ERASMUS

ECONOMETRIC INSTITUTE

A SURVEY OF ALGORITHMS FOR THE GENERALIZED ASSIGNMENT PROBLEM

D.G. CATTRYSSE AND L.N. VAN WASSENHOVE

REPORT 9010/A

ERASMUS UNIVERSITY ROTTERDAM - P.O. BOX 1738 - 3000 DR ROTTERDAM - THE NETHERLANDS

A SURVEY OF ALGORITHMS FOR THE GENERALIZED ASSIGNMENT PROBLEM

Dirk G. Cattrysse

Katholieke Universiteit Leuven
Afdeling Industrieel Beleid
Celestijnenlaan 300A
3030 Leuven Belgium

Luk N. Van Wassenhove

Erasmus Universiteit Rotterdam
Econometrisch Instituut
P.O.Box 1738
3000 DR Rotterdam, The Netherlands

Abstract

This paper surveys algorithms for the well-known problem of finding the minimum cost assignment of jobs to agents so that each job is assigned exactly once and agents are not overloaded. All approaches seem to be based on branch and bound with bounds supplied through heuristics and through relaxations of the primal problem formulation. From the survey one can select building blocks for the design of one's own tailor-made algorithm. The survey also reveals that although just about any mathematical programming technique was tried on this problem, there is still a lack of a representative set of test problems on which competing enumeration algorithms can be compared, as well as a shortage of effective heuristics.

A SURVEY OF ALGORITHMS FOR THE GENERALIZED ASSIGNMENT PROBLEM

1. INTRODUCTION

The generalized assignment problem (GAP) examines the minimum cost assignment of n jobs to m agents such that each job is assigned to exactly one agent subject to capacity restrictions on the agents.

The generalized assignment problem is an NP-hard combinatorial optimization problem (Fisher, Jaikumar and Van Wassenhove (1986)). A lot of research has been done over the last ten years to find effective enumeration algorithms to solve problems of reasonable size to optimality. The GAP has many real-life applications ,e.g., as a subproblem in routing problems (Fisher and Jaikumar (1981)), in fixed-charge plant location models in which customer requirements must be satisfied by a single plant, resource scheduling, scheduling of project networks, storage space allocation, in designing communications networks with node capacity constraints (Grigoriadis, Tang and Woo (1974)), scheduling of payments on accounts where 'lump sum' payments are specified, assigning software development tasks to programmers, assigning jobs to computers in computer networks (Balachandran (1972)), scheduling variable length television commercials into time slots, etc. Ross and Soland (1977) show how the p-median problem, the capacity constrained p-median problem and the plant location problem can be modelled as GAPs.

The formulation of the problem is :

$$\text{(GAP)} \quad \min \sum_i \sum_j c_{ij}\, x_{ij} \tag{1}$$

$$\text{s.t.} \quad \sum_j a_{ij}\, x_{ij} \le b_i \qquad i \in I \tag{2}$$

$$\sum_i x_{ij} = 1 \qquad j \in J \tag{3}$$

$$x_{ij} = 0 \text{ or } 1 \qquad i \in I,\ j \in J \tag{4}$$

where c_{ij} is the cost of assigning job j to agent i, a_{ij} the capacity absorption when job j is assigned to agent i, b_i the available capacity of agent i. The assignment variable x_{ij} equals 1 if agent i is to perform job j, 0 otherwise.

Most algorithms are based on branch and bound techniques and on relaxation of the assignment or the knapsack constraints. Sometimes valid inequalities are added to strengthen the bounds in a relaxation. The methods mainly differ in the way the bounds are computed in the branch and bound search. One can relax constraints (2), (3) or (4), use Lagrangean relaxation or surrogate relaxation. Furthermore, multipliers can be updated via the subgradient method or via a heuristic multiplier adjustment method. For more information on different types of relaxations we refer to Fisher (1981), (1985) and to Gavish and Pirkul (1985a), (1985b).

The purpose of this paper is to present a review of the different kinds of relaxations used to obtain bounds for the generalized assignment problem on the one hand and to compare the strength of these bounds as well as different branching strategies on the other hand (section 2 - 6). Section 7 presents a summary and discusses how an effective algorithm could be constructed. Section 8 discusses some extensions of the GAP : the 0-1 generalized assignment problem with nonlinear capacity constraints (NLGAP) and the multiconstraint generalized assignment problem (MCGAP). Finally, some conclusions are drawn in section 9.

