EDUCAÇÃO e TECNOLOGIA

Revista do Instituto Politécnico da Guarda

EDUCAÇÃO E TECNOLOGIA

Propriedade
Instituto Politécnico da Guarda

Director
João Bento Raimundo

Redacção
Serviços Centrais do I.P.G. - Av. Francisco Sá Carneiro nº 50
6300 Guarda
Telef. 222634 * Telecópia 222690

Composição
Gabinete Editorial do I.P.G.

Execução Gráfica e Impressão
Secção de Reprografia do I.P.G.

Periodicidade
Semestral

Tiragem
1.000 ex.

Depósito Legal
nº 17.981/87

nº XIII - Fevereiro de 1994

Foto da Capa: Vista parcial do edifício
dos Serviços Centrais do IPG, em dia de neve.

UM NOVO CICLO

A edição deste número, o décimo terceiro, da Revista, "Educação e Tecnologia" coincide com o alvorecer de um novo ciclo da vida do Instituto Politécnico da Guarda.

"*O homem e a hora são um sól quando Deus faz e a história é feita*", como escreveu o poeta. E nós não esquecemos os vectores da conjuntura em que nasceu este projecto, hoje concretizado nas suas principais e visíveis vertentes. Assumimos os desafios, não pactuámos com o tempo e com a burocracia, afirmámos uma postura e uma dinâmica próprias, alimentadas no empenho em implementar o ensino superior politécnico nesta zona.

Uma instituição de ensino superior vale, desde logo, pela capacidade de resposta às reais necessidades da juventude, da região e do País, bem como pela sua credibilidade científica e pedagógica, pela qualidade e rigor dos cursos que ministra. E para certificarmos que a nossa ideia estava, desde o início, correcta, bastaria para tanto atentarmos nas percentagens das candidaturas aos cursos aqui leccionados e outrossim na nossa actual realidade.

Evidentemente que isto, para além das instalações necessárias, passou, também, pela disponibilização de um bom corpo docente, estável, participativo neste projecto colectivo, e igualmente pela existência de um corpo técnico e administrativo eficaz, agente interventor assente nas potencialidades das tecnologias postas à sua disposição; passou, igualmente, pela contínua reafirmação da qualidade e pela afirmação de um espírito de escola.

Para se ter percorrido esta caminhada foi preciso ser "*Claro em pensar, e claro no sentir, / É claro no querer;*" como bem disse Pessoa.

Hoje o Politécnico da Guarda é uma referência em termos nacionais, com o seu projecto de Estatutos já concluído e prestes a deixar o regime de instalação. Abre-se, deste modo, um capítulo novo na vida deste estabelecimento de ensino superior que continuará a afirmar a sua divisa "*scientia lucet omnibus*".

João Raimundo
Presidente da Comissão Instaladora
do Instituto Politécnico da Guarda

RECENT DEVELOPMENTS IN ALGOTITHMS FOR THE CAPACITATED WAREHOUSE LOCATION PROBLEM

Barrie M. Baker,*
Amândio Pereira Baia**

ABSTRACT

This paper describes the results of o pilot study for a recently encountered Regional Water Authority Problem (RWAP). The RWAP differs from the well known Capacited Warehouse Location Problem (CWLP) only in that there are both weeks of normal demand levels and weeks of peak demand levels. This pilot study considers some possible methods of refining a well-known and straightforward approach to the CWLP using Lagrangean relaxation. Computational results are given for bench-mark problems.

INTRODUCTION

The Capacited Warehouse Location Problem (CWLP) consists of m potencial warehouse sites and n customers whose demands must be met in full. There is a fixed cost, f_i, of establishing warehouse i and, if the warehouse is established, a

* School of Mathematical Sciences, Coventry University, England
** Professor Adjunto na E.S.T.G.

supply of s_i units of a commodity will be available at the warehouse. Customer j has a demand for d_j units of this commodity, and each unit sent from warehouse i to the customer j incurs a cost of c_{ij}.

