Electronic Journal of SADIO
http://www.dc.uba.ar/sadio/ejs/
vol. 4, no. 1, pp. 14{25 (2002)

Some useful cost allocation strategies for the shortest route relaxation of the set covering problem

Farhad Djannaty

Kurdistan University,Sanandaj,Iran
fdjanaty@yahoo.com

Abstract

Shortest route relaxation (SRR) of the Set Covering Problem (SCP) is a way of relaxing the problem to find quick lower bounds on the value of the objective function. A number of useful cost allocation strategies for the SRR are introduced which are applied to a number of test problems and computational results are presented.

Key or s. et or re ax ti n, et ov ri g, es du l C st

1 Introduction

Set problems comprising set covering, set partitioning, and set packing have attracted attention for many years and have applications in airline crew scheduling, bus crew scheduling, plant location, circuit switching, information retrieval, assembly line balancing, political districting and truck delivery [6].

Let $M=\{1, 2, \ldots, m\}$ be the set of m integers and let S denote a set of n subsets of M. Thus

$N = \{1, 2, \ldots, n\}$

$S = \{s_1; s_2; \ldots; s_n\}$ where $s_j \subseteq M$; $j \in N$:

Let

$$\left\{ \begin{array}{lll} a_{ij} = 1 & \text{if} & i \in s_j \\ a_{ij} = 0 & \text{if} & i \notin s_j \end{array} \right\} \quad (i = 1; \ldots; m \;;\; j = 1; \ldots; n)$$

The set covering problem (SCP) can be defined as follows:

$$\text{Minimize} \sum_{j=1}^{n} c_j x_j$$

st

$$\sum_{j=1}^{n} a_{ij} x_j \geq 1; \; i = 1; \ldots; m$$

$$x_j \in \{0; 1\} \; ; \; j = 1; \ldots; n :$$

The decision variable x_j indicates whether s_j is selected or not and c_j is the cost associated with selecting s_j. The problem can be interpreted as finding the minimum cost selection of subsets of S such that each member of M is covered by at least one member of the selected subset of S.

If we replace the "$\geq$" by "$=$" in each of the constraints of the above model, the modified problem is called the set partitioning problem (SPP). If "$\geq$" is replaced by "$\leq$" and the objective function is to be maximized, the resulting model is the set packing problem (SPK).

During the past 30-35 years a number of procedures have been developed which can deal with set problems. They can be found in Lemke, Salkin, and Spielberg [15], Salkin, and Koncal [19], and Salkin [20], Gerbracht, [13], and so on. They used either cutting plane algorithm and/or branch algorithm and then found that these algorithms are showed exponential and data dependent computing time Nemhauser, G.L. [16]. Beasley [3] has developed a tree search method to solve the SCP. Christofied and Paixao [17] reported good computational results with a steady state relaxation method. Fisher and Kedia [10] have developed a fast algorithm for a mixed set covering/partitioning problem. Of recent interest is the work of Thomson and Harcheand [14] who have developed a new exact method called column subtraction.

Among the heuristic methods, Beasley [5] has developed a Lagrangian heuristic which is reported to produce good quality results. Afif and el al have developed a new heuristic based on the flow algorithm of Ford and Fulkerson.

Beasley and Chu [4] have done several modification to genetic procedures which produces high quality solutions. In recent years, there has been some advancements in solving NP-complete problems Xie J. and Xing W. [21].

Set problems are categorized as NP-complete, which means that no polynomial time algorithm is known that guarantees to solve every instance of these problems. This increases the importance of relaxations which yield sharp lower and upper bounds as quickly as possible to be used in a tree search procedure. The set covering problem can be relaxed to form an assignment problem, a minimal spanning tree problem, a shortest route problem [8]. The shortest route relaxation is considered in this paper.

2 Shortest route relaxation of SCP and SPP

Referring to the set covering model (SCP) notation, let $h_j = |s_j|$ be the number of unit entries in column j and let $h_j^0 = \{i \mid a_{ij} > a_{i-1,j}; i \in s_j\}$, where $a_{0j} = 0$ and s_j is the set of row indices which contain a unit entry in column j of the A matrix. Therefore $k_j = |h_j^0|$ denotes the number of segments of ones in column j.

