Computers & Industrial Engineering 93 (2016) 99–109

Contents lists available at ScienceDirect

Computers & Industrial Engineering

journal homepage: www.elsevier.com/locate/caie

An efficient tabu search for solving the uncapacitated single allocation hub location problem

Roya Abyazi-Sani, Reza Ghanbari *

Faculty of Mathematical Sciences, Department of Applied Mathematics, Ferdowsi University of Mashhad, Mashhad, Iran

ARTICLE INFO

Article history:
Received 5 January 2015
Received in revised form 29 October 2015
Accepted 26 December 2015
Available online 2 January 2016

Keywords:
Hub location problems
Uncapacitated single allocation hub location problem
Tabu search
Metaheuristic

ABSTRACT

We here propose an efficient tabu search (TS) for solving the uncapacitated single allocation hub location problem. To decrease the computational time, in the proposed tabu search, some new tabu rules are considered. Also, to compute the changes in the objective function's value in each move, some new results are given. The performance of the proposed tabu search is compared with a recently proposed tabu search (Silva & Cunha, 2009) on all standard ORLIB instances (CAB and AP data sets), modified AP data set and finally on four large instances with 300 and 400 nodes proposed by Silva and Cunha. The numerical experiments show that the proposed tabu search can find all optimal solutions of CAB data and the best known solution of other standard test problems in less computational time than Silva and Cunha's tabu search. Also, the proposed tabu search can improve the best known solutions for some standard test problems.

1. Introduction

Hub location problems are widely applied in postal networks (Ebery, Krishnamoorthy, Ernst, & Boland, 2000; Kuby & Gray, 1993), airlines (Aykin, 1995; Bania, Bauer, & Zlatoper, 1998; Karimi & Bashiri, 2011; Yang, 2009), telecommunication networks (Carello, Croce, Ghirardi, & Tadel, 2004; Helme & Magnanti, 1989; Kim & OKelly, 2009), transportation systems (Aversa, Botter, Haralambides, & Yoshizaki, 2005; Cunha & Silva, 2007; Don, Harit, English, & Whisker, 1995), emergency services (Campbell, 1994; Hakimi, 1964), etc. Generally, hub location problems are used in systems which have many origins and destinations, and direct link between each o–d[1] is impossible or expensive. In these location–allocation problems, flows consolidate at hub nodes. Thus, all transportation become possible through hubs and also, costs are reduced because of the economical advantages of the scale.

In hub location problems, usually, hubs are fully interconnected, transferring between hubs includes a discount factor, and direct links between non-hub nodes are not allowed. In addition, in most of hub location problems, hubs are a subset of discrete nodes, however there are some researches on continuous hub location problems (see Aykin & Brown, 1992; Campbell, 1990; OKelly, 1992).

Allocating non-hub nodes to hubs is done in two ways: single in which each non-hub node is allocated to a single hub (see Contreras, Fernández, & Marín, 2009; Labbé & Yaman, 2004; Labbé, Yaman, & Gourdin, 2005) and multiple in which each non-hub node can use more than one hub (see Contreras, Cordeau, & Laporte, 2012; Hamacher, Labbé, Nickel, & Sonneborn, 2004; Marín, 2005; Mayer & Wagner, 2002). Nevertheless, some researches contain both types of the allocation (see Campbell, 1994; OKelly, Bryan, Skorin-Kapov, & Skorin-Kapov, 1996; Skorin-Kapov, Skorin-Kapov, & OKelly, 1996).

Hub location problems can be classified based on allocation type, number of hubs, or capacitated (or uncapacitated) hubs, etc. Also, some other hub location problems are introduced in the past few years, such as: hub arc location (see Campbell, Ernst, & Krishnamoorthy, 2005a; Campbell, Ernst, & Krishnamoorthy, 2005b; Campbell, Stiehr, Ernst, & Krishnamoorthy, 2003), competitive hub location (see Eiselt & Marianov, 2009; Gelareh, Nickel, & Pisinger, 2010; Marianov, Serra, & ReVelle, 1999), dynamic hub location (see Campbell, 2010; Contreras, Cordeau, & Laporte, 2011), and also, hub location problems with stochastic elements and reliability considerations (see Contreras, Cordeau, & Laporte, 2011; Lium, Crainic, & Wallace, 2009; Sim, Lowe, & Thomas, 2009; Yang, 2009). Some overviews on hub location problems can be found in Farahani, Hekmatfar, Arabani, and Nikbakhsh (2013), Alumur and Kara (2008), Campbell, Ernst, and Krishnamoorthy (2002), and Klencewicz (1998). Here, we give a heuristic for solving the uncapacitated single allocation hub location problem (USAHLP). In USAHLP, each non-hub node is allocated

* Corresponding author. Tel.: +98 51 38805658; fax: +98 51 38828606.
E-mail addresses: r.abyazi@gmail.com (R. Abyazi-Sani), rghanbari@um.ac.ir (R. Ghanbari).

[1] Origin–destination.

http://dx.doi.org/10.1016/j.cie.2015.12.028
0360-8352/

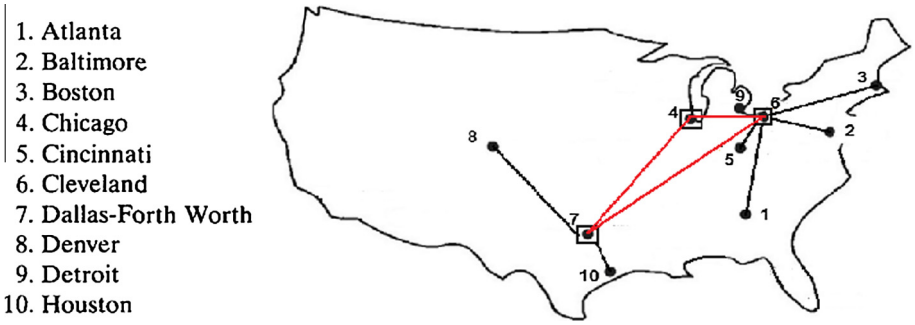

Fig. 1. A sample of CAB data set ($n = 10$).

to exactly one hub, there is no capacity constraint and establishing each hub contains a fixed cost. The aim of USAHLP is to determine the number of hubs, the location of hubs and the allocation of non-hub nodes to hubs, in order to minimize the sum of costs (transportation and establishing costs). There are various formulations for the USAHLP, such as the quadratic programming formulation (Abdinnour-Helm & Venkataramanan, 1998; OKelly, 1992) and the integer linear formulation (Campbell, 1994).

Since USAHLP is an NP-hard problem (Love, Morris, & Wesolowsky, 1988; Naeem & Ombuki-Berman, 2010; OKelly, 1987), many metaheuristics were proposed for solving the USAHLP. OKelly (1992) proposed a two-step procedure in which a good upper bound on the solution value is estimated based on three different heuristics, and then a thigh lower bound on the solution is computed. Abdinnour-Helm and Venkataramanan (1998) proposed an exact branch-and-bound algorithm and a genetic algorithm (GA) for USAHLP. The branch-and-bound algorithm can solve problems up to fifteen nodes, while the GA solves larger problems over the CAB data set (OKelly, 1987) and the random proposed data set, efficiently. Abdinnour-Helm (1998) proposed a hybrid procedure based on genetic algorithm and tabu search (GATS). In this algorithm, hubs were chosen by GA and then, distance-based allocations, in which each node was allocated to the nearest hub using procedure proposed by OKelly (1987), were improved by the tabu search. GATS solved CAB instances, efficiently. Topcuoglu, Corut, Ermis, and Yilmaz (2005) proposed a GA for solving both location and allocation parts of the USAHLP. This algorithm outperformed the GATS proposed by Abdinnour-Helm (1998) on the CAB data set and AP data set (Ernst & Krishnamoorthy, 1996) in both solution values and running times. Also, Cunha and Silva (2007) proposed a hybrid heuristic procedure for the USAHLP. In this procedure a GA is applied to determine the location of the hubs, and a SA is used to reallocate the distance-based allocation. The procedure is better than the GATS (Abdinnour-Helm, 1998) over the CAB data set. Chen (2007) proposed SATLUHLP for solving USAHLP. In SATLUHLP, two approaches were proposed to find an upper bound for the number of hubs, and hybrid heuristic based on SA, TS and an improvement procedure were used to solve the location and the allocation parts of the problem. Computational results showed that SATLUHLP outperformed the GA proposed by Topcuoglu et al. (2005) over CAB and AP data set. Filipović, Kratica, Tošić, and Dugošija (2009) proposed a GA-inspired heuristic method for USAHLP. They also proposed an exact total enumeration method for solving the allocation part of the problem. The GA-inspired heuristic outperformed the GA proposed by Topcuoglu et al. (2005) and SATLUHLP proposed by Chen (2007) over the AP data sets. Silva and Cunha (2009) proposed two efficient heuristics called MSTS and HubTS. Multi start tabu search heuristics (MSTS1, 2 and 3), in which several initial solutions were improved by a TS. How to construct

