A NEW BRANCH AND BOUND ALGORITHM FOR LOADING PROBLEMS IN FLEXIBLE MANUFACTURING SYSTEMS

Yeong-Dae Kim
Department of Industrial Engineering
Korea Advanced Institute of Science and Technology
Yusong-gu, Daejon 305-701
Korea

Candace Arai Yano
Department of Industrial and Operations Engineering
The University of Michigan
Ann Arbor, Michigan 48109-2117
USA

February 1989

Revised July 1990, August 1991, and May 1992

A New Branch and Bound Algorithm for Loading Problems in Flexible Manufacturing Systems

Abstract

Loading problems in flexible manufacturing systems involve assigning operations for selected part types and their associated tools to machines or machine groups. One of the objectives might be to maximize the expected production rate (throughput) of the system. Because of the difficulty in dealing with this objective directly, a commonly-used surrogate objective is the "closeness" of the actual workload allocation to the continuous workload allocation that maximizes throughput. We test several measures of closeness and discuss correlations between these measures and throughput. Using the best measure, we show how to modify an existing branch and bound algorithm which was developed for the case of equal target workloads for all machine groups to accommodate unequal target workloads. We also develop a new branch and bound algorithm which can be used for both types of problems. The efficiency of the algorithm in finding optimal solutions is achieved through the application of better branching rules and improved dominance results. Computational results on randomly generated test problems indicate that the new algorithm performs well.

1. Introduction

Closed queueing network (CQN) models are becoming a popular means of estimating the production capacity of flexible manufacturing systems (FMSs) and flexible assembly systems (FASs). Representing a FMS or FAS as a closed queueing network permits a modeler to capture certain aspects of the routing flexibility that exists in such systems without sacrificing the ability to estimate the production capacity quickly and easily.

A CQN model requires as input a routing matrix and the workload (total processing time demand per unit time) assigned to each machine group. A machine group is composed of one or more machines that can perform the same operations at the same speed. The routing matrix contains the probability that a job will move from one machine group to each of the other machine groups, for all possible pairs of machine groups. The routing matrix is easy to determine once it has been decided what operations will be performed by each machine group, and it is these operation assignments that determine the workload allocation.

A considerable amount of research has been done on the problem of allocating workload among machine groups when the total workload can be divided in a continuous fashion and without regard for the characteristics and capabilities of the machines (e.g., Stecke and Morin 1985, Yao 1985, Shanthikumar and Stecke 1986, and Yao and Kim 1987a,b). In real systems, operation assignments result in *discrete* allocations of workloads to machine groups, and the number of operations that can be assigned to each machine group is limited by the number of tools or components that can be loaded onto the machine at one time. A number of FMSs now use automatic tool changers, which significantly reduce the impact of tool magazine capacity constraints. However, recent dicussions with several Fortune 50 companies indicate that some FMSs and most FASs still face the difficulty of solving scheduling-related problems in view of tool or component capacity constraints.

We develop a new procedure for assigning a given set of operations to machine groups so as to maximize the production capacity (or throughput) of the system while ensuring that tool or component limitations are satisfied. Most of the earlier work has focused on the objective of balancing workloads under the assumption of identical machines and equal machine groups. We consider a generalization of the problem in which target workloads may differ across machine groups, which allows us to address problems in which there are machine groups of different sizes. Papers that consider other issues in this context include Dallery and Stecke (1990), Shanthikumar and Yao (1987, 1988), and Stecke and Solberg (1981). Even for the restricted version of the problem, existing optimization-based procedures are computationally-intensive. We develop rules to eliminate dominated solutions so that larger problems can be solved optimally.

The new procedure can be used in several different ways. First, it can be used as a capacity planning tool to provide more accurate estimates of production capacity, taking into account practical constraints. Since a typical FMS can manufacture a wide variety of parts, the mix of parts is likely to change over both the short and long run. Consequently, in order to obtain robust estimates, one actually needs to evaluate the capacity of a system under many different assumptions about the mix (combinations and relative proportions) of parts that might be processed simultaneously.

Secondly, it can be used to aid in the decision of which parts should be produced on the FMS, and if there is more than one FMS, how to partition parts among them. (The latter is in the spirit of group technology concepts.) The production capacity estimates derived from the approach can provide information about the compatibility (or incompatibility) of parts with regard to utilization of equipment and tools.

Finally, in some situations, the procedure can be used to solve the short-term problem of allocating operations and tools to machine groups for the imminent production run. This is known in the literature as the *FMS loading problem*. Since the CQN models that form the basis for the analysis are steady-state models, they are applicable only when a constant mix of parts is produced for a long enough duration that the system is in steady state for a substantial portion of that time. There are, however, systems with common due dates for orders (e.g., once a week shipments) and relatively short processing times (e.g., minutes), where steady-state approximations are realistic.

Most of the existing literature on this topic has focused on the short-term loading aspect of this problem. Stecke (1983) formulates it as a nonlinear mixed integer program and solves it through linearization techniques. A branch and bound algorithm is developed by Berrada and Stecke (1986) for this formulation with the objective of balancing the workloads. Also several heuristics are suggested by Stecke and Talbot (1985) for various objectives. Following the approach of Stecke (1983), Ammons *et al.* (1985) and Shanker and Tzen (1985) give formulations with bi-criterion objectives and present heuristic procedures. Ammons *et al.* use the objective of balancing workloads and minimizing workstation visits, while Shanker and Tzen use the objective of balancing workloads among machining centers and meeting due dates. Lashkari *et al.* (1987) also formulate the problem as a nonlinear mixed integer program for two different objectives, minimizing transport load and minimizing refixturing activities. Kusiak (1985) presents several models based on the generalized assignment problem and on the generalized transportation problem, considering the cost of processing an operation at a station.

Another goal of the loading problem concerns the life of cutting tools. Carrie and Perera (1986) show, using a simulation of a particular FMS, that the number of tool changes due to product variety is small compared to those due to tool wear. Sarin and Chen (1987) give a mixed integer program formulation for the objective of minimizing total machining costs associated with cutting tools and machine usage under the assumption that these costs depend upon the tool-machine combination. Tool life is considered as a constraint in the formulation.

The loading problem is combined with other problems in other research. Greene and Sadowski (1986) use several objective functions such as minimizing makespan, minimizing mean flow time, and minimizing mean lateness in their formulations of the FMS loading and scheduling problems. Since the numbers of variables and constraints are very large, computational experiments were not performed. Rajagopalan (1986) formulates the loading problem together with the part type selection and the production ratio problems as a mixed integer linear program. The part type selection problem is to choose a subset of part types for immediate and simultaneous production, and the production ratio problem is to determine the number of parts of each type circulating in the system. (See Stecke 1983 and Kiran and Tansel 1986 for details.) He gives heuristic algorithms and related computational results. Stecke (1986) presents a hierarchical approach for the loading problem and the machine grouping problem (partitioning machine into groups, where the groups differ only in their tooling).

In the next section, we evaluate surrogate objectives and show how an existing branch and bound algorithm for the workload balancing objective can be used to solve the loading problem with unequal workload targets. In the following section, we develop dominance properties for use within a new branch and bound algorithm. Section 4 gives computational results for the existing and new branch and bound approaches. Section 5 contains a summary and concluding remarks.

