Bonn, January 1991.
- [33] M. Deza and M. Laurent. Facets for the cut cone i. *Mathematical Programming*, 56:121–160, 1992.
- [34] M. Deza and M. Laurent. Facets for the cut cone ii. *Mathematical Programming*, 56:161–188, 1992.
- [35] W. E. Donath. Logic partitioning. In B. T. Preas and M. J. Lorenzetti, editors, *Physical design automation of VLSI systems*. The Benjamin/Cummings Pub. Co., Inc, Menlo Park, California, 1988.
- [36] W. E. Donath and A. J. Hoffman. Lower bounds for the partitioning of the graphs. *IBM Journal of Research and Development*, 17(9):420–425, 1973.
- [37] C. M. Fiduccia and R. M. Mattheyses. A linear time heuristic for improving network partitions. In *Proc. 19th Annual Design Automation Workshop*, pages 175–181. ACM & IEEE, 1982.
- [38] J. Garbers, H. J. Promel, and A. Steger. Partitioning graphs into dense subgraphs. Technical Report 89575-OR, Institut für Operations Research, Universität Bonn, Nassestr. 2, 5300 Bonn 1, Germany, July 1989.

- [39] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman and Company, New York, 1979.
- [40] A. M. H. Gerards. Testing the odd bicycle wheel inequalities for the bipartite subgraph polytope. *Mathematics of Operations Research*, 10:359–360, 1985.
- [41] R. E. Gomory and T. C. Hu. Multi-terminal network flows. *J. SIAM*, 9(4):551–570, 1961.
- [42] M. Grötschel and Y. Wakabayashi. A cutting plane algorithm for a clustering problem. *Mathematical Programming*, 45:59–96, 1989.
- [43] M. Grötschel and Y. Wakabayashi. Facets of the clique partitioning polytope. *Mathematical Programming*, 47:367–387, 1990.
- [44] M. Grötschel, F. Barahona, and A. R. Mahjoub. Facets of bipartite subgraph polytope. *Mathematics of Operations Research*, 10:340–358, 1985.
- [45] M. Grötschel and G. R. Nemhauser. A polynomial algorithm for the max-cut problem on graphs without long odd cycles. Technical report, Institut für Ukonometrie und Operations Research, Universtat, Bonn, 1982.
- [46] M. Grötschel and W. R. Pulleyblank. Weakly bipartite graphs and the max-cut problem. *Operations Research Letters*, 1(1):23–27, 1981.
- [47] M. Grötschel and Y. Wakabayashi. Composition of facets of the clique partitioning polytope. In R. Bodendiek and R. Hem, editors, *Topics in Combinatorics and Graph Theory*. Physica-Verlag, Heidelberg, 1990.
- [48] S. Hadley. A new approximation technique for hypergraph partitioning problems. Technical report, University of Waterloo, Department of Electrical and Computer Engineering, Waterloo, Ontario, CANADA N2L 3G1, April 1989.
- [49] F. Hadlock. Finding a maximum cut of a planar graph in polynomial time. *SIAM Journal on Computing*, 4:221–225, 1974.

- [50] M. Hajian, R. Levkovitz, and G. Mitra. A branch and bound algorithm for discrete programming using the interior point method and the simplex method. In *Extended Abstracts of the Symp. on Applied Mathematical Programming and Modelling, January 6–8 Budapest Hungary*, pages 257–265. Hungarian Academy of Sciences, 1993.
- [51] M. Hanan and J. M. Kurtzberg. A review of the placement and quadratic assignment problems. *SIAM Review*, 14:324–342, 1972.
- [52] F. S. Hillier. Efficient heuristic procedures for integer linear programming with an interior. *Operations Research*, 17:600–637, 1969.
- [53] T. C. Hu and Ernest S. Kuh, editors. *VLSI Circuit Layout: Theory and Design*. Number PC01875 in Selected Reprint Series. IEEE Press, New York, 1985.
- [54] T. C. Hu and K. Moerder. Multi-terminal flows in a hypergraph. In T. C. Hu and Ernest S. Kuh, editors, *VLSI Circuit Layout: Theory and Design*, number PC01875 in Selected Reprint Series. IEEE Press, New York, 1985.
- [55] D. S. Johnson, C. R. Aragon, L. A. McGeach, and C. Shevon. Optimization by simulated annealing: An experimental evaluation; part i, graph partitioning. *Operations Research*, 37(6):865–892, November 1989.
- [56] A. B. Kahng. Fast hypergraph partitioning. In *Proc. 26th Annual Design Automation Conference*. ACM & IEEE, 1989.
- [57] A. P. Kamath, N. K. Karmarkar, K. G. Ramakrishnan, and M. G.C. Resende. Computational experience with an interior point algorithm on the satisfiability problem. *Annals of Operations Research*, 25:43–58, 1990.
- [58] A. P. Kamath, N. K. Karmarkar, K. G. Ramakrishnan, and M. G.C. Resende. A continuous approach to inductive inference. *Mathematical Programming*, 57:215–238, 1992.