Table 1
Results of TSUSAHLP and HubTS – CAB ($n = 10, 15$).

α	F_k	$n = 10$					$n = 15$				
		TSUSAHLP				HubTS	*TSUSAHLP*				HubTS
		n_H	K_{opt}	t_{opt}	t	t	n_H	K_{opt}	t_{opt}	t	t
0.2	100	3	1	0.008	0.136	0.423	5	1	0.010	0.262	**2.197**
	150	2	2	0.024	0.129	0.197	4	2	0.032	0.259	**1.763**
	200	2	1	0.011	0.109	0.246	2	2	0.028	0.232	0.400
	250	2	2	0.020	0.114	0.251	2	1	0.018	0.236	0.357
0.4	100	3	2	0.018	0.130	0.434	4	1	0.014	0.272	**1.213**
	150	2	0	0.003	0.122	0.180	3	2	0.034	0.250	0.727
	200	2	0	0.003	0.121	0.217	2	2	0.035	0.258	0.392
	250	2	0	0.003	0.179	0.260	1	0	0.007	0.241	0.147
0.6	100	2	0	0.003	0.117	0.160	3	1	0.013	0.238	0.664
	150	2	0	0.004	0.152	0.201	2	1	0.015	0.219	0.406
	200	1	2	0.017	0.144	0.065	1	0	0.007	0.220	0.190
	250	1	7	0.104	0.155	0.069	1	0	0.007	0.201	0.183
0.8	100	2	0	0.003	0.133	0.187	2	5	0.094	0.251	0.432
	150	1	1	0.007	0.122	0.086	1	0	0.007	0.228	0.264
	200	1	4	0.033	0.114	0.087	1	0	0.007	0.213	0.300
	250	1	4	0.033	0.114	0.109	1	0	0.007	0.232	0.427
1.0	100	1	1	0.006	0.121	0.100	1	1	0.010	0.214	0.234
	150	1	1	0.007	0.117	0.112	1	0	0.007	0.220	0.381
	200	1	1	0.008	0.119	0.118	1	0	0.007	0.220	0.390
	250	1	1	0.008	0.118	0.113	1	0	0.007	0.202	0.365

The running times more than one second are bold.

Table 2
Results of TSUSAHLP and HubTS – CAB (n = 20, 25).

α	F_k	$n = 20$					$n = 25$				
		TSUSAHLP				HubTS	*TSUSAHLP*				HubTS
		n_H	K_{opt}	t_{opt}	t	t	n_H	K_{opt}	t_{opt}	t	t
0.2	100	5	1	0.018	0.373	**4.879**	4	1	0.078	0.912	**5.041**
	150	3	1	0.018	0.403	**1.358**	3	1	0.045	**1.076**	**2.400**
	200	3	1	0.044	0.410	**1.624**	3	1	0.050	0.856	**2.552**
	250	3	1	0.016	0.414	**3.459**	2	4	0.303	0.772	**1.688**
0.4	100	4	4	0.118	0.413	**2.934**	4	1	0.030	0.546	**7.109**
	150	3	1	0.044	0.390	**1.475**	3	5	0.257	0.545	**3.187**
	200	2	0	0.013	0.407	0.659	2	4	0.201	0.574	**2.501**
	250	2	0	0.013	0.375	0.802	2	2	0.136	0.586	**1.966**
0.6	100	4	1	0.018	0.398	**4.649**	3	2	0.073	0.582	**2.490**
	150	2	0	0.013	0.389	**1.114**	3	2	0.087	0.640	**2.876**
	200	2	0	0.013	0.397	0.811	2	9	0.553	0.687	**3.980**
	250	1	4	0.129	0.388	0.290	2	1	0.075	0.748	**0.983**
0.8	100	2	0	0.013	0.394	**1.077**	3	2	0.130	0.961	**3.308**
	150	2	0	0.013	0.401	**1.129**	2	5	0.386	0.969	**3.320**
	200	1	3	0.088	0.339	0.340	1	4	0.267	0.992	**1.131**
	250	1	2	0.070	0.359	0.352	1	3	0.237	**1.034**	**1.371**
1.0	100	2	1	0.040	0.414	**1.221**	3	3	0.204	0.968	**4.825**
	150	1	0	0.017	0.535	0.302	1	2	0.114	0.483	0.544
	200	1	0	0.018	0.505	0.353	1	2	0.102	0.487	0.543
	250	1	0	0.018	0.532	0.411	1	1	0.057	0.460	0.562

The running times more than one second are bold.

the initial solution was the main difference between MSTS1, 2 and 3. In HubTS, tabu searches were applied for both location and allocation parts of USAHLP. Their numerical experiments showed that HubTS was better than three variants of MSTS in solution values. MSTS3 and HubTS were more effective than SATLUHLP proposed by Chen (2007) in both solution values and running times. Naeem and Ombuki-Berman (2010) proposed two GAs for solving USAHLP that are better than GATS proposed by Abdinnour-Helm (1998) and the GA proposed by Topcuoglu et al. (2005) over the CAB data set. Also, their proposed GAs were comparable with the SATLUHLP proposed by Chen (2007), MSTS and HubTS proposed by Silva and Cunha (2009) over the AP data set. Han-yi (2010) proposed a GA by considering upper bound of the ratio of the number of hubs to the number of nodes. When the discount factor was low, this algorithm had better behaviour than the GA proposed by Topcuoglu et al. (2005). Marić, Stanimirović, and Stanojević (2013) proposed a memetic algorithm (MA) for solving USAHLP. Two efficient local searches were used to improve both the location and the allocation part of the problem. Computational results showed that the MA outperforms the GA proposed by Filipović et al. (2009). Bailey, Ornbuki-Berrnan, and Asobiela (2013) used a PSO framework for USAHLP. They compared their proposed discrete PSO, DPSO, with GATS proposed by Abdinnour-Helm (1998), the GA proposed by Topcuoglu et al. (2005), the SATLUHLP proposed by Chen (2007), and MSTS and HubTS proposed by Silva and Cunha (2009). The DPSO matches or beats performance of these methods in terms of solution quality over the CAB and AP data sets. Ting and Wang (2014) proposed a threshold accepting (TA) algorithm for solving USAHLP. They compared their proposed TA results with the GA proposed by Topcuoglu et al. (2005), the SATLUHLP proposed by Chen (2007), GASA proposed by Cunha and Silva (2007), and HubTS proposed by Silva and Cunha (2009). The results show that their algorithm outperforms GASA, the GA, SATLUHLP and HubTS in terms of solution quality and computational time.