2. LP-RELAXATION

The first relaxation we consider is the LP-relaxation obtained by replacing constraints (4) by $x_{ij} \geq 0$. The solution of the relaxed problem will in general contain some jobs j for which $x_{ij} \neq 0$ for several i. In other words, job j is split between several agents. An upper bound on the number of split jobs is given by Benders and van Nunen (1983). The number of non-unique assignments is less than or equal to the number of agents of which the capacity is fully utilized. When the number of agents is small compared to the number of jobs, the

solution of the relaxed problem may be a good starting point for a heuristic. The latter provides an upper bound while the LP-relaxation obviously yields a lower bound. The LP-bound tends to be strong when the number of jobs is large compared to the number of agents and when the capacity constraints are rather loose.
In literature very little can be found on the performance of LP-based approaches. Benders and van Nunen (1983) discuss the performance of a heuristic based on the LP-solution that assigns the remaining split jobs. In the practical problems (with number of jobs >> number of agents) that were solved the heuristic led to solutions within 0.1% of the lower bound and consequently within 0.1% of the optimal solution. When the same procedure is applied to problem sets that are strongly capacitated and with ratios of $|J|/|I|$ equal to 3 or 4, reassigning the split jobs often turns out to be infeasible and the gap between the optimal solution and the LP-bound is much larger.

A drawback of the LP-formulation of the GAP is that it is degenerate. As a result computation time tends to grow fast when dimensions get larger.

3. RELAXATIONS BASED ON DELETING CONSTRAINTS

The algorithm we consider next is discussed in Ross and Soland (1975). The authors first delete constraints (2) :

$$\text{(RS)} \quad \min \{ \sum_i \sum_j c_{ij} x_{ij} \mid (3), (4) \} \qquad (5)$$

The solution to problem (RS) is obtained by assigning each job to the least costly agent. This yields a lower bound. Then every constraint (2) is checked for feasibility. In a second step minimum penalties are computed for reassigning jobs from one agent to another in order to satisfy the capacity restrictions. These penalties z_i are added to the lower bound :

$$(PK_i) \quad \min z_i = \sum_{j \in Ji} p_j y_{ij} \qquad (6)$$

$$\text{s.t.} \sum_{j \in Ji} a_{ij} y_{ij} \geq d_i \qquad (7)$$

$$y_{ij} = 0 \text{ or } 1 \qquad (8)$$

where $d_i = \sum_j a_{ij} x^*_{ij} - b_i$ with x^*_{ij} denoting an optimal solution to (5), p_j the difference between the smallest and second smallest c_{ij} and $J_i = \{j \mid x^*_{ij} = 1\}$.

It should be observed that the same lower bound is obtained by setting the Lagrangean multipliers corresponding to constraints (3) equal to the second smallest c_{ij} and solving the knapsacks in that Lagrangean problem as shown by Fisher, Jaikumar and Van Wassenhove (1986).
Ross and Soland use a binary branching strategy based on the penalty for not using the least cost assignment for a job and on the remaining resources available to the agent. The separation variable x_{ij} is selected from those with $y_{ij} = 0$ in the optimal solution to the (PK_i) so that :

$$t_{ij} = \max \{ p_j / (a_{ij} / (b_i - \sum_j a_{ij} x^*_{ij}))\} \qquad (9)$$

Martello and Toth (1981) use a similar procedure. They reformulate the problem as a maximization problem and remove constraints (3) to determine an upper bound :

$$(MT) \quad \max \{ \sum_i \sum_j p_{ij} x_{ij} \mid (2), (4) \} \qquad (10)$$

$$\text{with } p_{ij} = \max_i \{c_{ij}\} - c_{ij} \qquad i \in I,\ j \in J \qquad (11)$$

This results in $|I|$ single knapsacks (K_i) with solution x^*_{ij} and solution value u_i.

The branching strategy is as follows :

- if a job has not been assigned yet ($\sum_i x^*_{ij} = 0$), it is assigned in turn to every agent (i.e. $|I|$ nodes are created).
- if a job has been assigned more than once (i.e. $\sum_i x^*_{ij} = q,\ q > 1$), $q + 1$ nodes are created.

The authors also try to improve upon the bound by computing a lower bound on the penalty incurred to satisfy constraints (3). This improvement step requires the computation of an upper bound on the solution value u_i by solving continuous versions of the single knapsack problems (K_i). The jobs in the optimal solution to (10) which do not satisfy constraints (3) are divided in two classes : J_0 and J_1. J_0 is the set of unassigned jobs and J_1 is the set of jobs assigned more than once.