- [59] L. Kandiller. The part family machine group formation problem in cellular manufacturing. Master's thesis, Bilkent University, 1989.
- [60] N. K. Karmarkar. A new polynomial algorithm for linear programming. *Combinatorica*, 4:373–395, 1984.
- [61] N. K. Karmarkar, M. G.C. Resende, and K. G. Ramakrishnan. An interior point algorithm to solve computationally difficult set covering problems. *Mathematical Programming*, 52:597–618, 1991.
- [62] L. Kaufman and P. J. Rousseeuv. *Finding Groups in Data: An Introduction to Cluster Analysis*. Probability and Mathematical Statistics. John Wiley and Sons, Inc., New York, 1989.
- [63] B. W. Kernighan and S. Lin. An efficient heuristic procedure for partitioning graphs. *Bell Syst. Tech. J.*, 49(2):291–307, February 1970.
- [64] J. R. King. Machine-component grouping in production flow analysis: An approach using a rank order clustering algorithm. *International Journal of Production Research*, 18(2):213–232, 1980.
- [65] J. R. King. Machine-component group formation in group technology: Review and extension. *International Journal of Production Research*, 20(2):117–133, 1982.
- [66] S. Kirkpatrick, C. D. Gelatt Jr., and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, May 1983.
- [67] B. Krishnamurty. An improved min-cut algorithm for partitioning vlsi networks. *IEEE Transaction on Computers*, C-33(5):438–446, 1984.
- [68] E. S. Kuh and T. Ohtsuki. Recent advances in VLSI layout. *Proceedings of the IEEE*, 78(2):237–263, February 1990.
- [69] K. R. Kumar, A. Kusiak, and A. Vannelli. Grouping of parts and components in flexible manufacturing systems. *European Journal of Operational Research*, 24:387–397, 1986.

- [70] A. Kusiak and W. S. Chow. Efficient solving of the group technology problem. *Journal of Manufacturing Systems*, 6(2):117–124, 1987.
- [71] A. Kusiak and W. M. Ibrahim. Kbg: A knowledge-based system for grouping machines and parts. Technical Report Working Paper # 03/88, Department of Mechanical and Industrial Engineering, University of Manitoba, Winnipeg–Manitoba, CANADA, 1988.
- [72] R. K. Kwatera and B. Simeone. Clustering heuristics for set covering. Technical Report n10, Dipartimento Di Statistica, Probabilità e Statistica Applicata, Università Degli Studi Di Roma 'La Sapienza', 1990.
- [73] E. L. Lawler, J. K. Lenstra, A. G. H. Rinnoy Kan, and D. B. Shmoys. *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*. John Wiley, New York, 1985.
- [74] T. Lengauer. *Combinatorial Algorithms for Integrated Circuit Layout*. Applicable Theory in Computer Science Series. John Wiley & Sons, University of Paderbon, Germany, 1990.
- [75] R. Levkovitz, G. Mitra, and M. Tamiz. Experimental investigations in combining IPM and simplex based LP solvers. In *Extended Abstracts of the Symp. on Applied Mathematical Programming and Modelling, January 6–8 Budapest Hungary*, pages 353–373. Hungarian Academy of Sciences, 1993.
- [76] R. J. Lipton and R. E. Tarjan. A separator theorem for planar graphs. *SIAM Journal on Applied Mathematics*, 36(2):177–189, 1979.
- [77] L. Lovász, M. Grötschel, and A. Schrijver. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica*, 1:169–197, 1981.
- [78] W. A. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller. Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, 21:1087–1092, 1953.