A network (E,V) can be constructed such that $V = \{v_1, \ldots, v_m, v_{m+1}\}$ is the set of vertices. For each column j of A associate a set of k_j arcs $E_j = \{(v_{i_1}, v_{i_1+l_1}), \ldots, (v_{i_{k_j}}, v_{i_{k_j}+l_{k_j}})\}$ $j = 1, \ldots, n$ such that arc $(v_{i_r}, v_{i_r+l_r})$ corresponds to rth segment in column j containing l_r unit entries running from row i_r up to row $i_r + l_r - 1$. We can define E such that $E = \bigcup_{j=1}^{j=n} E_j$. The two sets E and V as described above specify the structure of the shortest route relaxation of the set partitioning problem (SPP). Let the cost of an arc from node p to node q be the nonnegative real number d_{pq}^j. A valid relaxation is obtained provided that

$$\sum_{(v_p, v_q) \in E_j} d_{pq}^j = c_j \; ; \; j = 1, \ldots, n .$$

It can be verified that the shortest route relaxation of set problems (SRR) is a proper relaxation [8].

The shortest route relaxation for the set covering problem (SCP) can be obtained by a similar procedure. To provide for possible overcovers let E, the arc set , be increased by adding m backward arcs $\{(v_{i+1}, v_i) \mid i = 1, 2, \ldots, m\}$ with costs $d_{(i+1)(i)}^{n+i} = 0$ for $i = 1, 2, \ldots, m$. The introduction of these arcs creates cycles which allow rows to be overcovered.

3 Cost allocation strategies

In order to produce a lower bound or an upper bound to the set problem a strategy should be undertaken to allocate the full cost c_j among all segments associated with column j. Although there is no limitation to the number of ways in which a cost allocation can be made, some potentially useful ones are described below in detail:

Strategy 1: The first strategy for cost allocation is to distribute the full cost proportional to the cardinality of each segment. In other words, if a particular arc associated with column j starts at node v_p and ends at v_q then the cardinality of the associated segment is (q-p). If we allocate the cost $c_j/|h_j|$ to each row with a unit entry in column j, the cost for the segment becomes $d^j_{pq} = (q - p)c_j/|h_j|$ where h_j is the cardinality of column j.

Strategy 2: Let (v_p, v_q) be an arbitrary arc in column j and let $(rcount)_i$ be the row count of nonzeros in row $i \in \{p, (p+1), \ldots, (q-1)\}$ and let $(mincnt)_{pq} = \min\{(rcount)_p, (rcount)_{p+1}, \ldots, (rcount)_{q-1}\}$. In this strategy the cost allocation is made by scaling down the c_j in such a way that the cost allocated to the arc (v_p, v_q) is equal to

$$d^j_{pq} = \frac{\frac{1}{(mincnt)_{pq}}}{\sum_{(v_p,v_q)\in E_j} \frac{1}{(mincnt)_{pq}}} c_j$$

One reason for considering $(mincnt)_{pq}$ is that this number is equal to the number of multiple arcs from v_p to v_q and in order to increase the optimal value of the SRR solution we should allocate a greater cost to arcs having fewer multiple arcs.

Strategy 3: In this strategy the cost is allocated equally among all k_j segments, in column j, regardless of the segment cardinalities, that is, $d^j_{pq} = \frac{c_j}{k_j}\ \forall\ (v_p, v_q) \in E_j$. Since, it is assumed that each column of the SCP has, at least, one nonzero, therefore, $k_j > 0$ for $j = 1, \cdots, n$.

Strategy 4: Here the full cost is allocated to the first segment of each column and zero to the other segments in the column. For each column j

$$d^j_{pq} = \left\{ \begin{array}{lll} c_j & \text{if} & p = i_1 \\ 0 & \text{if} & p \neq i_1 \end{array} \right\}$$

A modification of this strategy is to allocate the full cost to a randomly selected segment and zero to other segments.

Strategy 5: In this strategy the cost is allocated randomly, that is, for each column j we find k_j nonnegative random numbers $p_1, p_2, \ldots, p_{k_j}$ such that $\sum_{i=1}^{i=k_j} p_i = 1$. The cost allocated to segment i is equal to $p_i \times c_j$. This strategy may be repeated with different sets of random numbers many times and the most desirable one chosen.