Set $v_{ijo} = \min \{u_i,$ (upper bound on (K_i) if $x_{ij}=0)\}$ for all $i\in I,\ j\in J_0$
and $v_{ij1} = \min \{u_i,$ (upper bound on (K_i) if $x_{ij}=1)\}$ for all $i\in I_j,\ j\in J_1$
where $I_j=\{s\,|\,x_{sj}=1\}$

$$\text{then } l_j = \left[\begin{array}{ll} \min\limits_i \{u_i-|v_{ijo}|\} & \text{if } j \in J_0 \\ \\ \sum\limits_{i\in I_j} \{u_i-|v_{ij1}|\} - \max\limits_i \{u_i-|v_{ij1}|\} & \text{if } j \in J_1 \end{array}\right.$$

The improvement is equal to max $\{l_j\}$. The job with the maximum penalty is also the branching variable.

Martello and Toth (1981) also describe a greedy heuristic that iteratively considers all unassigned jobs and determines the job with maximum difference between the largest and second largest weight factor. This job with maximum regret is then assigned to its maximum profit agent. This step is repeated until all jobs are assigned. A second step of the heuristic tries to improve upon the solution by simple interchange arguments. Different weight factors can be used in step 1 : cost, cost/capacity requirement etc. The advantage of the heuristic is that it can be used in a reduction algorithm that attempts to reduce the problem size by fixing variables to one or to zero.

Computational results are given for problem dimensions of maximum 5 agents and 20 jobs. The results of Ross-Soland and Martello-Toth are compared (number of nodes in the tree, CPU-time). For the easy problems the Ross and Soland algorithm performs better (running time) on average. For the hard small problems the Martello and Toth algorithm outperforms (running time and number of nodes) the Ross and Soland method (problems tend to be hard when capacity constraints are tight or when there is a correlation between objective function coefficients and capacity requirements). For the larger problems (# agents between 5 and 20 and # jobs between 50 and 200, i.e. $|J|/|I|$ varies between 2.5 and 40) the Martello-Toth heuristic solution is compared with the optimal solution as given by the Ross-Soland algorithm. The average error of the heuristic is less than 0.1%.

4. LAGRANGEAN RELAXATION

There are two natural Lagrangean relaxations for the generalized assignment problem. The first is obtained by dualizing constraints (3). This reduces the problem to $|I|$ 0-1 knapsack problems. The second relaxation is obtained by dualizing constraints (2). This yields $|J|$ generalized upper bound problems which are trivial to solve.

According to Geoffrion's (1974) integrality property, the Lagrangean relaxation can do no better than the LP-relaxation when the Lagrangean problem is naturally integer (i.e. when the integrality restrictions are superfluous). It follows that the second Lagrangean relaxation in which the subproblems are GUB problems is equivalent to the LP-relaxation. The first relaxation, however, leads to 0-1 knapsack Lagrangean subproblems whose solutions are not naturally integer. Hence the first relaxation may yield bounds superior to the LP-relaxation.

A two-phase heuristic based on the second Lagrangean relaxation is presented by Klastorin (1979). In a first phase a modified subgradient approach is used in searching for the optimal dual solution. The procedure is halted as soon as a primal feasible solution is found. In a second phase a tree search is performed in the neighbourhood of the initial solution in order to find a better primal solution. Klastorin's computational tests show that this procedure yields solutions which deviate from the optimal solutions by 1 % on average. The second phase does not appear to provide significant improvements to the solution of the first phase and is computationally expensive.

A Lagrangean relaxation of the first type is explored in Fisher, Jaikumar and Van Wassenhove (1986). They relax constraints (3) and set the multipliers (μ_j, j=1,..,n) equal to the second largest c_{ij} (maximization problem). This yields the same initial bound as in Ross and Soland.

$$\text{(FJV)} \quad \max \{ \sum_i \sum_j (c_{ij} - \mu_j)\, x_{ij} \mid (2), (4) \} \tag{12}$$

In a first step jobs are assigned to agents having $c_{ij}-\mu_j>0$, then an assignment problem with jobs having $c_{ij}-\mu_j=0$ is solved (jobs are assigned in order of decreasing c_{ij}). In a second step multipliers are adjusted by a heuristic procedure in order to assign more jobs and to strengthen the bound. Let δ_{ij} be the smallest decrease in μ_j required for item j to be included in an optimal solution of knapsack i with solution value $Z_{Di}(\mu)$:

$$\delta_{ij} = Z_{Di}(\mu) - c_{ij} - \max \sum_{h \in J\setminus\{j\}} (c_{ih} - \mu_{ih})\, x^*_{ih} \tag{13}$$

$$\text{s.t.} \qquad \sum_{h \in J\setminus\{j\}} a_{ih}\, x^*_{ih} \le b_i - a_{ij} \tag{14}$$

$$x_{ih} = 0 \text{ or } 1,\ h \in J \setminus \{j\} \tag{15}$$

The job j' with the second smallest δ_{ij}' and $\sum_i x^*_{ij}' = 0$ is chosen. Adjust $\mu_j'=\mu_j'-\delta_{ij}'$ and let i' = arg $\min_i \{d_{ij}'\}$. Only the solution of knapsack i' changes in the Lagrangean solution as a result of the change in μ_j'. The procedure is repeated until no further improvements are possible. The branch and bound algorithm selects the free variable x_{ij} with the largest a_{ij} to branch on.