The CWLP can be formulated as

$$P: \quad \text{minimise} \sum_{i=1}^{m} f_i y_i + \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij} , \tag{1}$$

subject to

$$\sum_{i=1}^{m} x_{ij} = d_j \qquad \text{for } j=1,\ldots,n, \tag{2}$$

$$\sum_{j=1}^{n} x_{ij} \le s_i y_i \qquad \text{for } i=1,\ldots,m, \tag{3}$$

$$x_{ij} \le d_j y_i \qquad \text{for all } i,j, \tag{4}$$

$$x_{ij} \ge 0 \qquad \text{for all } i,j, \tag{5}$$

$$y_i \in \{0,1\} \qquad \text{for all } i, \tag{6}$$

Where x_{ij} is the number of units sent from warehouse i to customer j, and

$$y_i = \begin{cases} 1 & \text{if warehouse } i \text{ is established,} \\ 0 & \text{otherwise.} \end{cases} \tag{7}$$

Constraints (4) can be omitted from the formulation, but they are generaly included as they are well known to give rise to a much tighter LP relaxation of P.

Given the set of warehouses to be established, the problem would reduce to an ordinary transportation problem. Without this knowledge, a general purpose integer programming package could solve small instances of the CWLP within a realistic amount of time.

To solve problems of practical size, special purpose algorithms have been developed for the CWLP by many researchers [1,2,3,4,5,6,7 for example].

Most of this use Lagrangean relaxation of either the demand constraints (2) or the supply constraints (3) to obtain lower bounds, for use in a branch and bound algorithm. In this paper, we describe some experiments carried out with the former of these two types of relaxation. One approach is to derive upper limits on the number of units of the commodity that could be used at each potential warehouse in any optimal solution. We also experiment with scaling of the constraints to be relaxed, and with storing binary strings corresponding to the most recently calculated upper bounds. Small improvements in CPU times were observed using these methods, as shown by the computational results given below.

BACKGROUND TO THE STUDY

The project arose with a request from original water authority in Britain. Their need to distribute water from reservoirs and bore holes to demand centres in the region had given rise to a problem that was almost identical to CWLP. The difference was that normal demand levels applied for 48 weeks of the year, with peak demand levels applying for the other 4 weeks. Thus the regional water authority problem (RWAP) is to:

$$\text{minimise} \quad \sum_{i=1}^{m} f_i y_i + \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij}(48x_{ij} + 4x'_{ij}), \tag{8}$$

subject to (2) - (6) and

$$\sum_{i=1}^{m} x'_{ij} = p_j d_j \qquad \text{for } j=1,\ldots,n, \tag{9}$$

$$\sum_{j=1}^{n} x'_{ij} \le s_i y_i \qquad \text{for } i=1,\ldots,m, \tag{10}$$

$$x'_{ij} \le p_j d_j y_i \qquad \text{for all } i,j, \tag{11}$$

$$x'_{ij} \ge 0 \qquad \text{for all } i,j, \tag{12}$$

where x_{ij} and y_i are as defined before, and x'_{ij} is the number of units sent from warehouse i to customer j in a week of peak demand. d_j is now defined as the demand level of customer j in a normal week and p_j is defined so that the demand of customer j in a peak week is $p_j d_j$. f_i is now defined as an annual fixed cost of maintaining source i.

The principal aim of our project was to consider the Lagrangean relaxations that have been applied to CWLP, and then to investigate and compare their adaptation to RWAP. Thus, a Lagrangean relaxation of the demand constraints (2), as reported by Geoffrion and McBride[2], Nauss[3], Christofides and Beasley[4], for example, could be adapted to a relaxation of (2) and (9). A Lagrangean relaxation of the supply constraints (3), as reported by Van Roy[6], could be paralleled by relaxing (3) and (10).

During this study, a programme was developed applying a Lagrangean relaxation of demand constraints to the CWLP. Novel features incorporated in the implementation gave good computational results. Details are given in the following sections.

LAGRANGEAN RELAXATION OF DEMAND CONSTRAINTS

For any given $\lambda_j \geq 0$, j = 1, ..., n, a Lagrangean relaxation of constraints (2) leads to

$$PR1_\lambda: \quad \text{minimise} \quad \sum_{i=1}^{m} f_i y_i + \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} + \sum_{j=1}^{n} \lambda_j \left(d_j - \sum_{i=1}^{m} x_{ij} \right) \tag{13}$$

$$= \sum_{i=1}^{m} f_i y_i + \sum_{i=1}^{m}\sum_{j=1}^{n} (c_{ij} - \lambda_j) x_{ij} + \sum_{j=1}^{n} \lambda_j d_j \ . \tag{14}$$

subject to (3), (5) and (6).