Tabu search is one of the most effective and efficient techniques for solving optimization problems. As Jaeggi et al. mentioned in Jaeggi, Parks, Kipouros, and Clarkson (2008), a true performance comparison between algorithms is difficult because of the lack of suitable set of benchmark problems. However a wide range of fields applied tabu search, because the obtained solutions often significantly surpass the best solutions previously found by other approaches. It has been justified by Pirim, Eksioglu, and Bayraktar (2008) in which around 150 articles are investigated. So, we here propose a tabu search, TSUSAHLP, for solving USAHLP. TSUSAHLP is inspired by the effective tabu search proposed by Sun (2006) for solving uncapacitated facility location problem (UFLP) (Korte & Vygen, 2008) which is a successful algorithm for solving uncapacitated facility location problem (UFLP) and it can easily be generalized to get a suitable procedure for USAHLP. In TSUSAHLP, similar to Sun's tabu search, the objective function's value (cost value) is updated from its previous value, all over the algorithm. Despite TSUSAHLP is inspired by Sun's algorithm, there are some modifications and differences in TSUSAHLP. In TSUSAHLP, we use an adoptive method for tabu tenure whereas Sun's algorithm did not. Also, we propose a new candidate list for moves in TSUSAHLP. As far as we know, the proposed candidate list has never been used before in heuristics for USAHLP. In addition, we could not use the proposed formulas by Sun in TSUSAHLP, directly. Therefore, we here give some new results for computing the changes in the objective function's value when a move is done. Also, based on computational experiments, in TSUSAHLP, we do not use the medium scale memory.

Here, it is provided a comprehensive comparison between our proposed algorithm and HubTS proposed by Silva and Cunha (2009). In addition, the solution quality is compared with the GA proposed by Naeem and Ombuki-Berman (2010), the MA proposed by Marić et al. (2013), the discrete PSO proposed by Bailey et al. (2013), and the TA proposed by Ting and Wang (2014). Computational results on CAB data set, AP data set, modified AP data set (Silva & Cunha, 2009), and four large instances based on AP data set (Silva & Cunha, 2009) show that not only TSUSAHLP can find the optimal or the best known solutions for these instances in less CPU times but also, it improves the best known solutions for large instances.

The rest of paper is organized as follows. In Section 2, we give a combinatorial formulation of USAHLP. TSUSAHLP is described in Section 3. Some new results on the computation of the objective function's value are given in Section 4. Section 5 contains the computational experiments. We conclude in Section 6.

Table 3
Comparison with other algorithms – AP data set.

n	TSUSAHLP solution	K_{opt}	t_{opt}	t	HubTS solution	t	GA-2 List, MA, DPSO, TA
10 L	opt	1	0.017	0.151	opt	0.335	opt
20 L	opt	0	0.012	0.412	opt	0.761	opt
25 L	opt	2	0.073	0.627	opt	1.582	opt
40 L	opt	1	0.082	1.527	opt	6.863	opt
50 L	opt	3	0.658	2.286	opt	11.070	opt
100 L	opt	1	0.496	11.991	opt	58.831	opt
200 L	**233802.98**	1	3.435	65.193	233803.02	379.379	**233802.98**
10 T	opt	3	0.028	0.160	opt	0.366	opt
20 T	opt	1	0.018	0.633	opt	1.485	opt
25 T	opt	3	0.185	0.833	opt	0.752	opt
40 T	opt	0	0.069	2.316	opt	2.704	opt
50 T	opt	6	1.701	3.325	opt	3.197	opt
100 T	opt	0	0.425	19.210	opt	24.060	opt
200 T	**272188.11**	7	53.171	**79.513**	272237.78	**463.994**	**272188.11**

The best known solution or the maximum running time are appeared in Bold.

2. Combinatorial formulation

Let $N = \{1, 2, \ldots, n\}$ be the set of nodes, C_{ij} be the transportation cost per unit of flow between nodes i and j, and W_{ij} be the amount of flow between nodes i and j. The flow W_{ij} is transferred from node i to node j via nodes $l(i)$ (the hub allocated to node i) and $l(j)$ (the hub allocated to node j). So, the cost per unit of flow from node i to node j is $\chi C_{il(i)} + \alpha C_{l(i)l(j)} + \delta C_{l(j)j}$, where factors χ, α and δ are the collection, transfer and distribution cost coefficients, respectively (see Ernst & Krishnamoorthy, 1996; OKelly, 1987). The fixed cost for establishing a hub at the node k is F_k. The aim of the USAHLP is to determine the set S, a subset of N as hubs, and an allocating function $l : N \to S$ which minimizes the objective function. Let $S' = N - S$, so the objective function value assigned to (S, l) is

$$f(S,l) = \sum_{i\in S'}\sum_{j\in S'} W_{ij}(\chi C_{il(i)} + \alpha C_{l(i)l(j)} + \delta C_{l(j)j}) + \sum_{i\in S}\sum_{j\in S'} W_{ij}(\alpha C_{il(j)} + \delta C_{l(j)j}) + \sum_{i\in S'}\sum_{j\in S} W_{ij}(\chi C_{il(i)} + \alpha C_{l(i)j}) + \sum_{i\in S}\sum_{j\in S} W_{ij}(\alpha C_{ij}) + \sum_{k\in S} F_k. \quad (1)$$

In other words, (S^*, l^*) is an optimal solution when $(S^*, l^*) = \arg\min_{S\subseteq N} f(S, l)$.

A sample of CAB data set ($n = 10$, $\chi = 1$, $\alpha = 0.2$, $\delta = 1$ and given matrixes F, W and C) is illustrated in Fig. 1 as a simple example for USAHLP. This sample contains 10 cities of U.S. airline passenger. In this example, (S^*, l^*), the optimal solution is $S^* = \{4, 6, 7\}$, $l^*(4) = 4$, $l^*(\{1, 2, 3, 5, 6, 9\}) = 6$, and $l^*(\{7, 8, 10\}) = 7$. In Fig. 1, hubs are shown with square, allocations with black line, and hub connectivity with red lines. As transportation in hub location problems is possible only via hubs, passengers just can travel through determined hubs. For instance, if airline passengers want to travel from Boston to Denver, firstly they must go to Cleveland. After that, they must have a stopped in Dallas-Fort-Worth, and then they could fly to Denver.

3. Tabu search for USAHLP (TSUSAHLP)

Tabu search (TS) is based on the neighbourhood search in which the most recent moves are set in the tabu list to avoid local optimal. In TS, not only cycling is avoided, but also a diversified search in the solution space is prepared. TS is very efficient because it usually searches only a small subset of feasible solutions that contains the optimal solution. The modern form of tabu search was originally developed by Glover (1986), and a comprehensive account of tabu search and its recent developments was given by Glover and Laguna (1997).

Usually, TS has three memory structures: short term memory, medium term memory and long term memory. We here propose a tabu search for solving USAHLP which contains only short term memory and long term memory. Our experimental results showed that using medium term memory increases the computational time without any improvement.

Short term memory and long term memory make a cycle. Short term memory is run at the beginning of each cycle. Then, long term memory is called, and the solution constructed by the long term

Table 4
Comparison with other algorithms – modified AP data set ($\theta = 0.8$).

n	TSUSAHLP solution	K_{opt}	t_{opt}	t	HubTS solution	t	MA solution	TA solution
10 L	opt	4	0.049	0.168	opt	0.581	opt	opt
20 L	opt	5	0.171	0.430	opt	2.566	opt	opt
25 L	opt	2	0.071	0.671	opt	1.424	opt	opt
40 L	opt	0	0.052	1.549	opt	12.685	opt	opt
50 L	opt	4	0.888	2.472	opt	9.181	opt	opt
100 L	opt	3	2.228	10.747	opt	51.552	opt	opt
200 L	223704.42	1	2.567	64.670	233704.44	369.544	223704.44	223704.44
10 T	opt	1	0.004	0.123	opt	0.443	opt	opt
20 T	opt	1	0.018	0.402	opt	0.998	opt	opt
25 T	opt	1	0.028	0.626	opt	2.246	opt	opt
40 T	opt	0	0.052	1.875	opt	2.503	opt	opt
50 T	opt	1	0.164	2.775	opt	10.683	opt	opt
100 T	opt	1	0.556	17.261	opt	63.500	opt	opt
200 T	**258797.48**	6	35.906	72.681	260312.67	**939.416**	260312.67	**258797.48**

The best known solution or the maximum running time are appeared in Bold.