Results are given for 160 problems with up to 20 jobs and 5 agents. They are compared with those from Ross-Soland and Martello-Toth. The FJV algorithm is about an order of magnitude faster and generates branch and bound trees with a factor of 50 fewer nodes. Larger problems have also been solved. Fisher and Jaikumar (1981) use the method as a subroutine in their vehicle routing algorithm.

Guignard and Rosenwein (1989) elaborate on the procedure of Fisher, Jaikumar and Van Wassenhove (1986). Their algorithm is based upon a Lagrangean dual ascent procedure which exploits violations of constraints (3) not considered in Fisher et al. (1986). They thereby improve upon the FJV bound. Guignard et al. also add a surrogate constraint $\Sigma \Sigma x_{ij} \geq |I|$ or $\Sigma \Sigma x_{ij} \leq |I|$ when constraints (3) are violated. The surrogate constraint is then dualized into the objective function with multiplier θ. The strengthened relaxation is given by :

$$\text{(GR)} \quad \min \{ \sum_i \sum_j (c_{ij} - w_j + \theta) x_{ij} - \theta n \mid (2), (4) \} \qquad (16)$$

Their branching scheme is a combination of depth-first and breadth-first branching. At the root node a subgradient optimization procedure is implemented. As a result fewer nodes and less CPU time are required on average (when compared to Fisher, Jaikumar and Van Wassenhove (1986)). Branching is done on a job j' with multiple assignment. The job selected is the one for which : $j' = \min_i \max_j \{ a_{ij} \mid \sum_i x^*_{ij} > 1 \text{ and } x^*_{ij} = 1 \}$. For each branch a distinct x_{ij}' with $x^*_{ij}' = 1$ in the current solution is fixed at 0. In total $\sum_i x^*_{ij}'$ branches are created and the node considered corresponds to $\max_i \{a_{ij}'\}$.

Wilcox (1989) uses a Lagrangean relaxation of constraints (3) and allows multiple job assignments like Guignard and Rosenwein (1989). The multipliers are not adjusted monotonically, but in a bidirectional way (increase multipliers for unassigned jobs, decrease multipliers of jobs assigned more than once). The adjustment is similar to the approach followed by Martello and Toth when calculating penalties for violating constraints (3). Wilcox defines a feasibility measure $f(x) = \Sigma_j |1-\Sigma_i x_{ij}|$. First every job is inspected for a potential bound improvement, and failing that, jobs are examined for their potential to decrease f(x) by the corresponding adjustment of x. This adjustment can lead to finding a feasible (and hence optimal) solution.

Wilcox compares the binary branching rule of Ross and Soland to the multiple branching rule used by Martello and Toth and concludes that the latter yields better results since it exploits the special structure of the multiple-choice constraints (3). He als tries to reduce the dimensions of the problem by fixing variables to zero and one. In fact, the branching rule is used in conjunction with the variable fixing procedure in that the job to branch on is the one with the largest number of fixed variables.
A three-phase heuristic is used to generate a feasible solution at the top of the search tree. Phase 1 generates a feasible solution which is searched for local improvements in phase 2. In phase 3 Balas and Martin's pivot and complement heuristic (1980) is applied to further improve upon the solution. The three-phase heuristic generates solutions which deviate from the optimum by 1% on average.

Wilcox compares his approach to the one of Fisher, Jaikumar and Van Wassenhove (1986) and concludes that his algorithm is faster (factor 2 to 10 and more for the more difficult problems) and generates smaller trees (factor 10).

Jörnsten and Värbrand (1987) present two algorithms for the generalized assignment problem. The bounds are generated through the solution of Lagrangean and surrogate relaxations respectively with the addition and subsequent relaxation of valid inequalities. Only the first method is elaborated here, the second algorithm will be discussed in the next section.