2. Objectives and Formulations

Stecke and Solberg (1985) have shown that there is a unique *continuous* workload allocation that maximizes the production capacity (or throughput) of a system, also referred to as *ideal workloads*. When two or more machine groups are identical both in size and machine capabilities, it is optimal to divide the relevant work among them equally, which they call *balancing*. On the other hand, when there are unequal size groups of otherwise identical machines, it is better to assign larger workloads per machine to larger groups to take advantage of the effects of pooling. They refer to the process of doing this optimally as *unbalancing*.

There is no universally accepted measure of workload (un)balance for loading problems where the operations assignments are *discrete*. Since the true optimization problem is a non-convex integer programming problem (because the throughput function is not convex), researchers have found it more convenient to use surrogate objectives. For example, Ammons *et al.* (1985), Shanker and Tzen (1985), and Berrada and Stecke (1986) use different surrogate objective functions for the same stated objective of balancing workloads. Ammons *et al.* try to minimize a function (denoted as $f(\cdot)$ in their paper) of the maximum deviation from the ideal workloads, and Shanker and Tzen have the objective of minimizing the weighted sum of overloads and underloads of the machines, while Berrada and Stecke use the objective of minimizing the maximum workload among the machine groups. While it is clear that the objective should be that of achieving a workload allocation "as close as possible" to the ideal workloads, "as close as possible" can be defined in many different ways, and we are not aware of any systematic studies of surrogate objectives.

In the Appendix, we investigate twelve objective functions as measures of (un)balance. It is useful to point out that because of the combinatorial nature of the problem, most objective functions lead to problems with similar computation requirements. Hence, the reason for studying surrogate objectives is not to reduce computing time, *per se*, but to find a simple objective that gives consistently good results across a wide variety of system parameters. We found that $\max_l \; (X_l - IX_l \;) \; / \; IX_l$, where X_l and IX_l denote the actual load and ideal load for machine group l (not per machine), respectively, outperformed the other objective functions.

Given this objective, an existing branch and bound (B&B) algorithm proposed by Berrada and Stecke (1986) for the case of balancing workloads can be modified to accommodate the unbalancing objective for situations in which the processing time of an operation is the same for all machine groups. Note that if the processing times differ across machine groups, the total workload, and consequently also the IX_ls, cannot be determined independently of the operation assignments. Let δ be an upper bound on $\max_l \; (X_l - IX_l) \,/\, IX_l$. Then the problem is to find $\{ x_{il} \mid i=1,\cdots,b, \; l=1,\cdots,M \}$ to:

Problem (UB)

Minimize δ

subject to

$$(\sum_{i=1}^{b} a_i \, p_l \, x_{il} - IX_l) \,/\, IX_l \le \delta , \quad l=1,\cdots,M \tag{1}$$

$$\sum_{i=1}^{b} d_i \, x_{il} + \sum_{p=2}^{b} (-1)^{(p+1)} \sum_{|B|=p} w_B \, \Pi_{i \in B} \, x_{il} \le t_l , \quad l=1,\cdots, M \tag{2}$$

$$\sum_{i=1}^{M} x_{il} = 1 , \quad i=1,\cdots,b \tag{3}$$

$$x_{il} = 0 \text{ or } 1 , \quad \text{for all } i \text{ and } l, \tag{4}$$

where

I	set of operations, $I = \{ i \mid i =1,\cdots,b \}$
L	set of machine groups, $L = \{ l \mid l =1,2,\cdots, M \}$
B	a subset of I
p_i	processing time of operation i
d_i	number of slots required in a tool magazine by operation i
t_l	capacity of the tool magazine of the machines in machine group l
w_B	number of slots saved when the operations in subset B I are assigned to the same machine group
a_i	relative production ratio at which operation i will be produced
IX_l	(relative) ideal workload of machine group l

x_{il} decision variable (1 if operation i is assigned to machine group l, 0 otherwise).

This problem (UB) is identical to the problem in Berrada and Stecke except for constraint (1), which can be rewritten as

$$\sum_{i=1}^{b} a_i \frac{p_i}{IX_l} x_{il} - 1 \leq \delta, \quad l=1,\cdots,M$$

If we let $q_{il} = \frac{p_i}{IX_l}$ and $\gamma = \delta + 1$, then the constraint is identical to

$$\sum_{i=1}^{b} a_i \, q_{il} \, x_{il} \leq \gamma, \quad l=1,\cdots,M \tag{5}$$

With this substitution, the problem becomes:

Problem (BUB)

Minimize δ

subject to (2), (3), (4), and (5).

It is important to note that the Berrada-Stecke algorithm is an ε-optimal algorithm, and one needs to be very careful in selecting ε if an optimal solution is desired. Refer to Berrada and Stecke (1986) for additional details. To overcome this drawback of the existing algorithm, we propose a new (exactly optimal) branch and bound algorithm in the next section. We use the above formulations in this new algorithm.

3. A New Branch and Bound Algorithm

There are several ways to build a branch and bound tree for a loading problem. For example, each level of the B&B tree can correspond to an operation, or alternatively to a machine group. See Figure 1 for an illustration of a case with four machine groups. In our approach, we initially branch on the number of operations assigned to each group, which we call a *number grouping*, because this permits us to fathom nodes more easily in the earlier stages of the algorithm.

>> Insert Figure 1 here <<

In the following, properties for fathoming or pruning nodes in the B&B tree are presented for the unbalancing objective, along with methods to avoid generating redundant nodes. (We also explain briefly how these results should be modified for the balancing objective.) We use these properties in the new branch and bound procedure for the objective of unbalancing workloads. The objective is to minimize the maximum (over all machine groups) of the ratios of the actual workload to the ideal workload. As discussed earlier, we first generate all possible *number groupings*, which specify how many operations are to be assigned to each machine group.

The machine groups can be partitioned into several sets, where each set contains groups with the same number of machines. For example, the first set will contain all machine groups with one machine, and the second set will contain all machine groups with two machines, and so forth. (Some sets may be empty.) A machine group in one set has a different number of machines than a machine group in another set, and therefore the two groups have different processing time capacities. Let k_{ij} be the number of operations to be assigned to the j-th machine group in the i-th set, and c_i be the number of machine groups in the i-th set. All possible number groupings can be generated for a loading problem with b operations and M machine groups by solving *NGUB*. Note that $\sum_{i=1}^{l} c_i$ is equal to M, where l is the number of sets.

(*NGUB*) Find all number groupings $(k_{11}, \cdots, k_{1c_1}, k_{21}, \cdots, k_{2c_2}, \cdots, k_{l1}, \cdots, k_{lc_l})$

such that $k_{i1} \le k_{i2} \le \cdots \le k_{ic_i}$, for $i=1,2,\cdots,l$,

$$\sum_{i=1}^{l} \sum_{j=1}^{c_i} k_{ij} = b,$$

k_{ij} is a positive integer, for all i and j.

For example, consider a problem with 7 operations and 3 machine groups. If the machine groups are of equal size, there are four number groupings for the operations: (1,1,5), (1,2,4), (1,3,3), and (2,2,3), where the numbers in parentheses denote the numbers of operations assigned to the respective machine groups. There are nine number groupings, namely, (1,1,5), (1,2,4), (1,3,3), (1,4,2), (1,5,1), (2,2,3), (2,3,2), (2,4,1), and (3,3,1), if two machine groups, say groups 1 and 2, are of equal size. If all three machine groups differ in size, there are 15 number groupings, i.e., (1,1,5), (1,2,4), (1,3,3), (1,4,2), (1,5,1), (2,1,4), (2,2,3), (2,3,2), (2,4,1), (3,1,3), (3,2,2), (3,3,1), (4,1,2), (4,2,1), and (5,1,1).