A stronger relaxation is obtained by appending an overall capacity constraint,

$$\sum_{i=1}^{m} s_i y_i \geq \sum_{j=1}^{n} d_j \ . \tag{15}$$

It is then necessary to solve m continuous knapsack problems and a 0-1 knapsack problem as follows. For i = 1, ..., m, let

$$z_i = f_i + \min_{0 \le x_{ij} \le d_j} \left\{ \sum_{j=1}^{n} (c_{ij} - \lambda_j) x_{ij} : \sum_{j=1}^{n} x_{ij} \le s_i \right\} . \quad (16)$$

Then solve

$$PR2_\lambda : \qquad minimise \sum_{i=1}^{m} z_i y_i + \sum_{j=1}^{n} \lambda_j d_j , \quad (17)$$

subject to (6) and (15).

Computational results for a branch and bound algorithm based on the lower bound of (17) are given by Nauss[3].

DERIVING A LIMIT ON USABLE CAPACITY

A possible weakness in calculating the lower bound (17) is that when the continuous knapsack problems (16) are solved to obtain $\{ z_i \}$, for some values of i the value $\sum_j x_{ij}$ could be much less than s_i. If the constraint

$$\sum_{j=1}^{n} x_{ij} \ge \hat{s}_i \quad (18)$$

is imposed on $PR2_\lambda$ for such a row i, where $< S_i$, then the continuous knapsack problem (16) can be solved exactly as before, but the possibility now arises of $x_{ij} > 0$ where $c_{ij} - l_j > 0$. A trivial modification of the 0-1 knapsack problem is required with warehouse i constraint open. If the resulting lower bound exceeds the current upper bound, then no solution satisfying the additional constraint (18) can give an improvement on the best solution previously found. Thus, the usable capacity at warehouse i can then be restricted to , and this in turn can be used to strengthen the overall capacity constraint (15). This can lead to an improved lower bound from (17), enabling further reductions in usable capacity.

Here, we develop an algorithm incorporating this approach within a routine for fixing warehouses. The routine considers each warehouse, i, in turn. If warehouse i was closed in the 0-1 knapsack solution, then it is temporarily constrained open, and the 0-1 knapsack problem is re-solved. This may enable the warehouse to be fixed closed immediately. Otherwise, it is a simple matter to determine whether there is a value such that imposing the constraint (18) closes the gap between the lower and upper bounds. If the usable capacity can be restricted in this way, then the current solution of the 0-1 knapsack problem with warehouse i constrained open can become infeasible, so that an improved lower bound and a further reduction in usable capacity are possible. This procedure can then be repeated, possibly leading to warehouse i being fixed closed.

Alternatively, if warehouse i is open in the initial 0-1 knapsack solution, then it is first temporarily constrained closed. This may lead to warehouse i being fixed open, with previous reductions in usable capacities of other warehouses being helpful towards this end, since (15) can be strengthened by using reduced capacities in place of the original values. If warehouse i cannot be fixed open, then the procedure for reducing usable capacity can be applied to warehouse i.

The above routine for fixing warehouses and reducing capacities was incorporated in a depth first branch and bound algorithm as detailed in Appendixe I.

ILLUSTRATIVE EXAMPLE

The method of restricting the usable capacity of a warehouse can be illustrated using the following example, taken from Baker[5].

Warehouse	f_i	Demand point 1	2	3	4	5	6	s_i
1	33	5	19	8	2	23	12	28
2	47	1	7	15	8	7	27	47
3	27	15	6	4	5	19	1	69
4	39	4	27	33	24	6	9	48
5	41	11	8	16	11	5	2	37
6	26	21	14	4	26	18	6	55
	d_j:	13	26	17	12	23	25	

Table 1

To apply the method of reducing usable capacities successfully, a near-optimal set of Lagrange multipliers is required. Normally, an algorithm would incorporate some heuristic and, possibly, subgradient optimisation, to obtain these multipliers. Here, for the purposes of illustration, we will take as our multiplier vector the set of dual variables corresponding to the optimal solution of a transportation problem as given by Baker[5]. Appendix II shows the referenced transportation problem, with the column duals derived as :

$$(4.62, 6.4, 4.4, 4.75, 6.78, 1.4).$$

These transportation duals should provide an optimal or near-optimal set of Lagrange multipliers. $(c_{ij} - l_j)$ is given by:

0.38	12.60	3.60	-2.75	16.22	10.60
-3.62	0.60	10.60	3.25	0.22	25.60
10.38	-0.40	-0.40	0.25	12.22	-0.40
-0.62	20.60	28.60	19.25	-0.78	7.60
6.38	1.60	11.60	6.25	-1.78	0.60
16.38	7.60	-0.40	21.25	11.22	4.60

Table 2

Table 2

Then, using (16), we obtain,

$$(Z_j) = (0.0, -0.06, -0.20, 13.0, 0.06, 19.20).$$

The 0-1 knapsack problem (17) is then trivial, and has an optimal solution given by :

$$Y_1 = Y_2 = Y_3 = 1; \quad Y_4 = Y_5 = Y_6 = 0.$$

A lower bound for the optimal cost is then given by:

$$\sum_{i=1}^{6} z_i y_i + \sum_{j=1}^{6} \lambda_j d_j$$

$$= -0.26 + 549.2 = 548.94.$$

Following common practice for Lagrangean heuristics, an upper bound is obtained by solving the transportation problem using only warehouses 1, 2 and 3. This has the following optimal solution, giving an upper bound of 554.0.

Demand point

Warehouse	1	2	3	4	5	6	Dummy
1	5	19	8	2 **12**	23	12	0 **16**
2	1 **13**	7	15	8	7 **23**	27	0 **11**
3	15	6 **26**	4 **17**	5	19	1 **25**	0 **1**

Table 3

From row 1 of Table 2, 12 units of capacity are used to obtain the value Z_1. Each further unit forced into use increases the lower bound by 0.38 for the next 13 units, and then by 3.6 for the remaining 3 units. Thus, $\hat{s}_1 = 25$, since the 26th unit at 3.6 causes the upper bound exceeded:

$$548.94 + 0.38 \times 13 + 3.6 = 557.48 > 554.0 .$$

Similarly, $\hat{s}_2 = 36$ and $\hat{s}_5 = 31$.

Warehouses 4 and 6 can be fixed closed, and warehouse 3 fixed open with no reduction in usable capacities.

In this case, the above solution of the 0-1 knapsack problem (17) remains feasible. However, an alternative optimal solution with $y_1 = 0$ becomes infeasible due to the reduction in usable capacity of warehouse 2.

THE BRANCH AND BOUND ALGORITHM

A detailed statement of the algorithm developed is given in Appendix I. This is a depth first algorithm. The Lagrange multipliers are initialised as $l_j = \min_i\{c_{ij} + f_i/s_i\}$. Subgradient optimisation is then used for finer adjustments of the Lagrange multipliers. Upper bounds are obtained by solving transportation problems using the warehouses for which $z_i = 1$ in the solution of PR2]. As each transportation problem is solved, the binary string corresponding to the sequence of y_i values is stored. The ten most recently calculated strings are checked at each interation to avoid the same transportation problems being solved unnecessarily

SUBGRADIENT OPTIMISATION

For any given set of Lagrange multipliers, and the corresponding optimal solution of the Lagrangean relaxation, the procedure is to calculate the error for each relaxed constraint:

$$N_j = d_j - \sum_{i=1}^{m} x_{ij} \qquad (20)$$

We could update the Lagrange multipliers using

$$\lambda_j^{k+1} = \max\{0, \lambda_j^k + t_k N_j\} \qquad (21)$$

where

$$t_k = c(Z_{UB} - Z_{LB}) \; / \; \sum_{j=1}^{n} N_j^2 \qquad (22)$$

and Z_{UB} and Z_{LB} are, respectively, the best known upper bound and the current lower bound from the Langrangean relaxation.