To strengthen the bound of the Lagrangean relaxation valid inequalities of the same form as the knapsack constraints are added :

$$(\sum_i \sum_j b_{kij} \, x_{ij} \leq d_k \;,\; k \in K) \qquad (17)$$

Balas (1975) discusses how such inequalities can be generated. The algorithm relaxes both the knapsack constraints and the valid inequalities. The formulation of the Lagrangean dual problem is as follows :

$$\max L(u,\delta) = \min \{\sum_i \sum_j c_{ij} x_{ij} + \sum_i u_i (\sum_j a_{ij} x_{ij} - b_i) + \sum_k \delta_k (\sum_i \sum_j b_{kij} x_{ij} - d_k)\} \qquad (18)$$

$$\text{s.t.} \sum_i x_{ij} = 1 \qquad j \in J \qquad (19)$$

$$x_{ij} = 0 \text{ or } 1 \qquad i \in I,\ j \in J \qquad (20)$$

The solution procedure is as follows. Choose initial multipliers and solve the problem using the subgradient method. If the duality gap is closed, then stop. If not, generate a valid inequality, relax it and add it to the extended Lagrangean. Then, go back and solve the subproblem again. If the algorithm terminates without closing the duality gap because it is no longer possible to generate a valid inequality, a branch and bound approach can be used to carry on the solution procedure. The bound obtained in this way is stronger than the one obtained by simply relaxing constraints (2) in the original GAP formulation. The authors give no comparison with other solution techniques except for their own surrogate relaxation procedure. The test problems contain 4 agents and 25 jobs.
Another approach by Jörnsten and Näsberg (1986) based on Lagrangean decomposition is discussed in section 6.

5. SURROGATE RELAXATION

Jörnsten and Näsberg (1986) discuss very briefly a surrogate relaxation of constraints (2) in the original formulation. This yields a multiple choice knapsack problem :

$$(SR) \min \{ \sum_i \sum_j c_{ij} x_{ij} \mid \sum_i \sum_j \mu_i a_{ij} x_{ij} \leq \sum_i \mu_i b_i, (3), (4)\} \qquad (21)$$

The bounds obtained by this relaxation are at least as good as those obtained by the LP-relaxation of the original problem (integrality property) and those resulting from the Lagrangean relaxation of constraints (2). In practice the bound is slightly inferior to the bound obtained by a Lagrangean relaxation of constraints (3).

The second procedure of Jörnsten and Värbrand (1987) generates bounds through the solution of a surrogate relaxation of the knapsack constraints (2) and valid inequalities of the knapsack type (17). The formulation is :

$$\max S\,(u,\delta) = \min \sum_i \sum_j c_{ij}\, x_{ij} \tag{22}$$

$$\text{s.t.} \sum_i u_i \sum_j x_{ij}\, a_{ij} + \sum_k \delta_k \sum_i \sum_j b_{kij}\, x_{ij} \le \sum_i u_i\, b_i + \sum_k \delta_k\, d_k \tag{23}$$

(3) and (4).

For the generation of the valid inequalities as well as the computation of the surrogate multipliers we refer to the original paper (the procedure is analogous to the one mentioned in section 4). The Lagrangean procedure of Jörnsten and Värbrand discussed in the previous section appears to be more efficient (stronger bound at root node, less valid inequalities added) than the surrogate one. There is no comparison with other solution techniques.

6. LAGRANGEAN DECOMPOSITION

Another approach is due to Jörnsten and Näsberg (1986). They reformulate the problem as :

$$\min \quad \alpha \sum_i \sum_j c_{ij} x_{ij} + \beta \sum_i \sum_j c_{ij} y_{ij} \tag{24}$$

$$\text{s.t.} \quad \sum_j a_{ij} x_{ij} \le b_i \qquad i \in I \tag{25}$$

$$\sum_i y_{ij} = 1 \qquad j \in J \tag{26}$$

$$x_{ij} = y_{ij} \qquad i \in I,\ j \in J \tag{27}$$

$$x_{ij},\ y_{ij} = 0 \text{ or } 1 \qquad i \in I,\ j \in J \tag{28}$$

Note that $\alpha+\beta=1$. Using a Lagrangean relaxation on constraints (27) the problem decomposes into binary knapsack problems in the x variables and semi assignment problems or GUBs in the y variables. The subgradient method is used to compute multipliers. The two problem formulations are as follows :

$$PX(\mu) \quad \min \quad \sum_i \sum_j (\alpha\, c_{ij} - \mu_{ij})\, x_{ij} \tag{29}$$

$$\text{s.t.} \quad \sum_j a_{ij} x_{ij} \le b_i \qquad i \in I \tag{30}$$

$$x_{ij} \in \{0,1\} \qquad i \in I,\ j \in J \tag{31}$$

$$PY(\mu) \quad \min \quad \sum_i \sum_j (\beta\, c_{ij} + \mu_{ij})\, y_{ij} \tag{32}$$

$$\text{s.t.} \quad \sum_i y_{ij} = 1 \qquad j \in J \tag{33}$$

$$y_{ij} \in \{0,1\} \qquad i \in I,\ j \in J \tag{34}$$

The proposed bound is compared with traditional Lagrangean relaxations and it is concluded (based on a rather small set of 10 test problems with 4 agents and 25 jobs) that it is stronger than the one obtained by relaxing either constraints (3) or (2) in the original problem formulation (GAP). The surrogate relaxation has not been considered in conjunction with the Lagrangean decomposition approach. No enumeration scheme is given to solve the problem to optimality.

