A Tabu Search Algorithm for the Resource-Constrained Assignment Problem

Author(s): Abraham P. Punnen and Y. P. Aneja

Source: *The Journal of the Operational Research Society*, Vol. 46, No. 2 (Feb., 1995), pp. 214-220
Published by: Palgrave Macmillan Journals on behalf of the Operational Research Society
Stable URL: http://www.jstor.org/stable/2583990
Accessed: 05/03/2010 04:11

Journal of the Operational Research Society (1995) 46, 214–220

A Tabu Search Algorithm for the Resource-Constrained Assignment Problem

ABRAHAM P. PUNNEN and Y. P. ANEJA
University of Windsor, Canada

Efficient algorithms are availabe to solve the unconstrained assignment problem. However, when resource or budgetary restrictions are imposed, the problem becomes difficult to solve. We consider such a resource-constrained assignment problem and present a tabu search heuristic to solve it. Extensive computational results are presented which establish the superiority of the proposed algorithm over the existing algorithms. Our adaptation of tabu search uses strategic oscillation, randomized short-term memory and multiple start as a means of search diversification.

Key words: tabu search, assignment problem, heuristics

INTRODUCTION

The assignment problem[1] is perhaps the most studied problem in the linear and integer programming literature. In addition to its applicability as a modelling tool in many real-life situations, the assignment problem also arises as a sub-problem in the solution procedures of more complex combinatorial optimization problems. In the recent past, researchers have developed efficient algorithms to solve the assignment problem with millions of variables in a reasonable amount of computational time.

However, in many applications of the assignment problem, resource (budget) constraints also play an important role[2–8]. Unlike the classical assignment problem, such resource-constrained assignment problems (RCAP) are difficult to solve exactly. In fact, it can be shown that the RCAP is strongly *NP*-complete. This rules out the existence (modulo $P = NP$) of even a pseudo-polynomial algorithm or fully polynomial approximation scheme to solve the problem. The RCAP can be formulated mathematically as follows.

RCAP:
$$\text{minimize} \sum_{j=1}^{n} C_{j\pi(j)}$$

subject to

$$\pi \in \Pi$$

$$\sum_{j=1}^{n} a^{k}_{j\pi(j)} \leqslant b_k, \qquad 1 \leqslant k \leqslant p,$$

where C_{ij}, a^k_{ij} and b_k, $1 \leqslant i \leqslant n$, $1 \leqslant j \leqslant n$, $1 \leqslant k \leqslant p$ are real given numbers, Π is the set of all permutations of $\{1, 2, \ldots, n\}$; and for any given permutation π, $\pi(j)$ identifies the job to which person j is assigned.

Several researchers have studied the RCAP in various contexts. Aggarwal[2], Kennington and Mohamadi[4] and Gupta and Sharma[3] have studied the case when $p = 1$. Specially structured side constraints are investigated by Aboudi and Nemhauser[5]. Aboudi and Jornsten[7] and Mazzola and Neebe[6] investigated the general RCAP. Mazzola and Neebe[6] presented exact and heuristic algorithms to solve the general RCAP and reported impressive computational results based on experiments conducted with a wide variety of test problems.

Although the Mazzola–Neebe heuristic performs well on the test problems they considered, it is observed to perform poorly when the range of C_{ij} is large and the side constraints are tight. This warrants the need for efficient heuristics to solve such difficult instances of RCAP.

Correspondence: Y. P. Aneja, Faculty of Business Administration, University of Windsor, Windsor, Ontario, Canada N9B 3P4

In this paper, we suggest a tabu-search-based heuristic algorithm to solve the RCAP. Our tabu search heuristic is observed to perform better in cases where the Mazzola–Neebe heuristic produced poor quality solutions. Thus, the combined use of the tabu-search and the Mazzola–Neebe heuristics produces good quality, near optimal, solutions to the RCAP. Our adaptation of tabu search uses multiple start, strategic oscillation and randomized short-term memory as a means of diversification of search paths. We also report results of extensive computational experiments conducted using our tabu search algorithm and Mazzola–Neebe heuristic. These results establish the desirability of the proposed tabu search approach.