- [79] J. Miltenburg and W. Zhang. A comparative evaluation of nine well-known algorithms for solving the cell formation problem in group technology. *Journal of Operations Management*, 10, 1992.
- [80] M. Minoux and M. Goldran. *Graphs and Algorithms*. John Wiley & Sons, 1984.
- [81] G. L. Nemhauser and L. A. Wolsey. *Integer and Combinatorial Optimization*. John Wiley & Sons, 1988.
- [82] E. Olivia-Lopez and G. F. Purcheck. Load balancing for group technology planning and control. *International Journal of MTDR*, 19:259–274, 1979.
- [83] M. Padberg. The boolean quadratic polytope: some characteristics, facets and relatives. *Mathematical Programming*, 45:139–172, 1989.
- [84] W. Pulleyblank. *Polyhedral Combinatorics*, volume I of *Handbooks of Operations Research and Management Science*, chapter Optimization. North-Holland, 1989.
- [85] G. F. Purcheck. Combinatorial grouping – a lattice-theoretic method for the design of manufacturing systems. *Journal of Cybernetics*, 4(3):27–60, 1974.
- [86] G. F. Purcheck. Machine–component group formation: An heuristic method for flexible production cells and flexible manufacturing systems. *International Journal of Production Research*, 23(5):911–943, 1985.
- [87] A. Schrijver. *Theory of linear and integer programming*. John Wiley & Sons, 1986.
- [88] D. G. Schweikert and B. W. Kernighan. A proper method for the partitioning of electrical circuits. In *Proc. 9th Annual Design Automation Workshop*, pages 57–62. ACM & IEEE, 1972.
- [89] C. De Simone. The cut polytope and the boolean quadratic polytope. *Discrete Mathematics*, 79:71–75, 1989.

- [90] C. De Simone, M. Deza, and M. Laurent. Collapsing and lifting for the cut cone. Technical Report 265, Istituto Di Anallisi Dei Sistemi Ed Informatica, IASI-CNR Consqlio Nazionale belle Ricerche, Roma, 1989.
- [91] J. D. Ullman. *Computational Aspects of VLSI*. Principles of Computer Science. Computer Science Press, Stanford University, 1984.
- [92] A. J. Vakharia and U. Wemmerlöv. Designing a cellular manufacturing system: A materials flow approach based on operation sequences. *IIE Transactions*, 6:84, 1990.
- [93] A. Vanelli and S. W. Hadley. A Gomory-Hu cut tree representation of a netlist partitioning problem. *IEEE Transactions on Circuits and Systems*, 37(9):1133–1139, 1990.
- [94] P. H. Waghodekar and S. Sahu. Machine–component cell formation in group technology: Mace. *International Journal of Production Research*, 22(6):937–948, 1984.
- [95] K. Wagner. Beweis einer abschwachung der hardwiger-vernutung. *Math. Ann.*, 153:139–141, 1964.
- [96] U. Wemmerlöv and N. L. Hyer. Procedures for the part family / machine group identification problem in cellular manufacturing. *Journal of Operations Management*, 6(2):125–147, 1986.
- [97] U. Wemmerlöv and N. L. Hyer. Research issues in cellular manufacturing. *International Journal of Production Research*, 25(3):413–435, 1986.
- [98] M. Yannakakis. Node - and edge - deletion NP-complete problems. In *Proc. of the 10th Annual Symp. on Theory of Computing*, pages 253–264. ACM, 1978.
- [99] G. M. Ziegler. Lectures on polytopes. Konrad-Zuse-Zentrum für Informationstechnik Berlin (ZIB), August 1993.

Vita

Levent Kandiller was born in Ankara, Turkey, in 1964. He received his B.S. degree as a top graduate from Department of Industrial Engineering, Middle East Technical University, Ankara, Turkey in 1986. In September 1986, he joined to Department of Industrial Engineering, Bilkent University as a research assistant. He got his M.S. degree in April 1989 with the thesis entitled as "*Part Family Machine Group Formation Problem In Cellular Manufacturing Systems*". During his Ph.D. study with Dr. Mustafa Akgül, he worked on combinatorial optimization. Currently, he is an instructor at Department of Industrial Engineering, Bilkent University.