3.1. Eliminating dominated number groupings

We derive properties that allow us to eliminate number groupings in which a machine group has too many or too few operations assigned to it. Let p_i be the processing time for operation i, k_j be the number of operations to be assigned to the j-th group in the current number grouping, and (i) be the index of the operation with i-th longest processing time, i.e., $p_{(1)} \geq p_{(2)} \geq \cdots \geq p_{(b)}$. In addition, let IX_j be the ideal workload of machine group j, and R_{max} be an upper bound (e.g., from an incumbent solution) of the maximum ratio of the actual workload to the ideal workload among the machine groups. Note that feasible assignments from a number grouping $(k_1, k_2, \cdots, k_M)$ must have k_1 operations assigned to the first group, k_2 operations assigned to the second group, etc.

One should not confuse the expressions for number groupings in the following propositions and with those in *NGUB*. For simplicity, the group identification indices, which were expressed as double subscripts above, are reindexed 1 through M in the following propositions.

Proposition 1. *The current incumbent solution dominates any solution from a number grouping* $(k_1, k_2, \cdots, k_M)$ *for which*

$$\sum_{i=1}^{k_j} p_{(b+1-i)} \;/\; IX_j \;\geq\; R_{max}, \qquad j=1,2,\cdots, \text{or } M. \tag{6}$$

Proof. For any j, k_j is the number of operations to be assigned to the jth machine group. Therefore, $\sum_{i=1}^{k_j} p_{(b+1-i)}$ is a lower bound on the workload of the jth machine group.

Hence, the left hand side of (6) is a lower bound on the ratio of the actual to the ideal workload of the jth group. Therefore, the number grouping $(k_1, k_2, \cdots, k_M)$ is dominated by the current incumbent solution. ■

Proposition 2. *There exists at least one feasible solution which is as good as any solution from a number grouping* $(k_1, k_2, \cdots, k_M)$ *for which*

$$\sum_{i=1}^{k_1} p_{(i)} + p_{(b)} \leq IX_j, \qquad j=1,2,\cdots, \text{or } M \tag{7}$$

and the tool slots needed for any k_1+1 *operations do not exceed the tool magazine capacity for the machines.*

Proof. $\sum_{i=1}^{k_1} p_{(i)}$ is an upper bound on the workload of the first machine group, which has k_1 operations assigned to it. The addition of operation (b) does not violate the tool magazine capacity constraint and does not increase the maximum ratio. Therefore, the number grouping $(k_1, k_2, \cdots, k_M)$ is dominated by a number grouping in which the first group has k_1+1 operations assigned to it. ■

The above properties permit fathoming of dominated number groupings. Proposition 1 eliminates number groupings in which a machine group or groups have too many operations assigned to them, while Proposition 2 eliminates those in which a machine group has too few operations.

From the above propositions, upper and lower bounds on the number of operations in each group can be obtained. Furthermore, these bounds can be improved by noting the relationships among these bounds and the total number of operations, b. For example, if the lower bound for group i is less than b minus the sum of the upper bounds of the other groups, the lower bound for group i can be reset to the latter value. Similarly, if the upper bound of group i is greater than the value obtained by subtracting the sum of the lower bounds of the other groups from b, an improved upper bound can be obtained for group i.

3.2. Branching

For each of the remaining number groupings, a B&B sub-tree must be constructed. In the B&B tree, nodes specify operations and the level of each node corresponds to a specific machine group. For example, if the number grouping considered is $(k_1^o, k_2^o, \cdots, k_M^o)$, the nodes at levels $1, 2, \cdots, k_1^o$ specify the operations to be assigned to machine group 1, the nodes at levels $k_1^o+1, k_1^o+2, \cdots, k_1^o+k_2^o$ specify the operations for machine group 2, and so on.

In the assignment problem, the order of inclusion of operations in a machine group does not make any difference. In other words, many different nodes in a B&B tree give exactly the same assignment. To prevent generating redundant nodes which result in the same assignment, we use two rules to generate successor nodes in our B&B algorithm.

Rule 1. The index of the i-th operation of a group should be greater than the index of the $(i-1)$-th operation of the group.

Rule 2. When there are two or more groups with the same processing time capacity and the same number of assigned operations, the index of the first operation of the i-th group should be greater than that of the $(i-1)$-th group.

Feasibility of the current operation assignments with respect to the tool magazine capacity constraints is checked at each branching step. The impact of using tools or components common to two or more operations assigned to the same machine group is considered in the feasibility checking routine.

3.3. Fathoming

Properties similar to those in Propositions 1 and 2 can be used to fathom operation assignments as well. Let I_j be the set of indices of operations assigned to the jth machine group, and b_j be the cardinality of the set I_j. Each set I_j can be represented by a node in the B&B tree. Recall that k_j operations are to be assigned to group j. The cardinality of I_j may, however, be strictly less than k_j except at terminal nodes in the tree. The following two propositions can be used to fathom nodes corresponding to dominated operation assignments.

Proposition 3. *The current incumbent solution dominates any solution obtained from a node for which*

$$\sum_{i \in I_a + I_j} p_i \geq R_{max} \cdot IX_j \tag{8}$$

where I_a is the set of indices of the $k_j - b_j$ shortest operations not in I_j, and the level of the node being considered is associated with the j-th group.

Proof. Since the left hand side of (12) is a lower bound on the workload of the group, the above result is obvious. ■

Proposition 4. *There exists at least one feasible assignment which is as good as any solution from a node for which*

$$\sum_{i \in I_c + I_j} p_i + p_d \leq IX_j \tag{9}$$

where I_c is the set of indices of the $k_j - b_j$ longest operations not in I_j, d is the index of the shortest operation not yet assigned to a machine group, and the operations in I_j and any other $k_j - b_j + 1$ operations do not violate the tool magazine capacity constraint for the machine group.

Proof. When branching from the node under consideration, there are $k_j - b_j$ more operations to be assigned to group j. The first term on the left hand side of (9) is an upper bound on the workload for machine group j. The addition of operation d does not violate the tool magazine capacity constraint and does not increase the maximum ratio. Therefore, having $k_j - b_j + 1$ more operations gives a better feasible solution, which cannot be reached from the node under consideration. ■

3.4. Branch and bound algorithm

In the proposed algorithm, all possible number groupings are generated by *NGUB* and some of them are eliminated by applying Propositions 1 and 2. A feasible solution is needed to eliminate dominated number groupings, which in turn reduces computation time significantly. If a feasible solution is not available, we can select a number grouping arbitrarily and build a B&B sub-tree to find a feasible solution. However, the solution found from the arbitrarily selected number grouping may not be good enough to eliminate many number groupings. The heuristic algorithms in Kim and Yano (1987), which incorporate tool magazine capacity constraints, are used to find an initial solution. We have observed that these heuristics provide excellent initial solutions which are generally better than the solutions obtained from the ε-optimal procedure of Berrada and Stecke (1986). The value of R_{max} obtained from the heuristic is used in applying Proposition 1.