TABU SEARCH AND RCAP

The metaheuristic, tabu-search, was proposed by Glover[9,10] as a technique to overcome entrapment at a local optimal solution in local search-based heuristic algorithms. For some early works on tabu search we refer to References 9–12. In this paper we adapt ideas from tabu search to obtain near-optimal solutions to RCAP. First, we summarize below some definitions adapted from tabu search to fit in the context of the RCAP.

A *move* is a transition from one permutation to another by interchanging positions of two elements i and j. The unordered pair (i, j) is called the *attribute of the move*. Let π be any permutation and π_{ij} be the permutation obtained by interchanging positions i and j (i.e. making a move from π with attribute (i, j)). Then π_{ij} is given by

$$\pi_{ij}(r) = \pi(r) \quad \text{if} \quad r \neq i, j$$

$$\pi_{ij}(i) = \pi(j)$$

and

$$\pi_{ij}(j) = \pi(i).$$

Let

$$f_k(\pi) = \sum_{j=1}^{n} a^k_{j\pi(j)} \quad \text{and} \quad C(\pi) = \sum_{j=1}^{n} C_{j\pi(j)}.$$

Given $f_k(\pi)$, $1 \leqslant k \leqslant p$ and $C(\pi)$, one can obtain $f_k(\pi_{ij})$ and $C(\pi_{ij})$ in $O(1)$ time using the formula:

$$f_k(\pi_{ij}) = f_k(\pi) - a^k_{i\pi(i)} - a^k_{j\pi(i)} + a^k_{i\pi(j)} + a^k_{j\pi(i)}, \quad 1 \leqslant k \leqslant p.$$

$$C(\pi_{ij}) = C(\pi) - C_{i\pi(i)} - C_{j\pi(j)} + C_{i\pi(j)} + C_{j\pi(i)}.$$

Let

$$W(\pi) = C(\pi) + \sum_{k=1}^{p} \alpha_k \max(0, f_k(\pi) - b_k),$$

where α_k, $1 \leqslant k \leqslant p$, is a parameter. We call $W(\pi)$ the working objective function and it is used to determine the value of a move. A best move (r, s) from a permutation π is determined by:

$$W(\pi_{rs}) = \min_{(ij) \in N \setminus T(\pi)} (W(\pi_{ij}))$$

where N is the set of $(n(n-1))/2$ unordered pairs (ij), $1 \leqslant i \leqslant n$, $i < j \leqslant n$ and $T(\pi)$ is a subset of N called the set of inadmissable moves. The membership of a pair (ij) in $T(\pi)$ is determined using a list called TABU_LIST and a parameter called TABU_SIZE, which is the length of the TABU_LIST and an aspiration criteron test. Generally, in most of the tabu-search algoritms[12], TABU_LIST is implemented as a deterministic circular list. If the length of the list, TABU_SIZE, is too small then the algorithm may end up cycling, and if the length is too large, good solutions are more likely to be skipped. To balance the trade-off between cycling and solution quality, we consider a randomized tabu list. In our implementation, the tabu list has two components, a deterministic component and a randomized component. A pair (ij) is said to be tabu if it belongs either to the deterministic component or to the randomized component and $P(i, j) \leqslant 0.5$, where $P(i, j)$ is a uniformly distributed random number drawn from the population [0, 1]. We use TABU_SIZE1 to denote the length of the deterministic part of the tabu list and TABU_SIZE2 to denote the length of the randomized part of the tabu list. Thus TABU_SIZE = TABU_SIZE1 + TABU_SIZE2.

The TABU_LIST is maintained as an $n \times n$ matrix TABU_MAT. It is initialized to zero and updated using an iteration counter ITER_C and the current tabu size CTSIZE. ITER_C and CTSIZE are initialized to zero and ITER_C is incremented by one after each iteration (move). Similarly, after each iteration, CTSIZE is increased by one if CTSIZE < TABU_SIZE and remains unchanged otherwise. After each move with attribute (i, j), TABU_MAT is updated by assigning TABU_ MAT(i, j) = ITER_C. Now, at any iteration, a pair (i, j) is tabu if and only if:

$$(\text{ITER\_C} - \text{TABU\_MAT}(i, j)) \leq \min\{\text{CTSIZE}, \text{TABU\_SIZE1}\}$$

or

$$(\text{ITER\_C} - \text{TABU\_MAT}(i, j)) \leq \text{CTSIZE} \quad \text{and} \quad P(i, j) \leq 0.5.$$

Thus, the tabu status of a pair (i, j) can be verified in $O(1)$ time.