Strategy 6: For each segment of column j running from row p to row q-1 the following summation is calculated:

$$\sum_{i=p}^{i=q-1} \frac{1}{rcount_i};$$

where $rcount_i$ is the row count of row i. The cost c_j is then allocated to the segments in proportion to the above quantity. In other words, the cost allocated to the segment (p,q-1) of column j is calculated as :

$$d^j_{pq} = \frac{\sum_{i=p}^{i=q-1} \frac{1}{rcount_i}}{\sum_{i=1}^{i=m} a_{ij} \frac{1}{rcount_i}} c_j$$

The rationale behind strategy 6 is as follows. If, for a given row l; $rcount_l$ is large, that row can be covered by many columns and the probability of selecting the arc corresponding to a given segment, covering row l is low. Let the segment $(p, q-1)$ of column j contain row l. The cost allocated to the segment, d^j_{pq}, may be considered as the sum of "row costs" of rows included in the segment. If a given row is covered by many segments then the row cost may be viewed as 1 divided over the number of such segments, that is, $\frac{1}{rcount_i}$. In this cost allocation strategy the row cost is inversely proportional to the number of segments containing that row. Therefore, the cost is scaled down in proportion to the sum of row costs of rows included in each segment.

Strategy 7: Let h_j be the number of nonzeros in column j. In this strategy the cost c_j is allocated in proportion to the following quantity.

$$\sum_{i=p}^{i=q-1} \left\{\frac{1}{rcount_i} \sum_{j'=1}^{j'=n} a_{ij'} \frac{c_{j'}}{h_{j'}}\right\}$$

In other words, the cost allocated to the above segment is calculated as follows:

$$d^j_{pq} = \frac{\sum_{i=p}^{i=q-1} \left\{\frac{1}{rcount_i}\right\} \sum_{j'=1}^{j'=n} a_{ij'} \left\{\frac{c_{j'}}{h_{j'}}\right\}}{\sum_{i=1}^{i=m} a_{ij} \left\{\frac{1}{rcount_i}\right\} \sum_{j'=1}^{j'=n} a_{ij'} \left\{\frac{c_{j'}}{h_{j'}}\right\}} c_j$$

In order to justify the above formula we can view the costs as being allocated in two stages. Let the costs be allocated initially using strategy 1. According to this strategy the cost $\frac{c_{j'}}{h_{j'}}$ is allocated to each nonzero entry in column j'. In this manner each unit entry in the A matrix is assigned an initial cost. In the second cost allocation stage we try to attach a fixed pseudo cost to each row, like the cost c_j attached to column j, by using the initial costs assigned to nonzeros of the A matrix. A reasonable cost for this purpose is the average cost of a unit entry in each row, that is, $averg_i = \frac{1}{rcount_i} \sum_{j'=1}^{j'=n} a_{ij'} \frac{c_{j'}}{h_{j'}}$ is the cost associated with row i. Another way of thinking of $averg_i$ is the cost or a proportion of the column cost to be paid by any route crossing row i through a segment of that column containing row i. We consider the quantity

$\frac{\sum_{i=p}^{i=q-1} averg_i}{\sum_{i=1}^{i=m} averg_i}$ as a proportion of the column cost to be allocated for crossing rows p to q - 1 via that column. Therefore, once this quantity is associated with row i it is not changed throughout the cost allocation process and any cost is allocated in proportion to such a quantity for each segment.

Strategy 8: This strategy is a modification of strategy 6 as follows: Let A3 be the set of columns whose costs are among the first one-third lowest cost columns. In other words, if columns are sorted in the increasing order of the costs, $A3 = \{j | j \le \lceil n/3 \rceil\}$. Let r_cnt_i be the row count of row i in a matrix comprised of all columns in A3. In this strategy the cost c_j is distributed between arcs associated with column j in proportion to the following quantity:

$$\sum_{i=p}^{i=q-1} \frac{1}{(r\_cnt_i + 1)^{1.5}}$$

In other words, the cost allocated to the above segment is calculated as follows:

$$d^j_{pq} = \frac{\sum_{i=p}^{i=q-1} \frac{1}{(r\_cnt_i+1)^{1.5}}}{\sum_{i=1}^{i=m} a_{ij} \frac{1}{(r\_cnt_i+1)^{1.5}}} c_j$$

The reason for considering the first one-third lowest cost columns is because, for non-unicost SCP, the optimal columns generally consists of columns which have low costs. [4]

The following theorem can help us to create new cost allocation strategies.

Theorem: Any weighted average of a number of cost allocation strategies for a column is a valid cost allocation strategy for that column.