The number groupings left after the elimination are heuristically ranked for consideration as follows. In the unbalancing case, number groupings are ranked in increasing order of the maximum number of operations *per machine* across all machine groups. In the balancing case, number groupings are considered in decreasing order of the variance of the number of operations per machine group (also per machine, since the group sizes are equal). The rationale for these rankings is to give preference to number groupings which are "mostly likely to" avoid over- or underloading the machine groups from the standpoint of capacity utilization. This allows us to generate improved solutions early in the search, which in turn, allows us to prune other portions of the tree quickly.

For each number grouping in turn, the branching scheme is used to construct a B&B sub-tree using a depth-first search. Nodes are fathomed by applying Propositions 3 and 4. The value of R_{max} obtained from the incumbent solution is used in the application of Proposition 3. Since our objectives have the "minimax" form, the incumbent R_{max} plays the role of an upper bound, and any partial solution with a larger R_{max} (or lower bound on R_{max}) can be fathomed. Lower bounds on the workloads obtainable at each node (and the consequent lower bounds on R_{max}) are computed using the formula in Proposition 3. Proposition 4 provides for fathoming of nodes when the upper bound on the workload of a machine group is so small that another easily-constructed solution with a larger workload on this underloaded group (and thus also a smaller workload on an overloaded group) will perform as well or better. The algorithm terminates when no unconsidered number groupings and no dangling nodes remain in the B&B tree.

3.5. Modifications for the Balancing Case

Since all groups are identical with respect to processing time capacity and the tool magazine capacity, the combinations of number groupings can be reduced by replacing the first set of constraints in *NGUB* by a constraint that the i-th group should have at least as many operations as the (i–1)-th group, i.e., $k_1 \le k_2, \le \cdots \le k_M$. When the group sizes are not all equal, this constraint cannot be employed. Thus, for a given number of operations and machine groups, the cardinality of all the groupings generated from *NGUB* is usually much larger than that of the comparable balancing problem, as illustrated in Figure 2.

The propositions should be modified by noting that for the case of balancing:

(a) $IX_j = \sum_{i=1}^{b} p_i \; / \; M$ for all j, and

(b) $R_{max}\, IX_j$ is simply is maximum workload among the groups.

Rule 2 should be modified as follows:

Rule 2. When there are two or more groups that have the same number of operations assigned, the index of the first operation of i-th group should be greater than that of (i–1)-th group.

Also, because all groups are identical in this case, Proposition 1 can be strengthened to consider multiple machine groups simultaneously. See Kim and Yano (1989) for details.

4. Computational Results

Since flexible manufacturing systems differ widely, a solution procedure which performs well on one system is not guaranteed to perform well on other systems. Moreover, even for a particular system, the demand characteristics can affect the performance of a solution procedure. Since our approach to the loading problem is designed to be general rather than to be system-specific, it is desirable to test the algorithms on a wide range of problems. Unfortunately, we were unable to obtain actual data for a wide variety of systems, partly because of the proprietary nature of such data. However, the test problems are generated in such a way that the resulting data represent characteristics of real systems relatively well. (In fact, the parameters for the problems came from one author's experience at a manufacturing company. The resulting data are reflective of FMSs within the company.)

Two sets of loading problems are generated randomly, one for balancing and one for unbalancing. Each set has 30 problems with 10 to 20 operations and 2 to 5 machine groups. For each problem, the other required data are generated as described below:

1) The tool magazine capacity (the number of tool slots in the magazine) ranged from 50 to 90, which is a selected parameter for each problem.
2) The processing time for each operation is generated from a discrete uniform distribution between the minimum and the maximum processing times, which are selected parameters for each problem. The minimum value ranged from 10 to 30, and the maximum value ranged from 40 to 100.

3) The total number of tools considered in each problem lies between 50 to 100, where the specific value is selected for each problem.

4) The number of tool slots needed for each tool is chosen randomly from 1, 3, and 5 with probabilities of 0.7, 0.25, and 0.05 for the values, respectively.

5) The number of tools needed for each operation is generated randomly from a discrete uniform distribution between 4 and 10.

6) Tools are assigned to operations as follows. The number of operations requiring each tool is generated from a discrete uniform distribution between 1 and 4. For each tool, the indices of operations sharing it are randomly selected from among those operations for which the number of tools from Step 5 had not yet been assigned. When there is a conflict between the results of Step 5 and the number of operations requiring specific tools (which occurs because of interdependency of these two sets of data), the former has priority. That is, the latter is modified as long as it does not exceed a maximum of 4.

7) A set of tools used solely for a single operation is treated as a single tool for simplicity. The number of tool slots needed for the single artificial tool is the sum of the tool slots required by the individual tools.

For the two sets of test problems, we tested both the algorithm of Berrada and Stecke (and a modification of it for the case of unbalancing, as described in Section 2), and our new algorithm. These are referred to as B&B1, and B&B2, respectively. These procedures, both of which were coded in FORTRAN, were run for up to one hour of CPU time on a personal computer with an 80486 processor. The computer code for the original branch and bound algorithm was provided by Berrada and Stecke. Note that their procedure is an ε-optimal algorithm. Their algorithm was run with two different values of ε. The first value is the same as in the original code, and is equal to the shortest operation processing time divided by the number of machine groups (the same as in their original code). The second value is set to a smaller value with the intent of finding a closer-to-optimal solution). We used $\varepsilon = 0.999$ for the balancing objective and $\varepsilon = 1./\max_l IX_l$ for unbalancing. These values are those that yielded the best results after much trial and error.

To distinguish the performance of the B&B procedures from that of the initialization routines, we used the heuristic of Kim and Yano (1987) to generate the initial incumbent solution in all cases. This heuristic is reported to perform well and to be expected to produce near-optimal or optimal solutions for a range of test problems.

The results are shown in Tables 1 and 2. Two performance measures appear in the tables. The maximum deviation ratio is the maximum ratio of actual workload to the ideal workload among machine groups, and expected production rates (EPR) in the tables were calculated on the basis of the actual workload (from the solution) using the closed queueing network model of Stecke and Solberg (1985). The latter measure was included to compare the performance of the algorithms for the final objective, maximizing the expected production rate.

>> *Insert Tables 1 and 2 here* <<

In the experiments, B&B2 outperformed B&B1 in both computation time and solution quality. For the balancing objective, most of the heuristic solutions used as the initial incumbent solutions were ε-optimal in B&B1. As mentioned earlier, to ascertain whether the procedure could improve upon the initial incumbent and find closer-to-optimal solutions, we ran B&B1 with the smaller ε value for one CPU hour, and obtained an improvement in a few problems. On the other hand, B&B2 found optimal solutions for most problems in much less time.

Similar results can be seen in the case of the unbalancing objective. B&B2 by far outperformed B&B1 in this case, also. In most problems for which B&B1 did not improve on the initial solution, B&B2 found an optimal solution or a better feasible solution. Although a few of the solutions from B&B2 have not been verified as optimal, they are good in an absolute sense, since the maximum deviation ratios of those solutions are less than 1.015, which indicates that the actual workloads are within 1.5% of the ideal workloads. Also, the solutions from B&B2 gave higher (or equal) expected production rates than those from B&B1 in all test problems except for problem 5. It should be noted, however, that in this problem, the surrogate objective value of the solution from B&B2 is better than that from B&B1. Thus, the small discrepancy in the expected production rates is a consequence of a less-than-perfect correspondence between the surrogate and true objectives.

