A Very Large-Scale Neighborhood Search Algorithm for the Multi-Resource Generalized Assignment Problem

Mutsunori Yagiura* Shinji Iwasaki† Toshihide Ibaraki*

*Graduate School of Informatics, Kyoto University
Kyoto 606-8501, Japan
{yagiura,ibaraki}@i.kyoto-u.ac.jp

†Web Service Group, NTT Data Corporation
Kayabacho Tower Bldg., 21-2, Shinkawa 1-Chome, Chuo-Ku, Tokyo 104-0033, Japan
iwasakis@nttdata.co.jp

1 Introduction

We propose a metaheuristic algorithm for the multi-resource generalized assignment problem (MRGAP) [5]. MRGAP is a generalization of the generalized assignment problem (GAP) [8, 10, 11], which is one of the representative combinatorial optimization problems known to be NP-hard. To our knowledge, not much has been done for MRGAP after the work of Gavish and Pirkul [5] in spite of its practical importance, while many metaheuristic algorithms have been proposed for GAP [3, 4, 7, 10, 11].

Our algorithm is based on tabu search, and features a very large-scale neighborhood search, which is a mechanism of conducting the search with complex and powerful moves, where the resulting neighborhood is efficiently searched via the improvement graph [1, 2]. We also incorporate an automatic mechanism for adjusting search parameters, to maintain a balance between visits to feasible and infeasible regions.

We conducted computational experiments on benchmark instances called types C, D and E, and compared the proposed method with other existing algorithms. The results show that our algorithm is effective, especially for types D and E instances, which are known to be quite difficult.

2 Multi-Resource Generalized Assignment Problem

Given n jobs $J = \{1, 2, \ldots, n\}$ and m agents $I = \{1, 2, \ldots, m\}$, we undertake to determine a minimum cost assignment subject to assigning each job to exactly one agent and satisfying resource constraints for each agent, where s resources $K = \{1, 2, \ldots, s\}$ are considered. Assigning job j to agent i incurs a cost of c_{ij} and consumes an amount a_{ijk} of resource for each

$k \in K$, whereas the total amount of the resource k available at agent i is b_{ik}. An assignment is a mapping σ: $J \to I$, where $\sigma(j) = i$ means that job j is assigned to agent i. Then the *multi-resource generalized assignment problem* (MRGAP) is formulated as follows:

$$\begin{array}{lll} \text{minimize} & cost(\sigma) = \sum_{j \in J} c_{\sigma(j),j} & \\ \text{subject to} & \sum_{\substack{j \in J \\ \sigma(j)=i}} a_{ijk} \le b_{ik}, \quad \forall i \in I \text{ and } \forall k \in K. \end{array} \tag{1}$$

MRGAP is known to be NP-hard, and the (supposedly) simpler problem of judging the existence of a feasible solution for GAP (i.e., MRGAP with $s = 1$) is NP-complete, since the partition problem can be reduced to MRGAP with $m = 2$ and $s = 1$.

3 Algorithm

Our algorithm, called TS-CS (tabu search with chained shift neighborhood), is an extension of local search. Local search starts from an initial solution σ and repeatedly replaces σ with a better solution in its *neighborhood* $N(\sigma)$ until no better solution is found in $N(\sigma)$. The resulting solution σ is *locally optimal* in the sense that no better solution exists in its neighborhood. Shift and swap neighborhoods N_{shift} and N_{swap} are usually used in local search methods for GAP, where $N_{\text{shift}}(\sigma) = \{\sigma' \mid \sigma' \text{ is obtained from } \sigma \text{ by changing the assignment of one job}\}$, and $N_{\text{swap}}(\sigma) = \{\sigma' \mid \sigma' \text{ is obtained from } \sigma \text{ by exchanging the assignments of two jobs}\}$. In addition to these standard neighborhoods, our algorithm uses a chained shift neighborhood, which consists of solutions obtainable by certain sequences of shift moves. The chained shift neighborhood $N_{\text{chain}}(\sigma)$ is the set of solutions σ' obtainable from σ by shifting l ($l = 2, 3, \ldots, n$) jobs $j_1, j_2, \ldots, j_l$ simultaneously, in such a way that satisfies

$$\begin{aligned} \sigma'(j_r) &= \sigma(j_{r-1}), \quad r = 2, 3, \ldots, l \\ \sigma'(j_1) &= \sigma(j_l). \end{aligned}$$