Table 5
Comparison with other algorithms – modified AP data set ($\theta = 0.9$).

n	TSUSAHLP solution	K_{opt}	t_{opt}	t	HubTS solution	t	MA solution	TA solution
10 L	opt	1	0.012	0.149	opt	0.308	opt	opt
20 L	opt	0	0.012	0.425	opt	0.994	opt	opt
25 L	opt	2	0.072	0.617	opt	1.639	opt	opt
40 L	opt	0	0.053	1.540	opt	14.373	opt	opt
50 L	opt	11	2.493	2.520	opt	10.470	opt	opt
100 L	opt	3	2.211	12.207	opt	57.393	opt	opt
200 L	228753.70	1	2.367	66.085	228753.70	374.954	228753.70	228753.70
10 T	opt	1	0.009	0.153	opt	0.287	opt	opt
20 T	opt	1	0.018	0.465	opt	0.973	opt	opt
25 T	opt	1	0.030	0.664	opt	1.581	opt	opt
40 T	opt	0	0.062	2.043	opt	1.818	opt	opt
50 T	opt	1	0.285	5.219	opt	9.383	opt	opt
100 T	opt	0	0.408	15.200	opt	26.281	opt	opt
200 T	**266116.32**	2	10.392	**75.255**	266275.22	461.035	266275.22	**266116.32**

The best known solution or the maximum running time are appeared in Bold.

Table 6
Comparison with other algorithms – large instances based on AP data set with $n = 300$ and 400.

n	TSUSAHLP solution	K_{opt}	t_{opt}	t	HubTS solution	t	MA solution	TA solution
300 L	**263913.15**	1	5.554	41.157	264837.88	4583.210	263964.00	264706.50
400 L	**267873.65**	3	78.110	106.336	268164.13	**8887.322**	267921.70	**267873.65**
300 T	**276023.35**	2	21.759	48.658	276047.75	3641.829	**276023.35**	**276023.35**
400 T	**284037.25**	2	48.498	**115.863**	284212.47	4664.384	**284037.24**	**284037.25**

The best known solution or the maximum running time are appeared in Bold.

memory is sent to the next cycle. The components of TSUSAHLP are introduced as follows.

3.1. Solution space

Let $N = \{1, 2, \ldots, n\}$ be the set of nodes, then the solution space is:

$$A = \{(S, l) | l : N \to S, S \subseteq N;\ l(i) = i, \forall i \in S\}, \tag{2}$$

where $l(\cdot)$ is an allocating function with respect to S.

3.2. Initial solution

To determine the set of hubs (S) for the initial solution, we here adopt DROP method (Sun, 2006) for solving USAHLP. In adopted DROP method, all of the nodes are open (hub) at first (i.e. $S = N$), then the node j is removed from the set of hubs which the associated decreasing in the objective function $f(\cdot,\cdot)$ defined in (1) is more than others. Thus, S is updated by $S \leftarrow S - \{j\}$. This procedure is repeated until there is not an improvement in the objective function's value. The distance-based rule is used to determine the allocating function (l) for each set of hubs. The details of constructing the initial solution is given in Algorithm 1. In this algorithm, Δz_j, which is calculated based on Proposition 4.2, shows the change in the objective function's value when the node j is removed from the set of hubs.

Remark 3.1. In many situations in the algorithm, we need to determine the nearest hub to each node in the algorithm. Thus, we define the auxiliary matrix R in which column j contains the nodes' index which their distance from j are sorted increasingly. It is obvious that $R_{1j} = j$ for all $j \in N$. Using R, computing and updating of $f(\cdot,\cdot)$ defined in (1) would be much faster.

Algorithm 1. Adopted DROP

Step 1: Let: $z \leftarrow \alpha \sum_{i \in N} \sum_{j \in N} C_{ij} W_{ij} + \sum_{i \in N} F_i$, $S \leftarrow N$ and $p \leftarrow |S|$.
Step 2: If $p = 1$ or $\triangle z_j > 0\ \forall k \in N$, then go to **Step 4**.
Step 3: Let: $\triangle z_{j^*} \leftarrow min\{\triangle z_j | j \in S, \triangle z_j < 0\}$, $S \leftarrow S - \{j^*\}$, $z \leftarrow z + \triangle z_{j^*}$, and go to **Step 2**.
Step 4: Let $S' = N - S$, and for each $j \in S'$ let $l(j) = R_{i^* j}$ where $i^* = \min\{i | R_{ij} \in S\}$. (see Remark 3.1)
Step 5: Return S, S' and l.

3.3. Possible moves

Here, we introduce the location moves that change the set of hubs (S), and the allocation move that modifies the allocating function (l).

3.3.1. Location moves

Two location moves are here used: The basic move that contains adding or removing a hub, and the swap move that exchanges a hub in S and a non-hub node in S'. In other words, the basic move and the swap move are moves from S to a $\overline{S}$ in $N_1(S)$ and $N_2(S)$, respectively, where $N_1(S)$ and $N_2(S)$ are defined as follows:

$$N_1(S) = \left\{\overline{S} | \overline{S} = S \cup \{i\}, i \in S',\ \text{or}\ \overline{S} = S - \{i\}, i \in S\right\}, \tag{3}$$

$$N_2(S) = \left\{\overline{S} | \overline{S} = S \cup \{i\} - \{j\}, i \in S', j \in S\right\}. \tag{4}$$

3.3.2. Allocation move

In this type of move, the allocation of a non-hub node changes from one hub to another hub. Note that the allocation

Table 7
Running times of the MA over AP data set, modified AP data set and new large AP data set with $n = 300$ and 400.

n	$\theta = 0.8$ t_{MA}	$\theta = 0.9$ t_{MA}	$\theta = 1$ t_{MA}
10 L	0.015	0.011	0.011
20 L	0.031	0.031	0.030
25 L	0.045	0.044	0.040
40 L	0.081	0.082	0.082
50 L	0.119	0.125	0.122
100 L	0.404	0.393	0.394
200 L	3.095	2.792	2.889
10 T	0.012	0.012	0.011
20 T	0.031	0.031	0.030
25 T	0.039	0.041	0.031
40 T	0.060	0.060	0.060
50 T	0.151	0.142	0.106
100 T	0.593	0.543	0.478
200 T	3.498*	3.111*	3.232
300 L	–	–	29.887*
400 L	–	–	131.245*
300 T	–	–	31.825
400 T	–	–	94.525

* The best known solution is not obtained by the MA.

Table 8
Running times of TA over the AP data set, modified AP data set and new large AP data set with $n = 300$ and 400.

n	$\theta = 0.8$ t_{TA}	$\theta = 0.9$ t_{TA}	$\theta = 1$ t_{TA}
10 L	0.00	0.00	0.00
20 L	0.03	0.02	0.02
25 L	0.04	0.04	0.04
40 L	0.34	0.28	0.20
50 L	0.60	0.45	0.47
100 L	3.18	2.28	2.22
200 L	76.12	69.02	84.05
10 T	0.00	0.00	0.00
20 T	0.02	0.02	0.01
25 T	0.04	0.04	0.02
40 T	0.11	0.11	0.12
50 T	0.50	0.49	0.32
100 T	1.98	0.39	0.45
200 T	122.92	135.10	65.41
300 L	–	–	729.10*
400 L	–	–	1340.98
300 T	–	–	286.50
400 T	–	–	749.88

* The best known solution is not obtained by the TA.

move is just defined for non-hub nodes. When the allocation move is done, the objective function's value is updated (see Proposition 4.1).