In addition to the relaxation scheme, feasible solutions (upper bounds) are generated starting from assignments satisfying either the x variable problem (PX(μ)) or the y variable problem (PY(μ)). Then an interchange procedure is started to obtain feasibility for the whole problem. Once this has been attained an improvement step is added. Starting from the y variable problem appears to give the best results for the problem sizes considered (4 agents and 25 jobs).

7. SUMMARY

From the discussion in the previous sections it can be concluded that an effective algorithm should have at least the following four well-balanced ingredients : a primal heuristic, a bounding scheme, a variable-fixing procedure and a branching rule. Wilcox (1989) was very careful in selecting these four ingredients by building on experience gathered in previous research. His experiments emphasize the importance of each ingredient and show that the four components put together provide the most effective algorithm to-date.

It is not a trivial task to compare the algorithms discussed in the preceding sections on a computational basis since different authors have used different computers, operating systems, programming languages and compilers. They have also used differently generated data sets or random number generators. Therefore, a standard data set which could be made available to all researchers (e.g. on floppy disk) would be very useful indeed. Such a data set would have to contain problems with varying degree of difficulty (tightness of capacity constraints, correlation between data, etc.) and of different size and shape (number of jobs and agents and the ratio between the two).

In spite of the lack of directly comparable computational results, we try to give some indication on the relative performance of the different algorithms in what follows. First, consider the Ross-Soland (RS), Martello-Toth (MT), Fisher-Jaikumar-Van Wassenhove (FJV), Guignard-Rosenwein (GR) and Wilcox (W) algorithms. Theoretically, the bounds relate to each other as : W, GR $\geq$ FJV $\geq$ RS, MT, LP, where LP denotes the bound obtained through LP-relaxation. In practice both GR and W improve upon FJV which in turn dominates RS and MT. In terms of CPU-time and number of nodes required, MT dominates RS, at least for the harder problems. Algorithm W claims to be 10 times faster than FJV whereas GR claims to be 1.5 times faster than FJV. In turn, FJV claims to require an order of magnitude less CPU-time than RS and MT. When comparing number of nodes W claims to require 10 times less than FJV, GR claims 1.5 less and FJV claims two orders of magnitude less nodes than RS and MT. All these results are of course only rough approximations because of the largely different experimental conditions under which these claims were made. Note however that the results mentioned above are at least all based on problem data generated by an identical procedure albeit perhaps with different random generators. Problem sizes used by the authors in their original papers varied from 5*20 (MT,FJV) to 5*40 (W) and 10*50 (GR).
Jörnsten and Näsberg (JN) and Jörnsten and Värbrand (JV) use 4*25 problems to test the quality of their bounding procedures. They do, however, fail to specify how the problems were generated. It is shown that their bounds dominate the ones obtained by FJV and LP both in theory and in practice. Nothing is said on CPU-times required to compute the bounds or on their effectiveness in a branch and bound procedure.

8. EXTENSIONS OF THE GAP PROBLEM

Mazzola (1989) introduces an important generalization which allows for capacity interaction among tasks assigned to the same agent : the 0-1 generalized assignment problem with nonlinear capacity constraints (NLGAP). The formulation is as follows :

$$\text{(NLGAP)}\quad \min \sum_i \sum_j c_{ij}\, x_{ij} \tag{35}$$

$$\text{s.t.}\quad \sum_j a_{ij}\, x_{ij} + \sum_h k_{ih}\, (\prod_l x_{il}) \le b_i \qquad i \in I \tag{36}$$

$$\sum_i x_{ij} = 1 \qquad j \in J \tag{37}$$

$$x_{ij} = 0 \text{ or } 1 \qquad j \in J,\ i \in I \tag{38}$$

All coefficients are as defined above, except k_{ih}. This coefficient models the capacity interaction between the variables x_{il} and can be positive, negative or zero.

Mazzola applies the results for general nonlinear 0-1 programming in Balas and Mazzola (1984a) and (1984b) to replace the nonlinear constraints by linear ones (the resulting linear 0-1 program is a GAP and a valid relaxation of NLGAP). This now linear expression lf_i satisfies the following condition :

$$lf_i < \sum_j a_{ij}\, x_{ij} + \sum_h k_{ih}\, (\prod_l x_{il}) \le b_i \qquad i \in I \tag{39}$$

Now the common body of knowledge for solving GAP can be used. At every node in the tree the corresponding subproblem has the structure of a NLGAP. One can generate the GAP relaxation at every node (which is time consuming but yields a stronger bound) or one can generate the relaxation at the top node only and use it throughout the search tree. The additional effort required to generate the relaxations at each node is more than compensated for by the decrease in overall solution effort and in the corresponding decrease in the number of nodes required. Dimensions of the problems discussed are 5 agents, 20 jobs with up to 1000 (=h) 2- and 3-way (=l) interactions.