In other words, for $r = 2, 3, \ldots, l$, job j_r is shifted from agent $\sigma(j_r)$ to agent $\sigma(j_{r-1})$ after ejecting job j_{r-1}. This is based on the idea of ejection chains by Glover [6]. Since the size of such a neighborhood can become exponential, we carefully limit its size by utilizing improvement graphs [1, 2]. Since $|N_{\text{shift}}| \le |N_{\text{swap}}| \le |N_{\text{chain}}|$ holds, N_{swap} is searched only if N_{shift} does not contain an improving solution, and N_{chain} is searched only if $N_{\text{shift}} \cup N_{\text{swap}}$ does not contain an improving solution unless otherwise stated.

When the search visits the infeasible region, we evaluate the solutions by an objective function penalized by infeasibility:

$$pcost(\sigma) = cost(\sigma) + \sum_{\substack{i \in I \\ k \in K}} \alpha_{ik} p_{ik}(\sigma), \tag{2}$$

where $p_{ik}(\sigma) = \max\left\{0, \sum_{j \in J, \sigma(j)=i} a_{ijk} - b_{ik}\right\}$. The parameters α_{ik} (> 0) are adaptively controlled during the search by using an algorithm similar to the method in [10].

Whenever the local search stops at a locally optimal solution σ_{lopt}, it resumes from an initial solution generated by the following rule. We keep a solution σ_{seed}, which is initially generated randomly, and is replaced with $\sigma_{\rm lopt}$ if $pcost(\sigma_{\rm lopt}) \leq pcost(\sigma_{\rm seed})$ holds (the most recent values for α_{ik} are used in $pcost$). Then we choose as the initial solution the solution in $N_{\rm shift}(\sigma_{\rm seed}) \setminus T$ with the smallest $pcost$, where T is the set of solutions already generated with shift moves from the current $\sigma_{\rm seed}$. Then the local search starts from the search in $N_{\rm swap}$, i.e., the search in $N_{\rm shift}$ is forbidden until an improved solution is found. This strategy is confirmed to be effective to avoid cycling of short period [10].

4 Computational Results

We compared the proposed algorithm TS-CS with the following three algorithms: (1) tabu search without chained shift neighborhood (denoted TS-noCS), (2) a general problem solver for the weighted constraint satisfaction problem proposed in [9] (denoted TS-WCSP), and (3) a commercial exact solver CPLEX 6.5 (denoted CPLEX). Note that TS-noCS is the same as TS-CS except that it does not use the chained shift neighborhood. All the algorithms were coded in C and run on a workstation Sun Ultra 2 Model 2300 (two UltraSPARC II 300MHz processors with 1 GB memory), where the computation was executed on a single processor.

Test instances were generated randomly by using benchmark instances for GAP. (We use GAP instances for $s = 1$.) There are five types of benchmark instances of GAP called types A, B, C, D and E [3, 7]. Out of these, we use three types C, D and E, since the other two are too easy to see differences among the tested algorithms. Types D and E are somewhat harder than type C, since c_{ij} and a_{ij1} are inversely correlated. We tested 18 instances of types C, D and E with n up to 200. Among them, types C and D instances were taken from OR-Library,[1] and type E instances were generated by ourselves, and are available at our web site.[2] For each of these GAP instances, we generated MRGAP instances by setting $a_{ijk} = 3a_{ij1}/4 + \gamma a_{ij1}/2$ for each $k = 2, 3, \ldots, s$ as in [5], where γ is a random number from [0, 1]. The generated MRGAP instances are available at our web site.[3]