3.4. Constructing a feasible allocation for a location move

For choosing a location move for a set of hubs, S, the allocation of the neighbourhoods of S must be determined. Since finding the proper allocation for each set of hubs is time consuming, we use a good feasible allocation based on the previous allocation and using the auxiliary matrix R. If the hub i is removed from the current set of hubs (i.e. $\overline{S} = S - \{i\}$), the nodes allocated to the hub i will be allocated to the nearest hub in $\overline{S}$. Otherwise, if the node i is added to the current set of hubs (i.e. $\overline{S} = S \cup \{i\}$), the nodes which are closer to this node than to their own hubs will be allocated to the new hub i. Updating the objective function's value is done using Propositions 4.2 and 4.3, respectively. Finally, for a swap move, a hub is removed from the set of hubs and then, a non-hub is added to the set of the hubs. So, the both above procedures will be done, respectively.

It is notable that when a location move is done, each non-hub node will be allocated to the hubs. Let (S, l) be the current solution. The allocating function, l^*, for the set of hubs, S, is determined as

$$l^*(j) = \arg\min\{f(S,l) | l(j) = i, i \in S\} \quad \forall j \in S'. \tag{5}$$

3.5. Short term memory

The aim of short term memory is to find new solutions that were not seen in the most recent moves of the search. So, when the status of a node is changed then that node is placed in the tabu list for a tabu tenure.

3.5.1. Candidate list

To avoid inappropriate movement throughout our algorithm, a candidate list is used. If the distance from the non-hub node j to its nearest hub is less than d defined in (6), j is flagged.

$$d = \frac{y}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} C_{ij}, \tag{6}$$

where y is a predefined parameter. So, all unflagged nodes are in the candidate list. As far as we know, this candidate list has never been used before in other tabu searches for location problems.

3.5.2. Tabu list

When the status of a node is changed, the node is added to the tabu list for a tabu tenure. The computational study showed that the constant or random tabu tenure used in Sun's algorithm decreases the computational time, but the quality of the solution is usually decreased. Therefore, we use an adaptive method for tabu tenure. We tend to keep nodes with higher (or lower) cost and flow transmission as hub (or non-hub) for a longer period. Let

$$A(i) = \frac{\sum_{j=1}^{n}(W_{ij}C_{ij} + W_{ji}C_{ji})}{\sum_{k=1}^{n}\sum_{j=1}^{n}(W_{kj}C_{kj} + W_{jk}C_{jk})} \quad \forall i \in N,$$

then, the vectors l_c and l_o represent the tabu tenure for non-hubs and hubs nodes, respectively. $l_c(i)$ and $l_o(i)$ for $i \in N$ are computed as follows

$$l_c(i) = \min\{(1 - A(i))(nx_1) + 1, nx_2\} \quad \forall i \in N, \tag{7}$$

$$l_o(i) = \min\{A(i)(n\sqrt{n}x_3) + 1, nx_4\} \quad \forall i \in N, \tag{8}$$

where x_1, x_2, x_3 and x_4 are predefined parameters. We use an array $t \in \mathbb{Z}$ to measure the recency of a solution. The element $t_j \in t$ represents the iteration in which the status of node j is changed (for $j = 1, \ldots, n$), and the initial value of t is determined based on values of l_o and l_c as follows

$$t(j) = \begin{cases} -l_o(j) & l(j) = j \\ l_c(j) & l(j) \neq j \end{cases}, \tag{9}$$

where l_c and l_o are defined in (7) and (8), respectively. When the status of node j is changed, t_j is updated. Node j is in the tabu list at iteration k, if

$$\begin{cases} k - t(j) < l_o(j) & l(j) = j \\ k - t(j) < l_c(j) & l(j) \neq j \end{cases}.$$

In other words, when the status of the node j changes to the hub, the node j must remain the hub for $l_o(j)$ iterations. Similarly, when the status of the node j is changed to the non-hub, the node j must remain the non-hub for $l_c(j)$ iterations.

3.5.3. Aspiration criterion

Sometimes, good solutions without risk for cycling are banned by strong tabu rules. To eliminate these tabus, the aspiration criterion is used. Consider z to be the current value of the objective function. Δz_j shows the change in the objective function's value when the status of node j is changed in the basic move, and $\Delta z_{i,j}$ shows the change in the objective function's value when the hub i and the non-hub j are exchanged in the swap move. Here, aspiration criterion in the basic move is satisfied for the node j, if $z + \Delta z_j < z_{00}$, and aspiration criterion in the swap move is satisfied for the hub i and the non-hub j, if $z + \Delta z_{i,j} < z_{00}$, where z_{00} is the best obtained objective function value.

3.5.4. Implementation of a location move

To do a basic move, the candidate list (unflagged nodes) is constructed (see Section 3.5.1). Let

$$j^* = \arg\min\{\Delta z_j |\ j \in N, \text{j is unflagged}\}.$$

where $\triangle z_j$ is the associated change in the objective function's value. If the node j^* is not in the tabu list or the aspiration criterion is satisfied for this node, the basic move is done for it, otherwise that node will be flagged in order to prevent the revaluation of the node j^* in the current iteration. This procedure continues until a move is done. After doing a move, all nodes will be unflagged.

Also, to do a swap move, (i^*, j^*) are determined as

$$(i^*, j^*) = \arg\min \Delta z_{i,j}.$$

where $\Delta z_{i,j}$ is the associated change in the objective function's value. If the hub i^* and the non-hub j^* are not in the tabu list or the aspiration criterion is satisfied for them, i^* and j^* will be swapped.

3.5.5. Stopping rule in the short term memory

The basic move in the short term memory stops when

$$k - k_0 > n_1, \tag{10}$$

where n_1 is a predefined parameter, k is the current iteration, and k_0 is the last iteration in which z_0 (see Algorithm 2) is updated. In other words, if the best solution in the current cycle have not been improved in the last n_1 iterations, the basic move stops. Next, a swap move is done then the short term memory stops.

3.6. Long term memory

In the long term memory, the solutions are diversified by changing the status of a node which its status were not changed for a longer time among nodes. Let

$$j^* = \arg\min_{j \in N} t(j). \tag{11}$$

If the node j^* is not the only hub, the status of the node j^* is changed and then the allocation of this solution is improved based on (5). This procedure is repeated n_2 times, where

$$n_2 = \max\{1, [bn]\}. \tag{12}$$

In (12), n is the number of nodes, and b is a predefined parameter.

After the cycle is repeated Ktimes (where K is a predefined parameter), the algorithm stops and the best found solution of the algorithm is returned.

3.7. TSUSAHLP

In TSUSAHLP, at first, a feasible solution is constructed by the greedy adopted DROP procedure (see Algorithm 1). In the first of each cycle, the short term memory is called (see Section 3.5). Then, the long term memory is performed (see Section 3.6). The next cycle starts with the solution from the last long term memory. This cycle is repeated for K times.

We now present TSUSAHLP in Algorithm 2. The cost of the best solution in the current cycle and the whole algorithm are denoted by z_0 and z_{00}, respectively.

Algorithm 2. TSUSAHLP

Step 0: {Initialization}
- Input $K \in \mathbb{Z}^+$ to determine the number of cycles (see Section), $y \in \mathbb{R}^+$ used in (6) to construct the candidate list, x_1 and $x_2 \in \mathbb{R}^+$ used in (7) to make l_c, x_3 and x_4 used in (8) to make l_o, $n_1 \in \mathbb{Z}^+$ (10) to determine the number of iterations in short term memory, and b used in (12) to determine n_2 (the number of iterations in long term memory).
- Construct the auxiliary matrix R (see Remark 3.1).
- Let: $kk \leftarrow 0$ (the counter of the number of cycles), $kl \leftarrow 0$ (the counter of the number of iterations in the long term memory), $UF \leftarrow N$ {all nodes are unflagged}.