However, in problems where the values d_j vary widely, a very large value may dominate the calculation of t_k. This can be overcome by scaling. Define

$$X_{ij} = x_{ij}/d_j \qquad \text{for all } i,j, \qquad (23)$$

and

$$C_{ij} = d_j c_{ij} \qquad \text{for all } i,j. \qquad (24)$$

Then the constraints to be relaxed are

$$\sum_{i=1}^{m} X_{ij} = 1 \qquad \text{for all } j, \tag{25}$$

and the Langrangean objective function (14) is re-written as,

$$\sum_{i=1}^{m} f_i y_i + \sum_{i=1}^{m} \sum_{j=1}^{n} (C_{ij} - \lambda_j) X_{ij} + \sum_{j=1}^{n} \lambda_j \ . \tag{26}$$

For this form, the errors, N_j, are divided by d_j before calculating each adjustment $t_K N_j$. These adjustments are appropriate for the scaled up objective coefficients, c_{ij}, and must be divided by d_j for use with the smaller values,c_{ij}.

In our implementation, c is initialised at 2 and is halved whenever the lower bound shows no significant increase in 10 successive iterations. These rules were chosen as a result of tests carried out on the problems described below.

COMPUTATIONAL RESULTS

A set of bench-mark problems for the CWLP was obtained by electronic mail from the OR-Library at Imperial College[8]. Tables 4 to 7 show examples of the results obtained, using a CCI-PC 486 DX (50 Mhz).

Overall, the method of reducing capacities is not beneficial. For the 37 test problems, 9 cases gave improved CPU times and 3 cases were unchanged when this method was used.The remaining problems had increased CPU times.

Storing a short list of binary strings, to record which warehouses where used in the most recent 0-1 knapsack solutions, is useful. Overall, the best results were obtained when the number of strings stored was 10.

The method of scaling the demand constraints was found to be beneficial for data sets where d_j values varied widely.

CONCLUSION

The experiments reported here were intended primarily as a pilot study for the Regional Water Authority Problem (RWAP). The sizes of RWAP that we have encountered in practice have been

mostly of the order of 25 x 25. The experiments carried out in this study of the CWLP suggest that practical sized RWAPs could be solved using a PC, to optimality, by adapting the very straightforward methods used here.

PROBLEMA	Size (m X n)	Root Node			Nodes in Tree	CPU* (Seconds)
		GAP %	Interations (Number)	Fixed Warehouse		
cap41	16 x 50	0.009	548	16	3	6.65
cap42	16 x 50	0.102	567	13	4	7.52
cap43	16 x 50	0.093	610	12	6	8.07
cap44	16 x 50	0.169	573	11	8	7.63
cap51	16 x 50	0.050	278	11	10	3.57
cap61	16 x 50	0.000	88	0	0	1.04
cap62	16 x 50	0.000	68	0	0	0.98
cap63	16 x 50	0.146	318	2	30	4.45
cap64	16 x 50	0.001	350	14	4	4.17
cap71	16 x 50	0.000	67	0	0	0.93
cap72	16 x 50	0.000	69	0	0	0.98
cap73	16 x 50	0.000	85	0	0	1.16
cap74	16 x 50	0.000	72	0	0	1.05
cap81	25 x 50	0.180	398	13	72	11.53
cap82	25 x 50	0.063	438	17	20	8.95
cap83	25 x 50	0.061	479	17	18	10.00
cap84	25 x 50	0.191	506	8	96	21.37
cap91	25 x 50	0.000	77	0	0	1.53
cap92	25 x 50	0.078	272	14	20	5.05
cap93	25 x 50	0.202	290	16	30	6.26
cap94	25 x 50	0.406	330	12	140	12.96
capl01	25 x 50	0.000	75	0	0	1.53
capl02	25 x 50	0.000	126	0	0	2.25
capl03	25 x 50	0.000	298	0	0	4.67
capl04	25 x 50	0.000	94	0	0	1.60
capl11	50 x 50	0.040	367	44	12	12.03
capl12	50 x 50	0.137	580	36	112	26.36
capl13	50 x 50	0.249	494	35	402	56.47
capl14	50 x 50	0.270	545	23	464	71.57
capl21	50 x 50	0.000	221	0	0	6.43
capl22	50 x 50	0.079	530	38	32	16.32
capl23	50 x 50	0.109	401	38	134	19.61
capl24	50 x 50	0.427	421	26	680	61.13
capl31	50 x 50	0.000	198	0	0	5.71
capl32	50 x 50	0.000	412	0	0	11.43
capl33	50 x 50	0.000	287	0	0	8.29
capl34	50 x 50	0.000	200	0	0	6.32

Table 4

Results with scaling of demand constraints and 10 binary strings stored; no reduction of warehouse capacities.
* CPU times for CCI-PC 486DX (50Mhz).