Tables 1, 2 and 3 show the results of TS-CS, TS-noCS, TS-WCSP and CPLEX, where the time limit is set to 300 (resp., 600) seconds for each instance with $n = 100$ (resp., 200). Columns "best" show the objective values of the best solutions obtained by the algorithms within the time limit, and columns "TTB" (time to best) show the CPU seconds when the best solutions were found for the first time. Columns "LB" show the lower bounds on the optimal values, where the mark "†" means the value is optimal. (Most of these lower bounds were obtained by CPLEX, where the time limit was set to 3600 seconds. Some optimal values for GAP (i.e., $s = 1$) were found by Nauss [8], and some LBs for GAP were reported in [10], which were found by solving a Lagrangian relaxation problem. If $s \geq 2$, then optimal values and LBs were found by CPLEX.) In the tables, each "∗" mark represents that the best cost is attained, and "—" means that no feasible solution was found. The average of "LB," "best"and "TTB" for $n = 100$ and 200, respectively, are also shown.

From the tables, we can observe the following.

[1] URL of OR-Library: http://mscmga.ms.ic.ac.uk/jeb/orlib/gapinfo.html
[2] URL of our web site for GAP instances: http://www-or.amp.i.kyoto-u.ac.jp/˜yagiura/gap/
[3] URL of our web site for MRGAP instances: http://www-or.amp.i.kyoto-u.ac.jp/˜yagiura/mrgap/

Table 1. Results for type C instances

n	m	s	LB	TS-CS best	TS-CS TTB	TS-noCS best	TS-noCS TTB	TS-WCSP best	TS-WCSP TTB	CPLEX best	CPLEX TTB
100	5	1	†1931	*1931	1.25	*1931	0.61	1933	30.00	*1931	2
100	5	2	†1933	*1933	2.39	*1933	58.54	*1933	15.68	*1933	0
100	5	4	†1943	*1943	172.35	*1943	197.17	1944	91.65	*1943	19
100	5	8	†1950	*1950	61.19	*1950	185.08	1956	179.87	*1950	26
100	10	1	†1402	*1402	5.49	*1402	28.92	*1402	15.87	*1402	9
100	10	2	†1409	*1409	133.34	1410	74.06	1411	261.89	*1409	35
100	10	4	†1419	*1419	37.40	*1419	53.13	*1419	93.59	*1419	38
100	10	8	†1435	1436	147.34	1440	208.26	*1435	227.31	*1435	271
100	20	1	†1243	1245	55.48	1245	43.30	1245	140.22	*1243	8
100	20	2	†1250	1251	45.39	1252	46.16	1253	202.30	*1250	5
100	20	4	†1254	1257	186.09	1256	294.35	1258	135.77	*1254	30
100	20	8	†1267	1269	205.60	1275	261.05	*1267	11.86	1272	199
average			1536.3	1537.1	87.78	1538.0	120.89	1538.0	117.17	1536.8	53.5
200	5	1	†3456	*3456	170.10	3458	5.91	3460	107.54	*3456	35
200	5	2	†3461	*3461	47.64	*3461	219.14	3462	180.82	*3461	17
200	5	4	†3466	*3466	167.53	*3466	196.33	3469	389.58	*3466	173
200	5	8	†3473	*3473	530.30	*3473	447.35	3478	294.82	3474	56
200	10	1	†2806	2807	46.23	2808	156.07	2811	53.63	*2806	249
200	10	2	†2811	*2812	316.68	2813	167.54	*2812	352.65	*2812	37
200	10	4	†2819	2821	399.99	2821	444.46	2823	151.04	*2819	347
200	10	8	2833	*2837	344.24	2842	381.29	2842	480.93	2842	134
200	20	1	†2391	2393	369.54	2399	136.93	2394	31.13	*2391	296
200	20	2	†2397	*2398	313.10	2400	155.53	2403	348.75	*2398	175
200	20	4	2408	*2409	430.56	2416	165.83	2415	15.17	2415	115
200	20	8	2415	2422	47.29	2424	419.02	2423	176.03	*2419	451
average			2894.7	2896.3	265.27	2898.4	241.28	2899.3	215.17	2896.6	173.8

- The performance of algorithm TS-CS is better than TS-noCS especially for types D and E instances. This indicates that incorporating the chained shift neighborhood is effective for hard instances.