Step 1:
- Construct S and l by using Algorithm 1, and let $z \leftarrow f(S, l)$ where $f(\cdot,\cdot)$ is defined in (1).
- $z_0 \leftarrow z$ (z_0 is the best solution found in the current cycle), $z_{00} \leftarrow z$ (z_{00} is the best found solution), $k \leftarrow 0$ and $k_0 \leftarrow 0$.
- Construct l_c and l_o based on (7) and (8), respectively.
- Construct t defined in (9).

Step 2: {Short term memory}

2-1: Let: $UF \leftarrow N - \{j |\ j \in S',\ |j - l(j)| < d\}$, where d is defined in (6).

2-2: Let: $j^* \leftarrow \arg\min_{j \in UF} \Delta z_j$. If $k - t_{j^*} \leqslant l(j^*)$ (j^* is in the tabu list), then go to **2–3**, else let basic move = 1, and go to **Step 3**.

2-3: If $z + \triangle z_{j^*} < z_{00}$ (aspiration criterion is satisfied), then let basic move = 1, and go to **Step 3**, else let $UF \leftarrow UF - \{j^*\}$ and go to **2–2**.

Step 3: {Updating the current solution}

3-1: If $j^* \in S$, then $S \leftarrow S - \{j^*\}$ and $p \leftarrow p - 1$, else $S \leftarrow S \cup \{j^*\}$ and $p \leftarrow p + 1$.

3-2: Construct the allocating function based on the distance-based rule OKelly (1987).

3-3: Let $z \leftarrow z + \triangle z_{j*}$, $k \leftarrow k + 1$ and $t(j^*) \leftarrow k$.

3-4: If $z < z_0$, then $z_0 \leftarrow z$ and $k_0 \leftarrow k$.

3-5: If $z_0 < z_{00}$, then $z_{00} \leftarrow z_0$.

3-6: If swap = 1, then swap = 0, $j^* \leftarrow i^*$, and go to **Step 3-1**.

3-7: If $k - k_0 < n_1$ and basic move = 1, go to **Step 2**, else if $k - k_0 = n_1$, let basic move = 0 and {Swap move} let: $(i^*, j^*) = \arg\min \Delta z_{i,j}$. If $k - t_{i^*} > l(i^*)$ and $k - t_{j^*} > l(j^*)$ (i^* and j^* are not in the tabu list), or $z + \triangle z_{i^* j^*} < z_{00}$ (aspiration criterion is satisfied), let swap = 1, and go to **Step 3**., else if $kl \leqslant n_2$, then go to **Step 4**.
else, let: $kk \leftarrow kk + 1$.
If $kk = K$, then **Stop**. Perform *TabuAlloc* proposed by Silva and Cunha (2009), and return z_{00} as the best value of the objective function, else $k_0 \leftarrow k$, $kl \leftarrow 0$, $z_0 \leftarrow z$ and go to **Step 2**.

Step 4: {Long term memory}

4-1: $kl \leftarrow kl + 1$, $UF \leftarrow N$.

4-2: Let: $j^* \leftarrow \arg\min_{j \in N} t_j$.

4-3: If $p > 1$ or j^* is not hub, then go to **Step 3** else go to **Step 3-7**.

4. Updating the objective function's value

Computing the objective function's value is a time consuming part of the algorithm. In TSUSAHLP, the objective function is calculated just once at first. Then, the change in the objective function's value is computed by using following propositions. So, the computational time of the algorithm is decreased, efficiently.

In a allocation move, the allocation of a non-hub node r is changed from $l(r)$ $(l: N \to S)$ to $l'(r)$ $(l': N \to S)$, in which $l'(i) = l(i),\ \forall i \neq r$. We denote the change in the objective function's value, from z to z', associated to l and l' respectively, by $\Delta z^r_{l,l'}$. The following proposition shows that $\Delta z^r_{l,l'}$ can be computed in $O(n)$.

Proposition 4.1. *For each $r \in S'$ $(N-S)$, $\Delta z^r_{l,l'}$ is as follows*

$$\begin{aligned}\Delta z^r_{l,l'} &= \chi\sum_{i=1}^{n} W_{ri}(C_{rl'(r)} - C_{rl(r)}) + \alpha\sum_{i=1}^{n} W_{ri}(C_{l'(r)l(i)} - C_{l(r)l(i)}) \\ &+ \delta\sum_{i=1}^{n} W_{ir}(C_{l'(r)r} - C_{l(r)r}) + \alpha\sum_{i=1}^{n} W_{ir}(C_{l(i)l'(r)} - C_{l(i)l(r)}) \\ &- \alpha W_{rr}(C_{l'(r)l(r)} + C_{l(r)l'(r)})). \end{aligned} \tag{13}$$

Proof. We know that

$$\begin{aligned}\Delta z^r_{l,l'} &= z' - z \\ &= \left(\sum_{i\in S} F_i + \sum_{i=1}^{n}\sum_{j=1}^{n} W_{ij}(\chi C_{il'(i)} + \alpha C_{l'(i)l'(j)} + \delta C_{l'(j)j})\right) \\ &- \left(\sum_{i\in S} F_i + \sum_{i=1}^{n}\sum_{j=1}^{n} W_{ij}(\chi C_{il(i)} + \alpha C_{l(i)l(j)} + \delta C_{l(j)j})\right) \end{aligned} \tag{14}$$

Since $l'(i) = l(i)$ for $i \neq r$, thus (14) turns to

$$\begin{aligned}\Delta z^r_{l,l'} &= \sum_{j=1,j\neq r}^{n} W_{rj}(\chi C_{rl'(r)} + \alpha C_{l'(r)l(j)} + \delta C_{l(j)j}) \\ &- \sum_{j=1,j\neq r}^{n} W_{rj}(\chi C_{rl(r)} + \alpha C_{l(r)l(j)} + \delta C_{l(j)j}) + \sum_{i=1,i\neq r}^{n} W_{ir}(\chi C_{il(i)} \\ &+ \alpha C_{l(i)l'(r)} + \delta C_{l'(r)r}) - \sum_{i=1,i\neq r}^{n} W_{ir}(\chi C_{il(i)} + \alpha C_{l(i)l(r)} + \delta C_{l(r)r}) \\ &+ W_{rr}(\chi C_{rl'(r)} + \alpha C_{l'(r)l'(r)} + \delta C_{l'(r)r}) \\ &- W_{rr}(\chi C_{rl(r)} + \alpha C_{l(r)l(r)} + \delta C_{l(r)r}). \end{aligned} \tag{15}$$

In USAHLP, $\forall i,\ C_{ii} = 0$. After simplification, from (15) we have:

$$\begin{aligned}\Delta z^r_{l,l'} &= \chi\sum_{i=1}^{n} W_{ri}(C_{rl'(r)} - C_{rl(r)}) + \alpha\sum_{i=1}^{n} W_{ri}(C_{l'(r)l(i)} - C_{l(r)l(i)}) \\ &+ \delta\sum_{i=1}^{n} W_{ir}(C_{l'(r)r} - C_{l(r)r}) + \alpha\sum_{i=1}^{n} W_{ir}(C_{l(i)l'(r)} - C_{l(i)l(r)}) \\ &- \alpha W_{rr}(C_{l'(r)l(r)} + C_{l(r)l'(r)}). \end{aligned}$$

Thus, the proof is complete. □

In a location move, when a hub node k is closed (k is changed to a non-hub node), all of the nodes allocated to node k (including node k, itself) are allocated to their nearest hub in $S - \{k\}$, where S is the current set of hubs. Let

$$P = \{j | j \in S' - \{k\}, l(j) = k\}. \tag{16}$$

Now, consider $P = \{j_1, j_2, \ldots, j_m\}$ in which $j_1 < j_2 < \cdots < j_m$. Let $y(j) = \min\{i | R_{ij} \in S - \{k\}\}$ and

$$l'_{j_0} = l, \quad l'_{j_t}(i) = \begin{cases} l(i) & i \in N - P \text{ or } i > j_t \\ R_{y(i)i} & i \in P,\ i \leqslant j_t \end{cases} \quad \text{for } t = 1, 2, \ldots, m,$$

$$\text{and} \quad l'_k(i) = \begin{cases} l(i) & i \in N - P,\ i \neq k \\ R_{y(i)i} & o.w. \end{cases}.$$

Now, in the following proposition, we show that the computation of Δz in this location move takes $O(mn)$ times, where $m = |P|$.