Mazzola also describes a heuristic that uses a two-phase solution approach in the spirit of Martello and Toth, i.e. construct a feasible solution and then improve upon that solution by the use of interchanges and shifts of jobs between agents.

Gavish and Pirkul (1985c) discuss the multiconstraint GAP. The formulation is as follows :

$$\text{(MCGAP)} \quad \min \sum_i \sum_j c_{ij} x_{ij} \qquad (40)$$

$$\text{s.t.} \sum_i x_{ij} = 1 \qquad j \in J \qquad (41)$$

$$\sum_j a_{ijk} x_{ij} \leq b_{ik} \qquad k \in K,\ i \in J \qquad (42)$$

$$x_{ij} \in \{0,1\} \qquad i \in I,\ j \in J \qquad (43)$$

where k indicates the type of resource used.

The authors discuss several relaxations of the problem. Relaxation of the constraint set (42) yields GUBs (RL1); relaxation of the constraint set (41) and all elements of set (42) except those where k=k* results in single constraint binary knapsack problems (RL2); relaxation of constraint set (41) leads to multiconstraint binary knapsack subproblems, Gavish and Pirkul (1986d) (RL3) :

$$RL1 = \min \{ \sum_i \sum_j c_{ij} x_{ij} + \sum_i \sum_k \delta_{ik} (\sum_j a_{ijk} x_{ij} - b_{ik}) \mid (41),(43) \} \qquad (44)$$

$$RL2 = \min \{ \sum_i \sum_j c_{ij} x_{ij} + \sum_j \mu_j (\sum_i x_{ij} - 1) + \sum_{i,k \neq k'} \delta_{ik} (\sum_j a_{ijk} x_{ij} - b_{ik}) \mid (43), \sum_j a_{ijk'} x_{ij} \leq b_{ik'} \} \qquad (45)$$

$$RL3 = \min \{ \sum_i \sum_j c_{ij} x_{ij} + \sum_j \mu_j (\sum_i x_{ij} -1) \mid (42), (43) \} \qquad (46)$$

Comparing the different bounding schemes the third method appears to be the stronger one while the second relaxation nearly always yields stronger bounds than the first one.
The authors also discuss several heuristics. The first heuristic is based on a regret basis (first assign the task that would yield a high

incremental cost if it were not assigned to the least costly agent). A second heuristic uses the solution of the first relaxation. All jobs have been assigned but some agents are overloaded, so the heuristic tries to reallocate some of their load to agents with excess capacity. The third heuristic is used in conjunction with the third relaxation. Some jobs are assigned twice or more, some are not assigned. A new problem which has the same structure as the original one can be created by taking those tasks which are unassigned or are assigned to multiple agents. The rest of the jobs are permanently assigned to their agents as determined by the third relaxation. The third heuristic generally generates better solutions than the first two (the first and second show a similar performance).

The branch and bound algorithm suggested uses the third relaxation (this one yields tighter bounds). Also the third heuristic strategy based on that relaxation determines feasible solutions which are found to be optimal in many cases. The separation is accomplished by taking a task and assigning it to every agent. The order in which tasks and agents are considered is based on the penalties derived by setting variables x_{ij} equal to $1 - x_{ij}{}^*$ (with $x_{ij}{}^*$ the solution of the third relaxation). If the penalty is larger than the gap between the lower bound and the best primal solution, then in any optimal solution x_{ij} must be equal to $x^*{}_{ij}$. Computational results are given for problems with up to 10 agents, 100 jobs and 2 types of resources.
Note that in this approach the four ingredients of a good algorithm can clearly be distinguished.

9. CONCLUSIONS

All different approaches discussed here are based on branch and bound techniques and on relaxation of the primal formulation. The few comparative results for optimal methods reported in literature are for rather small problmes. Nothing is said on the importance of the branching strategy (except by Wilcox (1989)). There are very few heuristics (Klastorin, Martello and Toth, Benders and van Nunen, Jörnsten and Näsberg) and, to the best of our knowledge, no comparison was ever made between them.
There seem to be no methods based on dual heuristics or generalized

linear programming. Looking at the dual of the LP-relaxation, it appears to be possible to obtain good dual solutions that can yield both a lower and an upper bound (via a primal heuristic). Such an approach is similar to the one suggested by Erlenkotter (1978) for the uncapacitated facility location problem. The authors are currently exploring this avenue.
A heuristic based on a set partitioning approach has already shown to give good results. This heuristic can be extended in order to give optimal solutions using a branch and bound procedure linked with the column generation procedure (Ribeiro, Minoux and Penna (1989)).