- The performance of TS-CS is better than TS-WCSP and CPLEX. CPLEX is very effective for type C instances; however, TS-CS is much better for types D and E instances.

5 Conclusion

In this paper, we considered the multi-resource generalized assignment problem and proposed a tabu search algorithm in which a sophisticated neighborhood called the chained shift neighborhood is used. It was confirmed through computational comparisons on benchmark instances that the method is effective, especially for type D and E instances, which are known to be very difficult.

Table 2. Results for type D instances

n	m	s	LB	TS-CS best	TS-CS TTB	TS-noCS best	TS-noCS TTB	TS-WCSP best	TS-WCSP TTB	CPLEX best	CPLEX TTB
100	5	1	†6353	*6357	109.87	6359	209.92	6370	115.87	6358	43
100	5	2	6352	*6359	136.10	6371	53.42	6380	106.03	6360	27
100	5	4	6362	*6379	207.25	6381	179.95	6404	297.04	6386	172
100	5	8	6388	*6425	67.89	6428	239.62	6500	264.45	6428	244
100	10	1	6342	*6361	246.00	6377	79.09	6418	192.77	6381	132
100	10	2	6340	*6378	174.39	6405	168.45	6411	241.94	6419	88
100	10	4	6361	*6430	274.92	6438	184.59	6516	126.63	6468	166
100	10	8	6388	*6478	241.80	6520	232.30	6679	255.64	6528	83
100	20	1	6177	*6231	194.94	6270	217.92	6305	204.90	6280	60
100	20	2	6165	*6261	253.83	6305	56.77	6389	223.84	6316	19
100	20	4	6182	*6321	277.59	6331	178.93	6529	58.70	6406	148
100	20	8	6206	6482	234.49	*6481	270.78	6736	34.08	6588	68
average			6301.3	6371.8	201.59	6388.8	172.65	6469.8	176.82	6409.8	104.2
200	5	1	12741	12751	191.63	12756	81.33	12760	87.70	*12750	62
200	5	2	12751	*12766	441.53	12772	110.91	12778	171.48	*12766	534
200	5	4	12745	12775	178.92	12778	151.80	12799	78.63	*12762	286
200	5	8	12755	12805	527.78	12809	292.53	12844	348.73	*12787	432
200	10	1	12426	12463	330.52	12482	555.38	12478	279.46	*12457	27
200	10	2	12431	*12477	476.07	12518	512.70	12533	590.21	12482	578
200	10	4	12432	*12496	471.01	12552	540.32	12586	548.07	12532	112
200	10	8	12448	*12571	481.10	12592	513.42	12812	346.33	12577	209
200	20	1	12230	*12312	230.72	12365	346.77	12409	445.56	12393	297
200	20	2	12227	*12332	597.11	12384	568.76	12442	587.38	12425	21
200	20	4	12237	*12396	576.45	12488	386.71	12605	573.62	12472	425
200	20	8	12254	*12485	337.27	12650	481.50	12918	250.78	12548	132
average			12473.1	12552.4	403.34	12595.5	378.51	12663.7	359.00	12579.3	259.6

References

[1] R.K. Ahuja, O. Ergun, J.B. Orlin, and A.P. Punnen, "A survey of very large-scale neighborhood search techniques," *Discrete Applied Mathematics*, 123 (2002) 75–102.

[2] R.K. Ahuja, J.B. Orlin, and D. Sharma, "Very large-scale neighborhood search," *International Transactions in Operational Research*, 7 (2000) 301–317.

[3] P.C. Chu and J.E. Beasley, "A genetic algorithm for the generalized assignment problem," *Computers and Operations Research*, 24 (1997) 17–23.