Proposition 4.2. *If the hub node k is removed from the set of hubs S, using above notations, the change in the objective function's value, Δz, is calculated by*

$$\Delta z = -F(k) + \sum_{t=1}^{m} \Delta z^{j_t}_{l'_{j_{t-1}}, l'_{j_t}} + \Delta z^{k}_{l'_{j_m}, l'_k}.$$

Proof. Consider that the associated allocating functions of S and $S - \{k\}$ are l and l', respectively. We denote the transportation costs corresponding to l and l' by z_l and $z_{l'}$, respectively.

Now, note that the associated hub for non-hub nodes in $N - P - \{k\}$ does not changed. Therefore, we have

$$\begin{aligned} z_{l'} - z_l &= z_{l'_k} - z_l \\ &= z_{l'_k} - z_{l'_{j_m}} + z_{l'_{j_m}} - z_{l'_{j_{m-1}}} + z_{l'_{j_{m-1}}} - \cdots - z_{l'_{j_1}} + z_{l'_{j_1}} - z_l \\ &= \left(z_{l'_k} - z_{l'_{j_m}}\right) + \sum_{t=1}^{m}\left(z_{l'_{j_t}} - z_{l'_{j_{t-1}}}\right). \end{aligned} \tag{17}$$

If P defined in (16) is $\varnothing$, then $l'_{jm} = l$ and $\sum_{t=1}^{m}(z_{l'_{j_t}} - z_{l'_{j_{t-1}}}) = 0$. Now, suppose that $P \neq \varnothing$. So, the functions $l'_{j_t}: N \to S$ and $l'_{j_{t-1}}: N \to S$ are different just in the allocation of non-hub node j_t, which changes from $l'_{j_{t-1}}(j_t)$ to $l'_{j_t}(j_t)$. Also, the functions $l'_k: N \to S$ and $l'_{j_m}: N \to S$ are different just in the allocation of non-hub node k changes from $l'_{j_m}(k)$ to $l'_k(k)$. So, from (17) we have

$$z_{l'_k} - z_l = \sum_{t=1}^{m} \Delta z^{j_t}_{l'_{j_{t-1}}, l'_{j_t}} + \Delta z^{k}_{l'_{j_m}, l'_k}.$$

Since no node is allocated to hub k, it can be omitted from the set of hubs. So, establishing costs will change. Eventually, Δz is:

$$\Delta z = \sum_{i\in S-\{k\}} F(i) - \sum_{i\in S} F(i) + z_{l'_k} - z_l = -F(k) + \sum_{t=1}^{m} \Delta z^{j_t}_{l'_{j_{t-1}}, l'_{j_t}} + \Delta z^{k}_{l'_{j_m}, l'_k}.$$

So, the proof is complete. □

Other possible case in a location move can occur when a non-hub node becomes a hub. Suppose that node k is added to the set of hubs. So, the node k must be allocated to itself, and the other non-hub nodes which are closer to node k than their previous hubs will take k as their hub. Let

$$Q = \{j | j \in S' - \{k\}, C(j, l(j)) > C(j, k)\}. \tag{18}$$

Consider $Q = \{j_1, j_2, \ldots, j_m\}$ in which $j_1 < j_2 < \cdots < j_m$, and let

$$l'_k(i) = \begin{cases} l(i) & i \neq k \\ k & i = k \end{cases}, \quad l'_{j_0} = l'_k,$$

and

$$l'_{j_t}(i) = \begin{cases} l(i) & i \in N - Q - \{k\} \text{ or } i > j_t \\ k & i \in Q \cup \{k\},\ i \leqslant j_t \end{cases}.$$

Now, Proposition 4.3 shows Δz can be computed in $O(mn)$ times, where $m = |Q|$.

Proposition 4.3. *Suppose that the non-hub node k is added to the set of hubs. So, using above notations, the change in the objective function's value, Δz, is*

$$\Delta z = F(k) + \Delta z^{k}_{l,l'_k} + \sum_{t=0}^{m} \Delta z^{j_t}_{l'_{j_{t-1}},l'_{j_t}}.$$

Proof. The change in the objective function's value is due to the change of set of hubs from S to $S\cup\{k\}$ and allocating function from l to l'_{j_m}. We have

$$\begin{aligned}\Delta z &= \sum_{i\in S\cup\{k\}} F(k) - \sum_{i\in S} F(i) + z_{l'_{j_m}} - z_l \\ &= F(k) + z_{l'_{j_m}} - z_{l'_{j_{m-1}}} + z_{l'_{j_{m-1}}} - \cdots - z_{l'_{j_1}} + z_{l'_{j_1}} - z_{l'_k} + z_{l'_k} - z_l \\ &= F(k) + \sum_{t=1}^{m}\left(z_{l'_{j_t}} - z_{l'_{j_{t-1}}}\right) + (z_{l'_k} - z_l).\end{aligned}$$

If Q defined in (18) is $\varnothing$, then allocating functions l'_k and l'_{j_m} are the same. So, $\sum_{t=1}^{m}(z_{l'_{j_t}} - z_{l'_{j_{t-1}}}) = 0$. Suppose that $Q \neq \varnothing$. Allocating functions $l'_k : N \to S\cup\{k\}$ and $l : N \to S\cup\{k\}$ are different just in the allocation of node k. Also, the allocating functions $l'_{j_t} : N \to S\cup\{k\}$ and $l'_{j_{t-1}} : N \to S\cup\{k\}$ are just different in the allocation of non-hub node j_t which changes from $l'_{j_{t-1}}$ to l'_{j_t}. Thus,

$$\Delta z = F(k) + \Delta z^{k}_{l,l'_k} + \sum_{t=1}^{m} \Delta z^{j_t}_{l'_{j_{t-1}},l'_{j_t}}. \quad \square$$

Using above propositions, the required time for computation of Δz is reduced, considerably. Thus, the running time will be decreased by applying above results in our implementations.

5. Computational experiments

All experiments were done on a notebook Intel core i5, 2.53 GHz with 4 GB RAM memory under Windows 7 home premium system. Our proposed algorithm (TSUSAHLP; Algorithm 2) and Silva and Cunha's algorithm (HubTS) are implemented in Matlab ver. 8.2. Proposition 4.1 is applied in the implementation of HubTS which obviously reduces its computational time.

Computational results over standard CAB and AP data sets from ORLIB (Beasley, 2011), modified version of AP data set proposed by Silva and Cunha (2009), and also, four larger instances based on AP data set proposed by Silva and Cunha (2009) are reported in Sections 5.1–5.4, respectively.

The notations used in these tables is defined as follows. $\mathbf{n_H}$ is the number of hubs in the optimal solutions. $\mathbf{K_{opt}}$ and $\mathbf{t_{opt}}$ are the number of iterations of the cycle and the time (in second) needed to reach the optimal or the best solution, respectively. $\mathbf{t}$ is the total running time (in second) for the associated algorithm, and **solution** shows the rounded solution value obtained by the algorithm. $k_{opt} = 0$ means that the optimal or the best solution is obtained in the greedy adopted DROP method. The sign *opt* in the solution means that the algorithm can find the optimal value.