PROBLEMA	Size (m X n)	Root Node			Nodes in Tree	CPU* (Seconds)
		GAP* %	Interations (Number)	Fixed Warehouse		
cap41	16 x 50	0.009	548	16	3	6.81
cap42	16 x 50	0.102	567	15	2	7.47
cap43	16 x 50	0.093	610	14	2	7.80
cap44	16 x 50	0.169	573	14	2	7.31
cap51	16 x 50	0.050	278	11	10	3.68
cap61	16 x 50	0.000	88	0	0	1.09
cap62	16 x 50	0.000	68	0	0	0.98
cap63	16 x 50	0.146	318	2	30	4.61
cap64	16 x 50	0.001	350	14	2	4.23
cap71	16 x 50	0.000	67	0	0	0.93
cap72	16 x 50	0.000	69	0	0	0.99
cap73	16 x 50	0.000	85	0	0	1.15
cap74	16 x 50	0.000	72	0	0	1.05
cap81	25 x 50	0.180	398	13	68	11.92
cap82	25 x 50	0.063	438	17	20	9.28
cap83	25 x 50	0.061	479	17	16	10.27
cap84	25 x 50	0.191	506	8	60	16.75
cap91	25 x 50	0.000	77	0	0	1.59
cap92	25 x 50	0.078	272	14	20	5.38
cap93	25 x 50	0.202	290	16	30	6.43
cap94	25 x 50	0.406	330	12	128	13.45
cap101	25 x 50	0.000	75	0	0	1.59
cap102	25 x 50	0.000	126	0	0	2.26
cap103	25 x 50	0.000	298	0	0	4.73
cap104	25 x 50	0.000	94	0	0	1.70
cap111	50 x 50	0.040	367	44	12	12.41
cap112	50 x 50	0.137	580	36	100	26.59
cap113	50 x 50	0.249	494	35	304	48.00
cap114	50 x 50	0.270	545	23	298	56.96
cap121	50 x 50	0.000	221	0	0	6.48
cap122	50 x 50	0.079	530	38	38	16.97
cap123	50 x 50	0.109	401	38	74	17.19
cap124	50 x 50	0.427	421	26	602	60.52
cap131	50 x 50	0.000	198	0	0	5.82
cap132	50 x 50	0.000	412	0	0	11.54
cap133	50 x 50	0.000	287	0	0	8.46
cap134	50 x 50	0.000	200	0	0	6.64

Table 5
Results with reduction of warehouse capacities, scaling of demand constraints and 10 binary strings stored.

PROBLEMA	Size (m X n)	Root Node			Nodes in Tree	CPU* (Seconds)
		GAP %	Interations (Number)	Fixed Warehouse		
cap41	16 x 50	0.009	548	16	3	7.80
cap42	16 x 50	0.102	567	15	2	8.52
cap43	16 x 50	0.093	610	14	2	9.07
cap44	16 x 50	0.169	573	14	2	8.57
cap51	16 x 50	0.050	278	11	10	3.90
cap61	16 x 50	0.000	88	0	0	1.10
cap62	16 x 50	0.000	68	0	0	0.99
cap63	16 x 50	0.146	318	2	30	5.22
cap64	16 x 50	0.001	350	14	2	4.45
cap71	16 x 50	0.000	67	0	0	0.93
cap72	16 x 50	0.000	69	0	0	0.99
cap73	16 x 50	0.000	85	0	0	1.21
cap74	16 x 50	0.000	72	0	0	1.04
cap81	25 x 50	0.180	398	13	68	14.99
cap82	25 x 50	0.063	438	17	20	10.54
cap83	25 x 50	0.061	479	17	16	11.43
cap84	25 x 50	0.191	506	8	60	23.24
cap91	25 x 50	0.000	77	0	0	1.59
cap92	25 x 50	0.078	272	14	20	5.50
cap93	25 x 50	0.202	290	16	30	6.81
cap94	25 x 50	0.406	330	12	128	15.66
cap101	25 x 50	0.000	75	0	0	1.54
cap102	25 x 50	0.000	126	0	0	2.25
cap103	25 x 50	0.000	298	0	0	4.72
cap104	25 x 50	0.000	94	0	0	1.71
cap111	50 x 50	0.040	367	44	12	12.52
cap112	50 x 50	0.137	580	36	100	29.44
cap113	50 x 50	0.249	494	35	304	65.80
cap114	50 x 50	0.270	545	23	298	84.53
cap121	50 x 50	0.000	221	0	0	6.48
cap122	50 x 50	0.079	530	38	38	17.24
cap123	50 x 50	0.109	401	38	74	17.85
cap124	50 x 50	0.427	421	26	602	69.48
cap131	50 x 50	0.000	198	0	0	5.77
cap132	50 x 50	0.000	412	0	0	11.48
cap133	50 x 50	0.000	287	0	0	8.41
cap134	50 x 50	0.000	200	0	0	6.65