There are two probable reasons why B&B2 works better. First, Propositions 1 and 2 eliminate a significant percentage of the possible number groupings, as shown in Table 3. Second, bounds are determined by solving small zero-one problems in the Berrada and Stecke procedure, which may require a significant amount of CPU time. On the other hand, in our procedure, lower bounds are obtained very quickly without solving optimization problems. These two factors are the primary differences between the new algorithm and the Berrada and Stecke procedure, and consequently, are likely to be responsible for much of the observed improvements.

Since each "limb" (sub-tree) of the branch and bound tree is constructed for a particular number grouping, the depth of the tree is no greater than the number of operations to be assigned, although in most cases, branches are fathomed before the full depth is reached. To provide an indication of the sizes of the trees, we also report in Table 3 percentages of nodes considered (100 times the ratio of the number of nodes generated in the algorithm to the maximum number of nodes that might have been generated without fathoming). The percentage decreases as the problem size increases, which demonstrates the power of the propositions in fathoming branches.

>> *Insert Table 3 here* <<

B&B2 is slower for the unbalancing objective than for the balancing objective. This was expected because more number groupings must be considered. Moreover, for the unbalancing case, the properties used to eliminate number groupings and fathoming partial assignments are not as powerful as those for the balancing case.

5. Summary and Concluding Remarks

The primary contribution of this paper is a more efficient branch and bound procedure for the problem of allocating operations to machine groups so as to maximize throughput while satisfying tool or component storage constraints. To the best of our knowledge, this algorithm is the first to consistently obtain truly optimal solutions for problems with up to 20 operations. When the number of machines in each group is equal, the objective is to minimize the maximum workload among the machine groups. Otherwise, we use the objective of minimizing the maximum ratio of the assigned workload to the ideal workload among the machine groups. These objectives are shown to outperform other similar objectives.

We generalize and improve an existing B&B algorithm which was designed for the case of equal machine group sizes. We show that a simple transformation of the data makes this algorithm applicable to the case of unequal machine groups. We develop a more efficient branch and bound algorithm, which uses several new properties that significantly reduce the number of alternatives to be considered. The new properties allowed us to find optimal (or in a few cases, better feasible solutions) for problems which could not be solved optimally by the existing B&B approach.

Our branch and bound algorithm can be applied in other contexts as well. For example, it can be applied directly to scheduling problems with the objective of minimizing makespan when there are identical or uniform processors. The only modification required is to eliminate the routine for checking feasibility of the tool magazine capacity constraints.

The results in this paper suggest that problems of reasonable size can be solved optimally or near-optimally. In practice, heuristic solutions such as those obtained using the heuristics of Kim and Yano (1987), which we used as initial incumbent solutions in our study, may be more than adequate. However, algorithms that allow us to solve problems to optimality are useful in their own right, even if their primary purpose is to serve as a benchmark in the evaluation of heuristics. In many problem arenas, optimal solutions cannot be found, even for relatively small problems, and the development of good heuristics has been hampered as a consequence.

Further research is needed to incorporate other objectives. For example, we do not consider minimizing material movements. Further research is also needed to permit processing times to vary across machine groups. In addition, sharper bounds are needed for the suggested branch and bound algorithms, since the computing time needed for larger problems is expected to be excessive.

Appendix

Comparison of Surrogate Objectives

The following measures of (un)balance are evaluated. All objectives are to be minimized. X_l and IX_l denote actual load and ideal load for machine group l (not per machine), respectively.

$$C1 = \max_l \, (X_l - IX_l) \qquad C7 = \max_l \, (X_l - IX_l) / IX_l$$

$$C2 = \max_l \, (IX_l - X_l) \qquad C8 = \max_l \, (IX_l - X_l) / IX_l$$

$$C3 = C1 + C2 \qquad C9 = C7 + C8$$

$$C4 = \max_l \, | X_l - IX_l | \qquad C10 = \max_l \, | X_l - IX_l | / IX_l$$

$$C5 = \sum_l | X_l - IX_l | \qquad C11 = \sum_l | X_l - IX_l | / IX_l$$

$$C6 = \sum_l (X_l - IX_l)^2 \qquad C12 = \sum_l (X_l - IX_l)^2 / IX_l$$

If C1 is used, the maximum workload among the groups is to be minimized, while if C2 is used the minimum workload is to be maximized. The sum of these two, C3, can be another measure of closeness. C4 is the maximum deviation of actual workload from the ideal workload and is equal to the maximum of C1 and C2. C5 is the sum of the deviations and C6 is the sum of squared deviations. These six measures are considered because they are commonly used to measure deviation from certain standard or target values (e.g., sum of absolute errors, sum of squared errors, maximum absolute error, etc.). C7 through C12 are the counterparts of C1 through C6 with the reciprocal of the ideal workload as a weighting factor for each group. These measures reflect *relative*, rather than absolute deviations. Note that the measures C7 through C12 are equivalent to the measures C1 through C6 for the objective of balancing workloads.

To compare these measures, we tested them on fifty system configurations using both balancing and unbalancing objectives. A configuration is defined by the number of machine groups (ranging from 3 to 7) and the number of machines in each group (ranging from 1 to 6). For each configuration, fifty scenarios with randomly generated actual loads for the machine groups were created. To calculate the expected production rate for a given configuration and workload assignment, the closed queueing network (CQN) model of Stecke and Solberg (1985) was used. For each system configuration, the ideal loads and the expected production rate were calculated with the CQN model, and the values of the measures of (un)balance and the expected production were also calculated for each of the fifty randomly generated workloads.

For each system configuration, we calculated the coefficient of correlation between the value of the measure and the corresponding expected production rate for the 50 randomly generated scenarios. To test for differences among the correlation coefficients, paired t tests were done for all pairs of measures. The paired t test was used since the correlation coefficients were computed for 50 different configurations. This lack of homogeneity among configurations contributes to the variability of the coefficients and tends to inflate the experimental error, thus making a true difference between measures harder to detect. The paired t test eliminates the problem of non-homogeneity by using matched pairs of data. In the paired t test, the difference between the correlation coefficients for the two measures is computed first for each configuration, then these differences for the 50 configurations are used in the statistical analysis. Thus, the differences between two measures for the 50 configurations constitute the population and the null hypothesis is that the differences are not statistically different from zero. (See Montgomery 1984 for details of the method and a discussion of the advantages of using paired comparisons.)

>> *Insert Tables A1 and A2 here* <<

The results are shown in Tables A1 and A2. Table A1 shows means of the correlation coefficients ($\bar{r}$) and the results of the paired t tests for the balancing objective, while Table A2 shows the same for the unbalancing objective. For the balancing case, C1 significantly dominates the others. This justifies the objective function in the formulation of Berrada and Stecke (1986), minimizing the maximum workload. For the unbalancing case, however, C1 does not work very well. Instead, C7 (the maximum ratio of overload to the ideal load among the machine groups) works well in this case. C9, whose value is the sum of C7 and C8, also works well, but this may be due to the effect of C7.

References

Ammons, J. C., C. B. Lofgren and L. F. McGinnis. "A Large Scale Machine Loading Problem in Flexible Assembly." *Annals of Operations Research*, Vol.3, pp. 319-322 (1985).