References

Balachandran, V., An integer generalized transportation model for optimal job assignment in computer networks, Working Paper 34-72-3, (1972) Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh.

Balas, E, Facets of the knapsack polytope, Math Prog., 8 (1975), p. 146-164.

Balas, E., Martin, E., Pivot and Complement - A heuristic for 0-1 programming, Management Science, 26.1 (1980), p. 86-96.

Balas, E. Mazzola, J.B., Nonlinear 0-1 programming : I linearization techniques, Math. Prog., 30 (1984a), p. 1-24.

Balas, E., Mazzola, J.B., Nonlinear 0-1 programming : II dominance relations and algorithms, Math. Prog., 30 (1984b), p. 22-45.

Benders, J.F., van Nunen, J.A.E.E., A property of assignment type mixed integer linear programming problems, Operations Research Letters, 2.2 (1983), p. 47-52.

Erlenkotter, D, A dual-based procedure for uncapacitated facility location, Opns. Res., 26.6 (1978), p. 992-1009.

Fisher, M.L., The Lagrangean relaxation method for solving integer programming problems, Management Science, 27.1 (1981), p. 1-18.

Fisher, M.L, An applications oriented guide to Lagrangean Relaxation Interfaces, 15.2 (1985), p. 10-21.

Fisher, M.L., Jaikumar, R., A generalized assignment heuristic for vehicle routing, Networks, 11 (1981), p. 109-124.

Fisher, M.L., Jaikumar, R., Van Wassenhove, L., A multiplier adjustment method for the generalized assignment problem, Management Science, 32.9 (1986), p. 1095-1103.

Gavish, B., Pirkul, H., Zero-one integer programs with few constraints - efficient branch and bound algorithms, EJOR, 22 (1985a), p. 35-43.

Gavish, B., Pirkul, H., Zero-one integer programs with few constraints - lower bounding theory, EJOR, 21 (1985b), p. 213-224.

Gavish, B., Pirkul, H., Efficient algorithms for the multiconstraint generalized assignment problem, Working paper, QM 8523, (1985c), University of Rochester.

Gavish, B., Pirkul, H., Efficient algorithms for solving multiconstraint zero-one knapsack problems to optimality, Math. Prog., 31.1 (1985d), p. 78-105.

Geoffrion, A.M., Lagrangean relaxation and its uses in integer programming, Mathematical Programming Study 2 (1974), p. 82-114.

Grigoriadis, M.D., Tang, D.J. and Woo, L.S., Considerations in the optimal synthesis of some communications networks, Presented at the 45th Joint National Meeting of ORSA/TIMS, Boston, MA, 1974.

Guignard, M., Rosenwein, M., An improved dual-based algorithm for the generalized assignment problem, Opns. Res., 37.4 (1989), p. 658-663.

Jörnsten, K., Näsberg, M., A new Lagrangean relaxation approach to the generalized assignment problem, EJOR, 27 (1986), p. 313-323.

Jörnsten, K.O., Värbrand, P., Two algorithms for the generalized assignment problem, Working paper, LiTH-Mat-R-87-02, (1987), Linköping Institute of Technology, Sweden.

Klastorin, T.D., An effective subgradient algorithm for the generalized assignment problem, Computers and Operations Research, 6 (1979), p. 155-164.

Martello, S., Toth, P., An algorithm for the generalized assignment problem, in J.P. Brans (ed.), Operational Research '81, North Holland Publishing Company, Amsterdam, 1981, p. 589-603.

Mazzola, J.B., Generalized assignment with nonlinear capacity interaction, Management Science, 35.8 (1989), p. 923-941.

Ribeiro, C.C., Minoux, M., Penna, M.C., An optimal column-generation-with-ranking algorithm for very large scale set partitioning problems in traffic assignment, EJOR, 41 (1989), p. 232-239.

Ross, G.T., Soland, R.M., A branch and bound algorithm for the generalized assignment problem, Math. Prog., 8 (1975), p. 91-103.

Ross, G.T., Soland, R.M., Modeling facility location problems as generalized assignment problems, Management Science, 24.3 (1977), p. 345-357.

Wilcox, S.P., A new multiplier adjustment method for the generalized assignment problem, Working paper, (1989), General Research Corporation, 7655 Old Springhouse Road, Mc Lean, Virginia 22102.