Table 6

Results with reduction of warehouse capacities and scaling of demand constraints; no storing of binary strings.

PROBLEMA	Size (m X n)	Root Node			Nodes in Tree	CPU* (Seconds)
		GAP* %	Interations (Number)	Fixed Warehouse		
cap41	16 x 50	0.009	548	16	3	7.69
cap42	16 x 50	0.102	567	13	4	9.01
cap43	16 x 50	0.093	610	12	6	9.67
cap44	16 x 50	0.169	573	11	8	9.50
cap51	16 x 50	0.050	278	11	10	3.74
cap61	16 x 50	0.000	88	0	0	1.10
cap62	16 x 50	0.000	68	0	0	0.99
cap63	16 x 50	0.146	318	2	30	4.78
cap64	16 x 50	0.001	350	14	4	4.40
cap71	16 x 50	0.000	67	0	0	0.94
cap72	16 x 50	0.000	69	0	0	0.93
cap73	16 x 50	0.000	85	0	0	1.15
cap74	16 x 50	0.000	72	0	0	1.04
cap81	25 x 50	0.180	398	13	72	13.24
cap82	25 x 50	0.063	438	17	20	9.67
cap83	25 x 50	0.061	479	17	18	11.04
cap84	25 x 50	0.191	506	8	96	27.41
cap91	25 x 50	0.000	77	0	0	1.54
cap92	25 x 50	0.078	272	14	20	5.16
cap93	25 x 50	0.202	290	16	30	6.53
cap94	25 x 50	0.406	330	12	140	14.34
cap101	25 x 50	0.000	75	0	0	1.54
cap102	25 x 50	0.000	126	0	0	2.25
cap103	25 x 50	0.000	298	0	0	4.73
cap104	25 x 50	0.000	94	0	0	1.65
cap111	50 x 50	0.040	367	44	12	12.19
cap112	50 x 50	0.137	580	36	112	28.45
cap113	50 x 50	0.249	494	35	402	68.77
cap114	50 x 50	0.270	545	23	464	100.24
cap121	50 x 50	0.000	221	0	0	6.43
cap122	50 x 50	0.079	530	38	32	16.47
cap123	50 x 50	0.109	401	38	134	20.33
cap124	50 x 50	0.427	421	26	680	66.35
cap131	50 x 50	0.000	198	0	0	5.71
cap132	50 x 50	0.000	412	0	0	11.48
cap133	50 x 50	0.000	287	0	0	8.29
cap134	50 x 50	0.000	200	0	0	6.31

Table 7

Results using scaling of demand constraints only: no reduction of warehouse capacities or storing of binary strings.

Akinc, A. and Khumawala, B. M. (1977), An Efficient Branch and Bound Algorithm for the Capacitated Warehouse Location Problem, Man Sci, Vol.23 n° 6, 585-594.

Baker, B.M., (1986), A partial Dual Algorithm for the Capacited Warehouse Location problem.EJOR, Vol. 23, 48-56.

Beasley, J.E., (1988), An Algorithm for Solving Large Capacitated Warehouse Location Problems, EJOR, Vol.33, 314-325.

Beasley, J.E., (1990), Or-Library: Distributing Test Problems by Electronic Mail, J. Opl. Res. Soc., Vol.11, 1069-1072.