Berrada, M. and K. E. Stecke. "A Branch and Bound Approach for Machine Load Balancing in Flexible Manufacturing Systems." *Management Science*, Vol.32, No.10, pp. 1316-1335 (October 1986).

Carrie, A. S. and D. T. S. Perera. "Work Scheduling in FMS under Tool Availability Constraints." *International Journal of Production Research*, Vol.24, No.6, pp. 1299-1308 (June 1986).

Dallery, Y. and Stecke, K.E. "On the Optimal Allocation of Servers and Workloads in Closed Queueing Networks," *Operations Research*, Vol. 37, No. 4, pp. 694-703 (July-August 1990).

Greene, T. J. and R. P. Sadowski. "A Mixed Integer Program for Loading and Scheduling Multiple Flexible Manufacturing Cells." *European Journal of Operational Research*, Vol.24, No. 3, pp. 379-386 (March 1986).

Kim, Y-D. and C. A. Yano. "A Heuristic Approach for Loading Problems in Flexible Manufacturing Systems." Technical Report 87-21, Dept. of Industrial and Operations Engineering, The University of Michigan, Ann Arbor, Michigan (September 1987), (to appear in *IIE Transactions*).

Kim, Y-D. and C. A. Yano. "A New Branch and Bound Approach for Loading Problems in Flexible Manufacturing Systems." Technical Report 89-5, Dept. of Industrial and Operations Engineering, The University of Michigan (February 1989).

Kiran, A. S. and B. C. Tansel. "The System Setup in FMS: Concepts and Formulation." *Proceedings of the 2nd ORSA/TIMS Conference on Flexible Manufacturing Systems,* Elsevier Science Publishers B.V., Amsterdam, pp. 321-332 (August 1986).

Kusiak, A. "Loading Models in Flexible Manufacturing Systems" in *Flexible Manufacturing: Recent Development on FMS, Robotics, CAD/CAM, CIM,* Raouf, A. and Ahmad, S. I. (Eds.), Elsevier Science Publishers B.V., Amsterdam, pp. 119-132 (1985).

Lashkari, R. S., S. P. Dutta, and A. M. Padhye. "A New Formulation of Operation Allocation Problem in Flexible Manufacturing Systems: Mathematical Modelling and Computational Experience." *International Journal of Production Research*, Vol.25, No.9, pp. 1267-1283 (September 1987).

Montgomery, D. C., 1984. *Design and Analysis of Experiments*. John Wiley & Sons, Inc., New York.

Rajagopalan, S. "Formulation and Heuristic Solutions for Parts Grouping and Tool Loading in Flexible Manufacturing Systems." *Proceedings of the 2nd ORSA/TIMS Conference on Flexible Manufacturing Systems,* Elsevier Science Publishers B.V., Amsterdam, pp. 312-314 (August 1986).

Sarin, S. C. and C. S. Chen. "The Machine Loading and Tool Allocation Problem in a Flexible Manufacturing System." *International Journal of Production Research*, Vol.25, No.7, pp. 1081-1094 (July 1987).

Shanker, K. and Y-J. J. Tzen. "A Loading and Dispatching Problem in a Random Flexible Manufacturing System." *International Journal of Production Research*, Vol.23, No. 3, pp. 579-595 (May-June 1985).

Shanthikumar, J. G. and K. E. Stecke. "Reducing Work-in-Process Inventory in Certain Classes of Flexible Manufacturing Systems." *European Journal of Operational Research*, Vol.26, No. 2, pp. 266-271 (August 1986).

Shanthikumar, J.G. and D.D. Yao. "On Server Allocation in Multiple Center Manufacturing Systems," *Operations Research*, Vol. 36, No. 2, pp. 333-342 (March-April 1988).

Shanthikumar, J.G. and D. D. Yao. "Optimal Server Allocation in a System of Multi-Server Stations," *Management Science*, Vol 33., No. 9, pp. 1173-1180 (September 1987).

Stecke, K. E. "Formulation and Solution of Nonlinear Integer Production Planning Problems for Flexible Manufacturing Systems." *Management Science*, Vol.29, No.3, pp. 273-288 (March 1983).

Stecke, K. E., 1986. "A Hierarchical Approach to Solving Machine Grouping and Loading Problems of Flexible Manufacturing Systems." *European Journal of Operational Research*, Vol.24, No. 3, pp. 369-378 (March 1986).

Stecke, K. E. and T. L. Morin. "The Optimality of Balancing Workloads in Certain Types of Flexible Manufacturing Systems." *European Journal of Operational Research*, Vol.20, No. 1, pp. 68-82 (April 1985).

Stecke, K. E. and J. J. Solberg. "The Optimality of Unbalancing Both Workloads and Machine Group Sizes in Closed Queueing Networks of Multi-Server Queues." *Operations Research*, Vol.33, No.4, pp. 882-910 (July-August 1985)

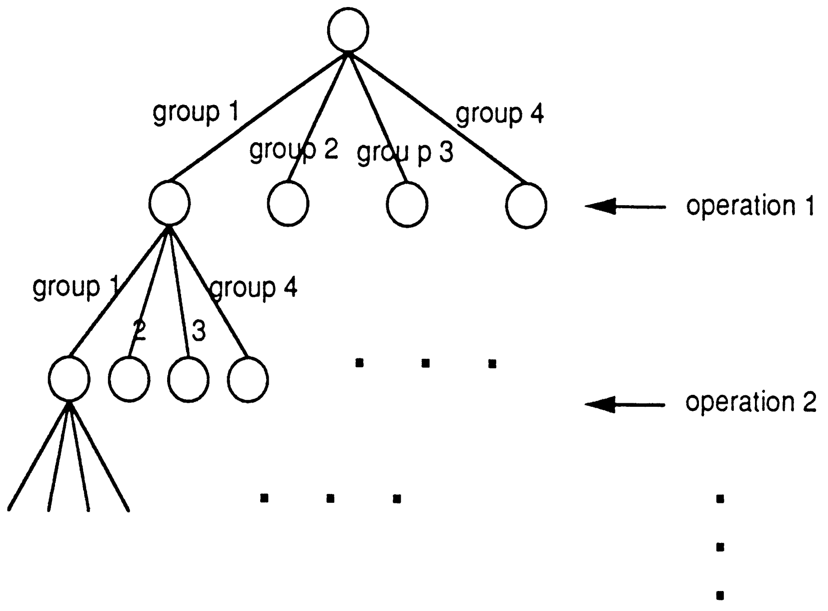
Stecke, K.E. and J.J. Solberg. "Loading and Control Policies for a Flexible Manufacturing System," *International Journal of Production Research*, Vol. 19, No. 5, pp. 481-490 (May 1981).

Stecke, K. E. and F. B. Talbot. "Heuristics for Loading Flexible Manufacturing Systems." *Flexible Manufacturing: Recent Developments in FMS, Robotics, CAD/CAM, CIM,* Raouf, A. and Ahmad, S.I., (Eds.), Elsevier Science Publishers B.V., Amsterdam, pp. 73-85 (1985).