Proof: Let $stno$ be the number of cost allocation strategies combined and c_{it} $i = 1, 2, \ldots, k_j$ and $t = 1, \ldots, stno$ be the cost allocated to segment i in the cost allocation strategy t. Let α_t; $t = 1, \ldots, l$ be positive real numbers such that $\sum_{t=1}^{t=stno} \alpha_t = 1$. It is enough to show that this weighted average of the cost allocation strategies are a valid cost allocation strategy. This can be carried out by showing that they add up to the full cost c_j as follows:

$$\sum_{i=1}^{i=k_j} \sum_{t=1}^{t=stno} \alpha_t c_{it} = \sum_{t=1}^{t=stno} \sum_{i=1}^{i=k_j} \alpha_t c_{it} = \sum_{t=1}^{t=stno} \alpha_t \sum_{i=1}^{i=k_j} c_{it} = \sum_{t=1}^{t=stno} \alpha_t c_j = c_j \sum_{t=1}^{t=stno} \alpha_t c_j \times 1 = c_j.$$

4 Computational evaluation of SRR

4.1 Generation of 2 part duty crew scheduling problems

Two part duty crew scheduling is a problem which occurs in bus crew scheduling. In this problem, each column of the A matrix comprises two separate segments of ones and represents a possible driver schedule. An important feature of this problem is that, each driver is required to work only on one bus in each of

his two duty periods. That is, each driver spends his first duty period on one bus and his second duty period on one, usually different, bus. An algorithm (CSGEN), described below, has been developed to generate a set of 2 part duty crew scheduling problems.
The number of rows (rowmax) and the number of columns (nvar) are input to the algorithm. For the sake of simplicity and because the costs will be sorted by the optimizer in ascending order it was decided that the first 10 columns be assigned the cost of 1, and the second 10 columns are assigned the cost of 2 and after each group of 10 columns the cost is increased by 1. The number 10 can be changed to any other number (fix), as it is also an input to the problem. Let the first segment be $(p_1;q_1)$ and the second segment be named $(p_2;q_2)$. p_1 and p_2 are taken as two independent integer random numbers in the interval $[1; rowmax - 2]$ and another pair of integer random numbers in $[2, 6]$ are made as the cardinalities of the two segments. A check is carried out for each column of the A matrix to make sure that the segments do not overlap each other. If segments overlap each other this column is ignored and another column is generated. This procedure is repeated until a feasible column is produced. Let the first segment start from row p_1 to row q_1 and the second segment start from row p_2 to row q_2 such that $q_1 < p_2$ then the nonzeros of the column are printed. The row counts are calculated and if a row count of zero occurs the problem is declared as infeasible and ignored and the procedure CSGEN is repeated until a feasible problem is produced. In general, 6 problems of this type were generated with sizes ranging from 200 by 1000 to 300 by 2000. In order to generate SCP problems in which LP and IP solutions are different the procedure CSGEN has to be executed many times.

4.2 The collection of test problems

The characteristics of 15 problems which are used as test problems in this paper are presented in Table 1. Problem AIR1 belongs to a set of 6 problems generated by Powers [18] and used by El-Darzi [8]. Problems RDM3, RDM4, RDM6, and RDM7 are taken from a set of 14 randomly generated problems supplied by Paixao [17]. Problem SCP51 is taken from a set of 25 problems from Balas and Ho [2], and Problems SCPA1, SCPB1, and SCPE1 are taken from a set of OR test problems provided by Beasley [3]. DUTY problems are two part duty crew scheduling problems generated randomly using the algorithm CSGEN. The data files for these models were converted to both MPSX format and a special purpose format for the SRR procedure. Columns 6 and 7 are optimal LP and IP found by FORTMP [11]. Since this optimizer solves test problems by an exact procedure, in which a message is sent to the user in case, optimality is not reached, the IP found for the last 6 problems, are assumed to be optimal.

4.3 Lower bounds and upper bounds

This first experiment involved the application of the eight cost allocation strategies in the shortest route relaxation algorithm. The lower bounds obtained by

Problem Characteristics						
Prob. name	No. of rows	No. of cols	No. of Nonzero	Av.Nonzero per column	LP solution	IP solution
AIR1	159	416	2203	5	16600	16610
RDM3	101	109	784	7	95.06	96
RDM4	100	106	742	7	93.57	97
RDM6	100	130	884	6	98.06	99
RDM7	98	98	704	7	86	87
SCP51	200	2000	11955	4	251.23	253
SCPA1	300	3000	18000	6	246.84	253
SCPB1	300	3000	47921	15	64.54	69
SCPE1	50	500	5414	10	3.48	5
DUTY1	200	1000	8944	8	245	245
DUTY2	200	1000	8906	8	245.5	246
DUTY3	200	2000	17839	8	259	260
DUTY4	200	2000	17700	8	310	311
DUTY5	300	2000	17862	8	521.5	523
DUTY6	300	2000	17896	8	507	508

Table 1: Test problems

each strategy for each problem is presented in Table 2.