A pair (i, j), which is tabu can overcome its tabu status if the resulting working objective function value (the $W(\pi_{ij})$) is strictly less than the best working objective function obtained value so far. This is the aspiration level criterion. Now, a pair (i, j) is inadmissable (i.e. $(i, j) \in T(\pi)$) if and only if (i, j) is tabu and failed the aspiration test.

DIVERSIFICATION OF SEARCH PATHS

In a simple tabu search algorithm, one starts with a solution (possibly infeasible) and moves to a 'best' admissible solution. The process continues until a fixed number of iterations is exhausted and the best feasible solution is obtained. However, recent research on the applications of tabu search on several classes of hard combinatorial optimization problems asserts the need for diversification of search paths to obtain good solutions. Several techniques are used by researchers to achieve this goal. These include:

(1) multiple starts[11,13];
(2) randomized moves[14,15];
(3) long-term memory[13,16];
(4) strategic oscillations[17];
(5) varying tabu_size and moving gaps[13];
(6) target analysis[12] etc.

Although each of the aforementioned diversification strategies are useful on different sets of problems, to reduce the number of combinations that need to be tried, we conducted some initial experiments to identify promising strategies in the context of RCAP. Strategic oscillation and multiple starts produced good quality results as compared with other methods, and hence we fixed them as diversification in our further experiments.

We have implemented strategic oscillation using a shifting penalty approach. At the beginning of the search, a penalty α_k, $1 \leq k \leq p$, is fixed for the violation of each constraint. Then, after a fixed number of iterations, say PLT_LIM, is exhausted, the behaviour of the constraints is examined. For constraints that are violated more than (PLT-LIM) − (TOL_LIM) times, where TOL_LIM is a parameter called the tolerance limit, in the past PLT_LIM iterations the penalty is increased by multiplying by 2. Similarly, for constraints that are satisfied more than (PLT_LIM) − (TOL_LIM) times in the last PLT_LIM iterations, the penalty is decreased by dividing it by 2. The search continues until a fixed number, say MAX_ITER, of iterations is exhausted.

To initiate the search, we have identified two potential candidates. One is the solution obtained from the Mazzola–Neebe heuristic and the other is the solution to the assignment problem without considering the resource constraints. The Mazzola–Neebe heuristic is observed to produce good solutions in several instances and, by using it as the starting solution, tabu search explores the possibility of improving it. The rationale behind choosing the unconstrained assignment problem solution as another starting solution is as follows: it is obvious that ranking the solutions to the unconstrained assignment problem[18], in increasing order of cost, and picking up the least index solution in this arrangement that satisfies the constraints, solves the RCAP optimally. Thus, exploring the vicinity of the unconstrained assignment solutions using tabu search coupled with different search diversification strategies is a promising approach. In our computational experiments we have several examples in which starting with the unconstrained assignment problem solution produced better results compared with using the Mazzola–Neebe heuristic solution as a starting point. Likewise, we have several other examples in which using the Mazzola–Neebe heuristic solution as a starting point produced better solutions compared with using the unconstrained assignment solution as the starting point. Thus, we decided on using a double start, once with the Mazzola–Neebe heuristic and then with the assignment problem solution, and such an approach indeed produced good computational results.

COMPUTATIONAL RESULTS

The tabu search algorithm for the RCAP was coded in FORTRAN 77 and tested on an IBM 4381 computer system (performance: mips = 7.9, operating system VM/CMS). The experiments were conducted in two phases. In the first phase, we tried to fine-tune various parameters and search diversification strategies. Although we could not conclude any general rule applicable to all test problems generated, the use of strategic oscillation and a double start, once with the Mazzola–Neebe heuristic solution and then with the unconstrained assignment solution, showed superiority over other alternatives for diversifying search paths. Regarding TABU_SIZE, we observed that TABU_SIZE1 = 9, TABU_SIZE = 15 behaved reasonably well on average and thus we fixed these parameter values. The values of PLT_LIM was set at 15 and TOL_LIM was set equal to 1. Having fixed these parameters, we used the following sequence in performing the tabu-search:

/* READ problem DATA */
read n; p; C_{ij}; a_{ij}^k, $1 \leq k \leq p$; $1 \leq i \leq n$; $1 \leq j \leq n$ and ε;
/* set parameter values */
TABU_SIZE = 15, TABU_SIZE1 = 9, PLT_LIM = 15
TOL_LIM = 1, MAX_ITER = 200,
/* compute starting solutions and lower bound */

Step 1. Compute the Mazzola_Neebe heuristic solution, say π^*.
Step 2. Compute the unconstrained assignment solution, say π^0.
Step 3. Compute a lower bound LB to the optimal objective function (*possibly using the Lagrangian method *).
Step 4. If (π^0 is feasible) or (π^* is feasible and $(C(\pi^*) - \text{LB}) < \varepsilon$) then stop.
Step 5. Set START (π) = π^*, COUNT = 1. $\alpha_k = 1$, $1 \leq k \leq p$.
/* main loop */
Step 6. Perform tabu search (as explained earlier using the strategic oscillation solution) with START(π) as the starting solution for MAX_ITER iterations. Let BEST(π) be the best feasible solution obtained.
Step 7. COUNT = COUNT + 1, if (COUNT .GT. 3) GO TO 9
/* Restarting from the best solution */
Step 8. If BEST(π) $\neq \emptyset$ START(π) = BEST(π) else START(π) = π^*.
Set $\alpha_k = \alpha_k/2$, $1 \leq k \leq p$ and GO TO Step 6.
Step 9. If (COUNT .GT. 6) GO TO 11.
/* Restarting with assignment solution */
Step 10. START(π) = π^0, IF (COUNT .EQ. 4) set $\alpha_k = 1$ else $\alpha_k = \alpha_k/2$, $1 \leq k \leq p$. GO TO Step 6.
Step 11. Output the best solution so far.

The data for the test problems are generated randomly with variation in size and density. Four classes of test problems are considered:
(1) random problems;
(2) negatively correlated problems;
(3) decentralized problems; and
(4) decentralized negatively correlated problems.
Similar test problems were used by Mazzola and Neebe[6] in their computational experiments. Random problems are those in which C_{ij} and a_{ij}^k are uniformly distributed random integers in the interval [5, 500] and [5, 150] respectively. In decentralized problems, the C_{ij} are uniformly distributed random integers in the range [5, 500] but a_{ij}^k are specially structured. In this case, the set of indices (ij), $1 \leqslant i \leqslant n$, $1 \leqslant j \leqslant n$ is partitioned into p different groups $S_1, S_2, \ldots, S_p$ and

$$a_{ij}^k = \begin{cases} 0 \text{ if } (ij) \notin S_k \\ \text{a uniformly distributed random integer in the range } (5, 150) \text{ if } (ij) \in S_k.\ 1 \leqslant k \leqslant p. \end{cases}$$

In negatively correlated problems, C_{ij} are generated randomly in the range [5, 500] and strong negative correlation is incorporated between elements of the cost matrix (i.e. C_{ij}) and elements of the constraint matrices (i.e. a_{ij}^k). The range of elements of the constraint matrices is set at [5, 150]. Decentralized negatively correlated problems are decentralized problems in which strong negative correlation is incorporated between elements of the cost matrix and non-zero elements of the constraint matrix. The right-hand side, the b_k, is set equal to

$$\left(\sum_i \sum_j a_{ij}^k\right) - 10\text{n}$$

for the random and the negatively correlated problems. For the decentralized and negatively correlated decentralized problems we set

$$b_k = 1 + \left(\sum_i \sum_j a_{ij}^k\right) \Big/ 2n.$$

Note that in the test problems generated by Mazzola and Neebe[6], b_k is set to

$$b_k = \left(\sum_i \sum_j a_{ij}^k\right) \Big/ n$$

and C_{ij} are uniform random integers in the range (50, 150). For such problems, we observed that the Mazzola–Neebe heuristic produced good quality solutions. However, as the range of C_{ij} increased and/or the b_k decreased, the Mazzola–Neebe heuristic tended to produce poor quality solutions. Thus, we decided to choose moderately tight right-hand sides with a cost range of [5, 500].