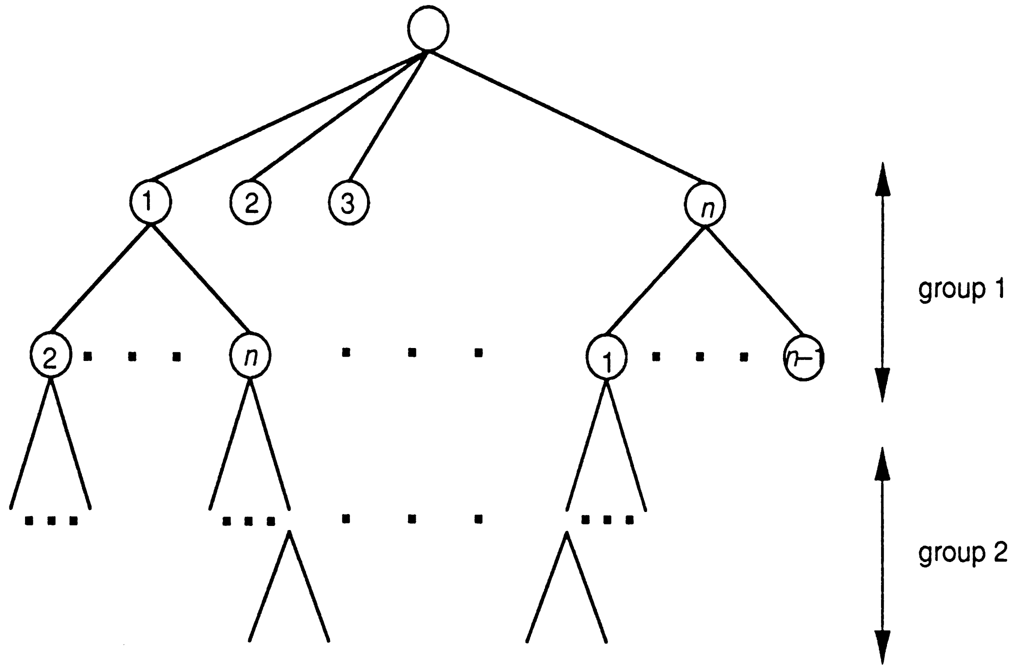
Yao, D. D. "Some Properties of the Throughput Function of Closed Networks of Queues." *Operations Research Letters*, Vol.3, No.6, pp. 313-317 (April 1985).

Yao, D. D. and S. C. Kim. "Some Order Relations in Closed Networks of Queues with Multiserver Stations." *Naval Research Logistics*, Vol.34, No.1, pp. 53-66 (February 1987a).

Yao, D. D. and S. C. Kim. "Reducing the Congestion in a Class of Job Shops." *Management Science*, Vol.33, No.9, pp. 1165-1172 (September 1987b).



a) each level corresponds to an operation



b) each level corresponds to a machine group
(numbers in the nodes denote indices of operations)

Figure 1. Two types of branch and bound trees for the loading problem

Table 1. Computational results for the balancing objective

	B&B1 (ε-optimal)						B&B2 (optimal)		
	with original ε value			with smaller ε value					
Problem†	Max. Dev. Ratio	EPR	CPU Time (sec.)	Max. Dev. Ratio	EPR	CPU Time (sec.)	Max. Dev. Ratio	EPR	CPU Time (sec.)
1 (10,2)	# 1.002	.937	0.77	*‡		3600.	# 1.002	.937	0.83
2 (10,2)	# 1.002	.952	0.77	*‡		3600.	# 1.002	.952	0.77
3 (10,3)	# 1.024	.879	1.21	*‡		3600.	1.005	.882	1.55
4 (10,3)	# 1.044	.897	1.15	*‡		3600.	1.008	.909	1.42
5 (10,3)	# 1.043	.900	2.64	*‡		3600.	1.006	.909	3.01
6 (12,2)	# 1.004	.952	1.15	*‡		3600.	1.001	.952	1.15
7 (12,3)	# 1.007	.909	0.94	‡		9.93	# 1.007	.909	0.94
8 (12,3)	# 1.005	.909	0.88	‡		10.89	# 1.005	.909	0.93
9 (12,4)	# 1.057	.859	10.18	‡		44.08	1.042	.859	12.85
10 (12,4)	# 1.006	.869	0.98	‡		10.47	# 1.006	.869	0.98
11 (14,3)	# 1.040	.902	1.26	1.010	.908	64.33	1.000	.909	1.76
12 (14,3)	# 1.033	.906	2.25	1.005	.909	18.28	1.005	.909	2.30
13 (14,4)	1.001	.870	8.80	‡		10.62	1.001	.870	2.22
14 (14,4)	# 1.038	.865	20.92	‡		194.60	1.005	.869	44.56
15 (14,5)	# 1.047	.827	27.57	‡		113.08	1.012	.833	68.94
16 (16,3)	# 1.000	.909	0.16	‡		0.16	# 1.000	.909	0.16
17 (16,3)	# 1.003	.926	1.87	‡		298.57	1.000	.926	2.08
18 (16,4)	# 1.018	.892	1.71	‡		1668.68	1.000	.893	21.16
19 (16,4)	# 1.022	.891	1.31	‡		285.55	1.001	.893	32.87
20 (16,5)	# 1.044	.859	77.67	‡		447.81	1.002	.862	1.96
21 (18,3)	# 1.006	.909	1.43	‡		769.27	1.002	.909	1.43
22 (18,4)	# 1.008	.893	1.54	‡		263.74	1.004	.893	1.54
23 (18,4)	# 1.013	.892	2.19	‡		548.43	1.000	.893	2.25
24 (18.5)	# 1.016	.861	1.54	‡		672.15	1.001	.862	1.59
25 (18,5)	# 1.027	.878	1.60	*‡		3600.	1.004	.882	315.55
26 (20,3)	# 1.008	.909	1.48	*‡		3600.	1.001	.909	3.35
27 (20,4)	# 1.010	.893	1.48	‡		839.29	1.004	.893	1.53
28 (20,4)	# 1.004	.893	1.92	‡		376.03	# 1.004	.893	1.97
29 (20,5)	# 1.008	.862	2.09	*‡		3600.	1.003	.862	11.48
30 (20,5)	# 1.014	.881	483.94	‡		1218.89	*# 1.014	.881	3600.

† The numbers in parentheses denote the numbers of operations and machine groups.
* Incumbent solutions after 3600 seconds of CPU time on the personal computer.
Final solution was the same as the initial incumbent solution obtained from the heuristic.
‡ The solution is the same as the epsilon-optimal solution with original epsilon value.