Problem Name	Cost allocation strategy							
	1	2	3	4	5	6	7	8
AIR01	12188.7	5629.94	11996.75	4390.0	8918.76	13362.8	13631.4	12830.0
RDM3	35.68	31.03	35.59	10.0	24.53	38.85	48.92	66.04
RDM4	38.72	35.51	38.51	5.0	24.67	42.31	48.72	61.11
RDM6	47.26	38.0	46.3	12.0	18.83	50.21	55.51	64.61
RDM7	38.72	30.28	34.09	9.0	24.53	42.31	48.72	56.4
SCP51	113.62	106.64	109.98	7.0	78.9	115.08	116.24	129.05
SCPA1	97.72	96.6	96.6	7.0	62.27	98.53	100.2	105.2
SCPB1	22.17	21.19	21.46	2.0	5.41	22.35	22.38	22.97
SCPE1	2.97	1.34	1.45	1.0	0.23	2.97	2.99	2.95
DUTY1	145	136	170	59	149.55	177.7	167	187.32
DUTY2	132.67	117.77	154	62	141.2	164.54	163.46	187.32
DUTY3	159.4	146.57	184	15	98.9	190.4	192.34	201.11
DUTY4	171.87	157.9	199	34	75.18	208.15	215.79	233.44
DUTY5	270.9	257.8	335	74	178.35	345.9	322	388.28
DUTY6	276.8	264.6	323.5	74	183.97	332.92	346.89	398.51

Table 2: SRR lower bounds for 8 strategies

Strategy 8 produces the strongest lower bound in 13 problems strategy 7 is best for the remaining two problems. Strategy 4 produces the weakest lower bounds in 8 problems. In general strategy 8 is the best strategy to produce lower bounds. These results are not surprising because in strategy 4 which is the weakest, the only item of information used in the cost allocation of column j is the column cost c_j which is allocated to the ¯rst segment in each column. In strategies 3, and 5 $\{c_j\}$ plus the number of segments in each column k_j are used. In strategy 1 the column cost $\{c_j\}$, the number of segments k_j, and the segment cardinalities are used. In strategy 2 the column cost and the number of multiple arcs are involved. In strategy 6 all items of strategy 1 plus the row counts (rcount) are used. In strategy 7 all information of strategy 6 plus the costs and number of nonzeros of other columns are utilized. In strategy 8 which produces the strongest bounds in 13 problems, all information of strategy 6 and problem speci¯c knowledge that the optimal columns of SCP problems are usually among the least cost columns[4], is utilized. It turns out from the computational results and the above comment that when more information about the structure of the columns and rows of the A matrix are used in allocating the cost, the lower bound produced is likely to be stronger.
In order to get an idea about the performance of this procedure, the lower bounds produced by strategies 6, 7 and 8 are compared with the ones obtained by the Assignment Relaxation version 1 (ASP1), and Assignment Relaxation version 2 (ASP2). The lower bounds on problems AIR01, RDM4, RDM6, and RDM7 obtained by ASP1 and ASP2 relaxations are directly extracted from El-Darzi [8]. Of the problems considered in this paper, only these four problems were considered by El-Darzi. The ASP1 and ASP2 codes used by El-Darzi are not available and it was therefore impossible to apply them to other problems. These lower bounds and the ones produced by strategies 6, 7,and 8 are summarized in Table 3. Based on the ¯gures of Table 3, in terms of the quality of

P.Name.	ST6	ST7	ST8	ASP1	ASP2
AIR01	13362.8	13631.5	12830.0	13424	14488.2
RDM4	42.31	48.21	61.11	47.8	53.26
RDM6	50.55	55.54	64.61	47.3	51.14
RDM7	42.31	48.72	56.4	47.2	50.66

Table 3: Lower bounds obtained by 2 network relaxations

the bounds, shortest route relaxation using strategy 8 is better than ASP1 and, ASP2 for 3 problems and ASP2 is better than both strategies 6 and it is also better than strategy 7 for 3 problems. The execution times corresponding to Table 3 are provided in Table 4. ASP1 and ASP2 runs were originally carried out on a Honeywell Multics DP68. This machine is estimated to be 1.7 times slower than a Pentium PC operating at 90 Mhz. Thus the execution times for ASP1 and ASP2 have been converted to be comparable with Pentium execution times. According to Table 4 SRR is quicker than both ASP1 and ASP2.