In our final experiments, for each problem class, eight problem sizes (i.e. n and p) were considered and for each problem size, five problems were solved and the average of the following observations were noted: objective function value in the Mazzola–Neebe heuristic, time taken in seconds of CPU-time; lower bound for the optimal objective function value; objective function value produced in the tabu-search heuristics, the iteration at which the best solution is obtained; and the time in seconds of CPU time to reach the best solution. The results are summarized in Table 1 to Table 4. In the tables, ITER refers to the average (over five problems) iteration index at which the tabu search heuristic produced the best solution, and the TIME in the tabu search heuristic is the average time taken to reach the iteration given by ITER. The quantity given in brackets refers to the total time taken by the tabu search heuristic. In the Mazzola–Neebe heuristic, as well as in the tabu search heuristic, the column denoted RATIO in Tables 1 to 4 refers to the average value of

$$\frac{\text{heuristic solution value} - \text{lower bound to the optimal solution value}}{\text{lower bound to the optimal objective function value}}$$

over five problems.

TABLE 1. *Random problems*

n	p	Lower bound	Mazzola–Neebe heuristic			Tabu search heuristic			
			Objvalue	TIME	RATIO	Objvalue	ITER	TIME	RATIO
30	6	1360	1815	09.3	0.334	1641	709	47.8 (81)	0.206
40	7	1497	2357	32.3	0.574	1842	395	54.9 (167)	0.230
50	8	1693	2468	65.2	0.457	2132	149	37.3 (301)	0.253
60	7	1710	2489	89.4	0.455	2106	119	38.5 (389)	0.231
70	7	1725	2700	172.3	0.565	2105	159	71.0 (536)	0.220
80	5	1478	1978	181.3	0.338	1702	97	42.1 (522)	0.151
90	5	1582	1869	235.0	0.181	1711	64	35.8 (672)	0.081
100	5	1596	1797	297.0	0.125	1639	37	25.5 (830)	0.026

TABLE 2. *Decentralized Problems*

n	p	Lower bound	Mazzola–Neebe heuristic			Tabu search heuristic			
			Objvalue	TIME	RATIO	Objvalue	ITER	TIME	RATIO
30	6	1523	2045	12.0	0.342	1865	644	44.0 (82)	0.224
40	7	1568	2015	26.7	0.285	1889	290	40.8 (169)	0.204
50	8	1534	2286	66.2	0.490	1953	187	47.0 (302)	0.273
60	7	1547	1961	92.0	0.267	1782	42	13.7 (393)	0.151
70	7	1703	2073	156.4	0.217	1957	61	27.5 (542)	0.149
80	5	1772	2083	179.1	0.175	1971	11	4.93 (538)	0.112
90	5	1831	2216	230.2	0.210	2050	21	12.0 (687)	0.119
100	5	1851	2006	159.1	0.082	1925	23	16.2 (848)	0.038

TABLE 3. *Negatively correlated problems*

n	p	Lower bound	Mazzola–Neebe heuristic			Tabu search heuristic			
			Objvalue	TIME	RATIO	Objvalue	ITER	TIME	RATIO
30	6	5268	5594	20.1	0.061	5517	449	31.0 (83)	0.047
40	7	6954	7438	55.5	0.069	7299	240	33.4 (167)	0.049
50	8	8287	8772	114.6	0.058	8695	270	67.5 (300)	0.049
60	7	9661	10 171	177.9	0.052	10 067	70	22.8 (392)	0.042
70	7	11 078	11 625	268.7	0.049	11 486	7	3.1 (539)	0.036
80	5	11 960	12 487	273.5	0.044	12 288	35	15.9 (538)	0.022
90	5	13 125	13 650	456.1	0.040	13 470	16	9.1 (683)	0.026
100	5	14 597	15 073	499.4	0.032	14 998	53	37.2 (843)	0.027