Table 2. Computational results for the unbalancing objective

Problem†	B&B1 (ε-optimal)						B&B2 (optimal)		
	with original ε value			with smaller ε value					
	Max. Dev. Ratio	EPR	CPU Time (sec.)	Max. Dev. Ratio	EPR	CPU Time (sec.)	Max. Dev. Ratio	EPR	CPU Time (sec.)
1 (10,2)	# 1.003	.941	0.71	*‡		3600.	1.001	.941	0.76
2 (10,2)	# 1.019	.951	0.82	‡		0.90	1.002	.955	0.87
3 (10,3)	1.006	.886	3.15	*‡		3600.	1.003	.886	1.03
4 (10,3)	# 1.005	.919	0.83	*‡		3600.	# 1.005	.919	0.87
5 (10,3)	# 1.013	.919	0.99	*‡		3600.	1.006	.918	1.03
6 (12,2)	# 1.007	.955	0.77	1.003	.956	67.28	1.001	.956	0.82
7 (12,3)	# 1.008	.912	0.87	*‡		3600.	1.007	.912	2.02
8 (12,3)	# 1.014	.917	50.96	*‡		3600.	1.006	.918	1.29
9 (12,4)	# 1.057	.854	276.27	1.029	.868	3600.	1.029	.868	3.45
10 (12,4)	# 1.023	.877	102.65	*‡		3600.	1.004	.880	1.48
11 (14,3)	1.002	.913	790.91	‡		973.87	1.002	.913	8.10
12 (14,3)	# 1.003	.919	0.99	*‡		3600.	1.001	.919	1.72
13 (14,4)	# 1.013	.872	703.85	‡		2218.77	1.002	.873	11.95
14 (14,4)	# 1.036	.871	2017.05	*‡		3600.	1.004	.879	6.57
15 (14,5)	1.044	.826	1471.35	*‡		3600.	1.005	.836	31.03
16 (16,3)	*# 1.018	.910	3600.	*‡		3600.	1.000	.913	5.33
17 (16,3)	*# 1.029	.925	3600.	*‡		3600.	1.001	.934	7.75
18 (16,4)	# 1.014	.895	1.32	*‡		3600.	1.002	.896	329.53
19 (16,4)	*# 1.031	.900	3600.	*‡		3600.	1.001	.902	46.50
20 (16,5)	*# 1.064	.845	3600.	*‡		3600.	1.006	.865	431.69
21 (18,3)	# 1.004	.913	1.32	*‡		3600.	1.000	.913	3.53
22 (18,4)	*# 1.024	.895	3600.	*‡		3600.	1.000	.896	206.60
23 (18,4)	*# 1.024	.895	3600.	*‡		3600.	1.002	.902	418.85
24 (18.5)	*# 1.031	.859	3600.	*‡		3600.	1.001	.865	1698.91
25 (18,5)	*# 1.029	.889	3600.	*‡		3600.	1.001	.890	1538.47
26 (20,3)	*# 1.010	.912	3600.	*‡		3600.	1.002	.913	2581.80
27 (20,4)	*# 1.014	.894	3600.	*‡		3600.	*# 1.014	.894	3600.
28 (20,4)	*# 1.026	.897	3600.	*‡		3600.	* 1.002	.902	3600.
29 (20,5)	# 1.004	.865	1.32	*‡		3600.	* 1.004	.865	3600.
30 (20,5)	*# 1.012	.888	3600.	*‡		3600.	* 1.004	.890	3600.

†, ‡, #, * See the footnotes of Table 1.

Table 3. Percentages of number groupings and nodes considered by B&B2

Problem	Balancing			Unbalancing		
	% of possible no. groupings considered		% of nodes considered	% of possible no. groupings considered		% of nodes considered
1 (10,2)	40.0	(2/5)	0.879	22.2	(2/9)	6.25
2 (10,2)	40.0	(2/5)	0.830	22.2	(2/9)	6.64
3 (10,3)	12.5	(1/6)	1.75	15.0	(3/20)	0.978
4 (10,3)	25.0	(2/8)	1.81	10.0	(2/20)	0.242
5 (10,3)	12.5	(1/8)	2.06	10.0	(2/20)	0.269
6 (12,2)	33.3	(2/6)	0.476	27.3	(3/11)	1.41
7 (12,3)	16.7	(2/12)	0.00489	6.7	(2/30)	0.763
8 (12,3)	25.0	(3/12)	0.00427	26.7	(8/30)	0.237
9 (12,4)	13.3	(2/15)	0.258	7.3	(3/41)	0.0526
10 (12,4)	26.7	(4/15)	1.79E–4	29.3	(12/41)	0.0125
11 (14,3)	18.8	(3/16)	0.0329	9.5	(4/42)	0.464
12 (14,3)	6.3	(1/16)	0.00142	4.8	(2/42)	0.0498
13 (14,4)	21.7	(5/23)	9.51E–4	14.9	(10/67)	0.0130
14 (14,4)	21.7	(5/23)	0.0522	20.9	(14/67)	0.00668
15 (14,5)	13.0	(3/23)	0.00395	28.2	(20/71)	0.00164
16 (16,3)	‡ 0	(0/21)	‡ 0	8.9	(5/56)	0.0241
17 (16,3)	14.3	(3/21)	0.00151	8.9	(5/56)	0.0502
18 (16,4)	5.9	(2/34)	0.00143	6.9	(7/102)	0.0250
19 (16,4)	8.8	(3/34)	0.00231	3.9	(4/102)	0.00357
20 (16,5)	5.4	(2/37)	3.17E–7	6.6	(8/121)	9.25E–4
21 (18,3)	18.5	(5/27)	1,29E–5	11.1	(8/72)	0.00158
22 (18,4)	8.5	(4/47)	1.16E–7	10.9	(16/147)	9.27E–4
23 (18,4)	10.6	(5/47)	2.14E–7	6.1	(9/147)	0.00195
24 (18,5)	15.8	(9/57)	6.08E–9	10.3	(20/194)	1.42E–4
25 (18,5)	3.5	(2/57)	2.63E–5	2.6	(5/194)	1.37E–5
26 (20,3)	18.2	(6/33)	1.42E–4	13.3	(12/90)	0.199
27 (20,4)	3.1	(2/64)	7.78E–9	3.4	(7/204)	† 0.00104
28 (20,4)	10.9	(7/64)	6.96E–9	7.8	(16/204)	† 0.00102
29 (20,5)	22.6	(19/84)	2,83E–8	12.9	(38/295)	† 1.14E–5
30 (20,5)	10.7	(9/84)	† 1.11E–5	6.8	(20/295)	† 1.19E–5

† These are lower bounds of the actual percentages, since the problems were not solved optimally.
‡ Initial feasible solution was perfectly balanced.

Table A1. Test results for difference of the correlation coefficients (Balancing Case)

Measure ($\bar{r}$ †)	C1	C3	C6	C4	C5	C2
C1 (-.979)		**	**	**	**	**
C3 (-.946)			**	**	**	**
C6(-.925)				=	**	**
C4 (-.919)					**	**
C5 (-.892)						**
C2 (-.546)						

† mean value of the correlation coefficients from the 50 configurations
** : statistically different at significance level of .01
= : not statistically different at significance level of .05

Table A2. Test results for difference of the correlation coefficients (Unbalancing Case)

Measure ($\bar{r}$ †)	C7	C9	C11	C10	C12	C5	C3	C1	C4	C6	C2	C8
C7 (-.945)		**	**	**	**	**	**	**	**	**	**	**
C9 (-.938)			**	**	**	**	**	**	**	**	**	**
C11 (-.917)				=	**	**	**	**	**	**	**	**
C10 (-.916)					**	**	**	**	**	**	**	**
C12 (-.897)						**	**	**	**	**	**	**
C5 (-.853)							*	**	**	**	**	**
C3 (-.840)								**	**	**	**	**
C1 (-.813)									=	**	**	**
C4 (-.811)										**	**	**
C6 (-.793)											**	**
C2 (-.750)												**
C8 (-.288)												

† mean value of the correlation coefficients from the 50 configurations
** : statistically different at significance level of .01
* : statistically different at significance level of .05
= : not statistically different at significance level of .05