P.Name.	ST6	ST7	ST8	ASP1*	ASP2*
AIR01	1.672	1.603	1.618	7.866	38.592
RDM4	0.851	0.838	0.841	1.214	7.628
RDM6	0.914	0.919	0.900	1.807	5.793
RDM7	0.833	0.817	0.813	0.847	6.443

Table 4: Execution times (secs) on a (Pentium 90MHz) Machine
* Converted from Honeywell Multic DP68 times

Conclusions

The computational results presented in this paper shows that the shortest route relaxation (SRR) of the set covering problem can produce better and quicker lower bounds than the assignment relaxation (ASP) of the set covering problem established by Elia El-Darzi [8]. Computational result reveals that the more speci¯c knowledge of the problem is used in the cost allocation strategy, the stronger is the lower bound obtained by that strategy. The lower bounds obtained for two part duty crew scheduling problems are better in comparison to other problems. More investigations may lead to sharper lower bounds for the SCP.

References

[1] A¯f M., Hi¯ M., Paschos V., and Zissimopoulos V. (1995), A new e±cient heuristic for the minimum set covering problem, Journal of the Operational Research Society, 46, pp1260-1269.

[2] Balas, E. and Ho, A.(1980), Set covering algorithms, using cutting planes, heuristics and subgradient optimization: A new computational study, Mathematical Programming Study, 12, pp37-60.

[3] Beasley, J.E. (1987), An algorithm for set covering problems, European Journal of Operational Research, 31, pp85-93.

[4] Beasley, J.E. and Chu, P.C.(1996), A genetic algorithm for the set covering problem, European Journal of Operational Research, 94, 2, pp392-404.

[5] Beasley J.E. (1990), A Lagrangian heuristic for set covering problems, Naval Research Logistics Quarterly, 37, pp151-164.

[6] Darby-Dowman, K. and Mitra, G.(1985), An extension of set partitioning with application to scheduling problems, European Journal of Operational Research, 21, 200-205.

[7] Marchiori E. and Steenbeek A.:(1998), An iterated heuristic algorithm for the set covering problem

[8] El_Darzi,E.(1988), Methods for solving the set covering and set partitioning problems, using graph theoretic (relaxation) algorithms, PhD thesis, Brunel University

[9] Djannaty,F.(1993), Network based heuristics for the set covering problem, Phd thesis, Brunel University.

[10] Fisher M.L., and Kedia P. (1990), Optimal solution of set covering /partition roblems using dual heuristics, Management Science, 36, No 6, pp674-687.

[11] FortMP Manual, Release 2 (1997), Numerical Algorithms Group Limited and Brunel University

[12] Gar¯nkel R., and Nemhauser G.L (1972) Integer Programming, John Wiley&Sons.

[13] Gerbracht R. (1978) A new algorithm for very large crew pairing problems, 18th AGIFORS Symposium, Vancouver, Canada.

[14] Harche F., and Thompson J.L. (1994) The column subtraction algorithm: An exact method for solving weighted set covering, packing and partitioning problems, Computers and Operations Research, 21, 6, pp689-705.

[15] Lemke C.E., Salkin H.M., and Spielberg K. (1969), Set covering by single branch enumeration with linear programming subproblems, Operations Research, 19, pp998-1022.

[16] Nemhausewr G.L.,and Wolsey L.A.(1988), Integer and Combinatorial Optimization, Wiley, 1988.

[17] Paixao, J.(1983), Algorithms for large scale set covering problems, Imperial College, London, internal report.

[18] Powers D. (1987), Investigation and construction of set covering and set partitioning test problems, BSc dissertation, Brunel University.

[19] Salkin H.M., and Koncal R.D. (1973), Set covering by an all-integer algorithm: computational experience, Journal for the Association of Computer Machinery, 20, pp189-193.

[20] Salkin H.M.(1975), Integer programming, Addison-Wesely.

[21] Xie J. and Xing W. (1998), Incorporating Domain Speci¯c Knowledge into Evolutionary Algorithms, Bejing Mathematics, Vol.4, no 2, pp. 131-139.