[4] J.A. Díaz and E. Fernández, "A tabu search heuristic for the generalized assignment problem," *European Journal of Operational Research*, 132 (2001) 22–38.

[5] B. Gavish and H. Pirkul, "Algorithms for the multi-resource generalized assignment problem," *Management Science*, 37 (1991) 695-713.

[6] F. Glover, "Ejection chains, reference structures and alternating path methods for traveling salesman problems," *Discrete Applied Mathematics*, 65 (1996) 223–253.

Table 3. Results for type E instances

n	m	s	LB	TS-CS best	TS-CS TTB	TS-noCS best	TS-noCS TTB	TS-WCSP best	TS-WCSP TTB	CPLEX best	CPLEX TTB
100	5	1	†12681	*12681	54.08	12682	124.34	12753	2.72	*12681	44
100	5	2	†12692	*12692	120.93	*12692	210.25	12727	28.90	*12692	212
100	5	4	†12810	12812	104.03	12812	83.04	12893	79.48	*12810	24
100	5	8	†12738	*12738	53.98	12739	210.64	12876	152.79	12749	61
100	10	1	†11577	*11577	90.03	11584	91.42	11712	209.98	11584	200
100	10	2	†11582	*11587	179.65	11604	290.33	11665	41.33	11612	80
100	10	4	11636	*11676	289.32	11689	106.93	11864	102.15	11753	257
100	10	8	11619	*11701	260.90	11756	230.60	11836	86.49	11739	258
100	20	1	†8436	*8447	142.39	8488	110.94	8655	144.76	8565	72
100	20	2	10123	*10150	207.09	10219	247.15	10471	79.65	10251	234
100	20	4	10794	*11029	160.57	11075	82.28	11271	117.97	11443	78
100	20	8	11224	*11610	265.33	11817	285.93	11957	143.07	12458	291
average			11492.7	11558.3	160.69	11596.4	172.82	11723.3	99.11	11694.8	150.9
200	5	1	†24930	24933	43.21	24933	399.77	25002	101.77	*24930	18
200	5	2	†24933	24936	430.29	*24933	520.42	25024	519.47	*24933	131
200	5	4	24990	*24999	537.27	25017	555.53	25091	280.81	25003	228
200	5	8	†24943	24950	192.28	24970	449.00	25090	271.98	*24943	68
200	10	1	†23307	*23312	411.12	23326	30.46	23414	494.72	23321	140
200	10	2	23310	*23317	436.01	23333	51.82	23538	303.94	23325	386
200	10	4	23344	*23363	376.65	23412	56.24	23628	83.62	23543	309
200	10	8	23339	23412	198.96	*23410	357.69	23714	550.29	23744	56
200	20	1	†22379	*22386	178.16	22455	519.64	22815	207.84	22457	185
200	20	2	22387	*22408	333.97	22459	121.13	22834	377.76	22558	302
200	20	4	22395	*22439	462.53	22517	566.93	22990	413.64	22782	238
200	20	8	22476	*22614	317.52	—	—	23057	394.42	23482	85
average			23561.1	23589.1	326.50	—	—	23849.8	333.36	23751.8	178.8

[7] M. Laguna, J.P. Kelly, J.L. González-Velarde and F. Glover, "Tabu search for the multilevel generalized assignment problem," *European Journal of Operational Research*, 82 (1995) 176–189.

[8] R.M. Nauss, "Solving the generalized assignment problem: An optimizing and heuristic approach," *INFORMS Journal on Computing*, to appear.

[9] K. Nonobe and T. Ibaraki, "An improved tabu search method for the weighted constraint satisfaction problem," *INFOR*, 39 (2001) 131–151.

[10] M. Yagiura, T. Ibaraki and F. Glover, "An ejection chain approach for the generalized assignment problem," *INFORMS Journal on Computing*, to appear.

[11] M. Yagiura, T. Ibaraki and F. Glover, "A path relinking approach for the generalized assignment problem," *Proc. International Symposium on Scheduling*, Japan, June 4–6 (2002) 105–108.