Christofides, N. and Beasley, J.E., (1983), Extensions to a Lagrangean Relaxation Approach for the Capacitated Warehouse Location Problem, EJOR, Vol.12, 19-28.

Geoffrion, A.M. and McBride, R.D., (1977), Lagrangean Relaxation Applied to facility Location Problems, Working paper 263, Western Man Sci Inst, Univ. of California, LA.

Nauss, R.M. (1978), An Improved Algorithm for The Capacitated Facility Location Problem, J. Opl. Res. Soc., Vol.29, N°12, 1195-1202.

Van Roy, T.J., (1986), A Cross Decomposition Algorithm for Capacitated Facility Location, Opns. Res., Vol.34, 145-163.

APPENDIX I

1. Initialise l_j, and the list of binary strings corresponding to upper bounds, L = ø.

2 **Calculate** $z_i = f_i + \min \sum_{j=1}^{n} (c_{ij} - \lambda_j) x_{ij}$ *for all i*

3. Solve the 0-1 knapsack problem:

minimise $\sum_{i=1}^{m} z_i y_i$

subject to $\sum_{i=1}^{m} s_i y_i \geq \sum_{j=1}^{n} d_j$

4. Lower Bound = $\min \left\{ \sum_{i=1}^{m} z_i y_i \right\} + \sum_{j=1}^{n} \lambda_j d_j$

If the Lower Bound shows no improvement in 10 sucessive iterations perform step otherwise goto step 6.

5. Update the step-size for subgradient optimisation.

If the solution for $\{y_i\}$ corresponds to a string in L, go to step 6. Otherwise calculate upper bound using these warehouses;
add the corresponding string to the beginning of list L; delete the last string from L if $|L| > L_k$.

6. Total the allocations, x_{ij} for each column, using only those rows for which $y_i = 1$.
Use the totals to adjust Lagrange multipliers.

Repeat steps 2-6 till Lower Bound has shown no improvement for several interations.

If the Lower Bound ≥ best upper bound at any stage, go to step 11.

7. Take solutions to 0-1 knapsack problem corresponding to best Lower Bound.

8. Cycle throught rows trying to fix warehouses open or closed and trying to establish upper limit, $< s_i$ on warehouses which cannot be fixed.

9. When a complete pass throughl all warehouses gives no fixing or new , choose warehouse were fixing open narrowly failed (biggest increase in Lower Bound), or where fixing closed narrowly failed.

10. Branch to constraint this warehouse open (or closed).
Return to step 2.

11. If all the nodes are fathomed, optimal solution has been found; STOP.
Otherwise, backtrack to last non-fathomed node and go to step 2.

APPENDIX II

Demand point

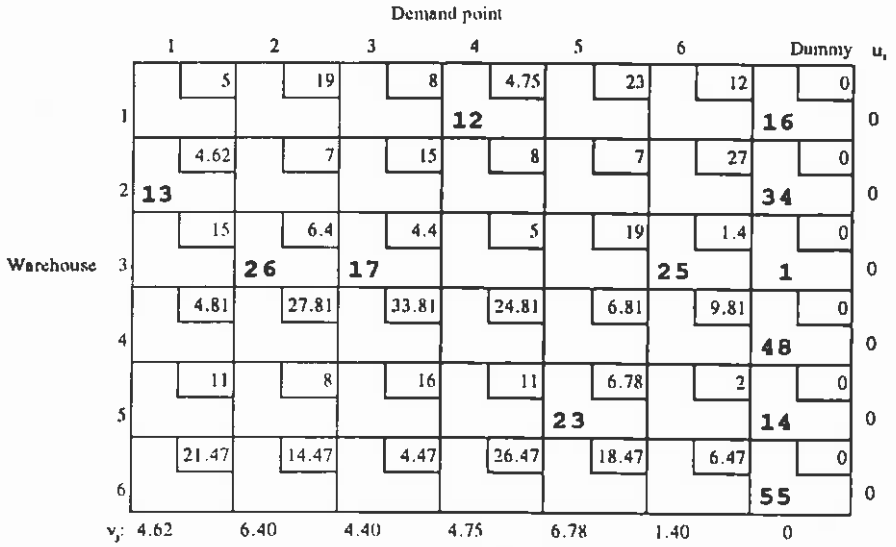

Table showing the optimal solution of a transportation problem, as given by Baker[5].