TABLE 4. *Negatively correlated decentralized problems*

n	p	Lower bound	Mazzola–Neebe heuristic			Tabu search heuristic			
			Objvalue	TIME	RATIO	Objvalue	ITER	TIME	RATIO
30	6	3326	3844	20.7	0.155	3653	251	17.1 (82)	0.098
40	7	4058	4760	61.4	0.172	4477	256	35.6 (167)	0.103
50	8	4795	5650	127.8	0.178	5290	428	106.6 (299)	0.103
60	7	5385	6163	178.0	0.144	5892	58	18.9 (392)	0.094
70	7	6098	6621	253.6	0.085	6520	84	37.7 (539)	0.069
80	5	6695	7413	310.6	0.107	7133	238	106.5 (537)	0.065
90	5	7611	8022	420.4	0.054	7960	8	4.4 (681)	0.045
100	5	8235	8814	575.9	0.070	8701	205	144.5 (846)	0.056

CONCLUSIONS

In this paper we have presented a tabu-search-based algorithm to solve the RCAP. The algorithm uses randomized short-term memory, multiple start and strategic oscillation as means of search diversification. The results given in Table 1 through Table 4 are based on a general-purpose tabu search algorithm using general settings for various parameters. However, for particular applications, fine tuning of the problem parameters, on a problem by problem basis, can produce improved results. Such an approach is useful in applications which are difficult to solve by an exact algorithm but the quality of the solution is important. In summary, our study indicates that using the Mazzola–Neebe heuristic coupled with the tabu search is a good approach to obtaining quality approximate solutions for a wide variety of RCAPs.

Acknowledgement—We sincerely thank Dr J. B. Mazzola for providing us with the FORTRAN code for obtaining the heuristic solution used as a starting step in our algorithm.

REFERENCES

1. A. W. Kuhn (1955) The Hungarian method for the assignment problem. *Naval Res. Logist. Q.* **2**, 83–97.
2. V. Agarwal (1985) A Lagrangian relaxation method for the constrained assignment problem. *Comps and Opns Res.* **12**, 97–106.
3. A. Gupta and J. Sharma (1981) Tree search method for optimal core management of pressurized water reactors. *Comps and Opns Res.* **8**, 263–266.
4. J. L. Kennington and F. Mohammadi (1991) The singly constrained assignment problem. Technical Report 91-CSE-1, Department of Computer Science and Engineering, Southern Methodist University, Texas.
5. R. Aboudi and G. L. Nemhauser (1991) Some facets for an assignment problem with side constraints. *Opns Res.* **39**, 244–250.
6. J. B. Mazzola and A. W. Neebe (1986) Resource constrained assignment scheduling. *Opns Res.* **23**, 91–106.
7. R. Aboudi and K. Jornsten (1990) Resource constrained assignment problem. *Discrete Appl. Math.* **26**, 175–191.
8. J. P. Brans, M. Leclercq and P. Hansen (1973) An algorithm for optimal reloading of pressurized water reactors. In *Operational Research, '72.* (M. Ross, Ed). pp. 417–428. North Holland, Amsterdam.
9. F. Glover (1989) Tabu search part I. *ORSA. J. Computing* **1**, 190–206.
10. F. Glover (1990) Tabu search part II. *ORSA. J. Computing* **2**, 4–32.
11. D. de Werra and A. Hertz (1989) Tabu search techniques: A tutorial and an application to neural networks. *OR Spectrum* **11**, 131–141.
12. M. Laguna and F. Glover (1991) Integrating target analysis and tabu search for improved scheduling systems. Research Report, Graduate School of Business Administration, University of Colorado at Boulder.
13. F. Skorin-Kapov (1990) Tabu search applied to quadratic assignment problem, *ORSA J. Computing* **2**, 33–45.
14. E. L. Mooney and R. L. Rardin (1991) Tabu search for a class of scheduling problems. Research report CC-91-9, Institute for Interdisciplinary Engineering Studies, Purdue University.
15. A. P. Punnen and Y. P. Aneja (1993) Categorized assignment scheduling: a tabu search approach. *J. Opl Res. Soc.* **44**, 673–679.
16. J. Skorin-Kapov (1991) Extensions of a tabu search adaptation to quadratic assignment problem. Harriman School working paper HAR-90-006, State University of New York.
17. F. Glover, E. Taillard and D. de Werra (1991) A users guide to tabu search. Research Report, Graduate School of Business Administration, University of Colorado at Boulder.
18. K. G. Murty (1968) An algorithm for ranking all the assignments in order of increasing cost. *Opns Res.* **16**, 682–687.

Received June 1992; accepted June 1994 after one revision