Here, we describe the way that we determined parameters, briefly. At first, an interval for each parameter is considered. Then, some numbers of each interval are chosen and the corresponding objective function's value is calculated. Afterward, the intervals are shortened based on the quality of associated solution. This procedure is iterated until desirable parameters are achieved.

We set K to 11 for the problems CAB, AP and modified AP instances, and 3 for four larger instances based on AP data set. Other values were considered as: $y = 0.33$ defined in (6), $x_1 = 0.25$ and $x_2 = 0.25$ defined in (7), $x_3 = 0.05$ and $x_4 = 0.33$ defined in (8), $n_1 = 2$ defined in (10) and $b = 0.025$ defined in (12).

5.1. CAB data set

CAB data set, which is used for evaluating many algorithms, is introduced by OKelly (1987). This data set is based on airline passenger flow between 25 cities of 100 US cities in 1970 estimated by the Civil Aeronautics Board. The distances between cities satisfy the triangle inequality, flow matrix is symmetric (i.e. $W_{ij} = W_{ji}$) and it is achieved by dividing the flow matrix given in ORLIB by total flow $\left(\text{i.e. } \sum_{i\in N}\sum_{j\in N} W_{ij}\right)$.

The size of problem in this data set, n, is 10, 15, 20 and 25. The discount factor, α, equals to 0.2, 0.4, 0.6, 0.8, and 1.0. The collection factor, χ, and the distribution factor, δ, equal to the constant value 1. The fixed cost for establishing a hub, F_k, is chosen in {100, 150, 200, 250} that is identical for all nodes.

The optimal results for all 80 CAB instances are given in Silva and Cunha (2009). But the optimal solution value for the problem with $n = 10$, $\alpha = 1$ and $F_k = 150$ is reported 1181.05 by Silva and Cunha (2009), while we obtained 1081.05 as the optimal value for this problem. We believe that it is just a typos, since the set of hubs that they reported was the same as we do. In addition, it has just one hub which means the allocation is unique. Furthermore, we obtained the right optimal value by running HubTS in Matlab. It is notable that this wrong report has not been corrected in some of the next literature, for example in (Marić et al., 2013; Naeem & Ombuki-Berman, 2010) the value 1181.05 is reported for this problem, too. While in Ting and Wang (2014), this is reported 1081.05. Since both TSUSAHLP and HubTS were able to find the optimal solutions in all 80 CAB instances, we compare only the corresponding running times in Tables 1 and 2. Moreover TSUSAHLP, the GA (Naeem & Ombuki-Berman, 2010), the MA (Marić et al., 2013), the discrete PSO (Bailey et al., 2013), and the TA (Ting & Wang, 2014) obtained the optimal solution over the CAB data set. Therefore, the table comparing these algorithms is omitted.

In many CAB instances, TSUSAHLP obtained the optimal solutions in the early iterations, and the maximum k_{opt} occurred in the 9th iteration of the cycle. In almost all cases, TSUSAHLP was able to find the optimal solution much faster than HubTS, especially when the optimal solution contains more than one hub. It seems that increasing the number of established hubs in optimal solutions increases the running time of HubTS, while it does not affect on TSUSAHLP. The maximum running time for TSUSAHLP and HubTS are 1.076 and 7.109 s, respectively, while, the maximum running time for TSUSAHLP, to find the optimal solution, equals to 0.553 s. The running times more than one second are bold in these tables.

5.2. AP data set

AP data set, which is used by Ernst and Krishnamoorthy (1996) for the Australian Post primarily, contains the information of 200 nodes that is available at ORLIB (Beasley, 2011). The collecting factor, χ, transferring factor between hubs, α, and distributing factor, δ, equal to 3, 0.75 and 2, respectively. The flow matrix is not symmetric (i.e. $W_{ij} \neq W_{ji}$) and the flow from each node to itself is not necessarily zero (i.e. $W_{ii} \neq 0$). AP data set consists of two types of fixed costs: tight costs which is depended on the flows and loose costs which is independent of the flows. Ernst and Krishnamoorthy (1999) claimed that instances with tight costs are more difficult in comparison with instances with loose costs. The optimal solutions for AP data set up to 100 nodes are given in Silva and Cunha (2009), but the Optimal value of AP instances with 200 nodes still remains a challenge.

Table 3 presents the results of TSUSAHLP and HubTS, the GA (Naeem & Ombuki-Berman, 2010), the MA (Marić et al., 2013),

the discrete PSO (Bailey et al., 2013), and the TA (Ting & Wang, 2014) over AP data set. Types of instances, loose and tight, are singed with *L* and *T*, respectively.

All of these algorithms obtained the known optimal solutions over the AP data set. But for the AP instances with 200 nodes, all the algorithms except HubTS obtained the best known solutions. It is notable that the best known value for the AP instance with 200 nodes and loose type is 2338020.976. It is reported 233802.97 or 272802.98 in the other algorithms. The maximum running time for TSUSAHLP and HubTS are 79.513 and 463.994 s, respectively, and the maximum value of k_{opt} in TSUSAHLP equals to 7. The results show that TSUSAHLP is more efficient than HubTS in both computational times and solution values. Furthermore, TSUSAHLP obtained the optimal or the best known solution in all AP instances.

5.3. Modified AP data set

Silva and Cunha (2009) proposed a modified version of AP data set by applying a reduction factor (θ) to the fixed costs of establishing hubs in order to increase the number of hubs in the optimal solutions when θ is either 0.8 or 0.9. Optimal results on modified AP data set with $\theta = 0.8$ and 0.9 for instances up to 100 nodes are presented in Silva and Cunha (2009). But the optimal solution value for the problem $n = 50$ and $\theta = 0.9$ with tight costs was reported 295239.13 with the set of hubs {17, 48} by Silva and Cunha (2009), while TSUSAHLP obtained the value 291435.49 for this problem with the same set of hubs. Moreover, Marić et al. (2013) and Ting and Wang (2014) reported the same value for this problem, as we do. However, HubTS were able to find the optimal solution while we ran it in Matlab. TSUSAHLP, HubTS (Silva & Cunha, 2009), the MA (Marić et al., 2013), and the TA (Ting & Wang, 2014) are compared in Tables 4 and 5 over modified AP data set.

The performance of TSUSAHLP on the modified AP data set was approximately similar to its performance on AP data set. Therefore, the number of hubs in the optimal solution did not play a major role in the performance of TSUSAHLP. All TSUSAHLP, HubTS, the MA and the TA found the optimal solution over the modified AP instances up to 100 nodes, but TSUSAHLP and the TA found better solutions on the modified AP instances with 200 nodes. The running times of TSUSAHLP was less than HubTS in all of these instances, considerably. The maximum running time for TSUSAHLP and HubTS are 75.255 and 939.614 s, respectively.

5.4. Large instances based on AP data set

Four new large instances are generated based on the full AP data set by Silva and Cunha (2009). Results of TSUSAHLP, HubTS (Silva & Cunha, 2009), the MA (Marić et al., 2013), and the TA (Ting & Wang, 2014) are presented in Table 6. TSUSAHLP found the best known solutions, and modified the best known solution for the instance with 300 nodes and loose type. The chosen hubs in this problem are 26, 79, 170, 187, and 251. In all of these instances, TSUSAHLP found better solution than HubTS (TSUSAHLP is at least 40 times faster than HubTS). The maximum running time of TSUSAHLP and HubTS are 115.863 and 8887.322, respectively. It seems that when the size of problem increases, TSUSAHLP significantly outperformed HubTS.

It should be mentioned that the obtained solution by TSUSAHLP is not dependent on the runs. In other words, TSUSAHLP obtained the optimal or best known solution is in each run, while the other proposed procedures are dependent on the runs. For instance the TA (Ting & Wang, 2014) found the optimal solutions on average 18.5 in 20 runs.

TSUSAHLP outperformed HubTS, the GA (Naeem & Ombuki-Berman, 2010), the MA (Marić et al., 2013), the discrete PSO (Bailey et al., 2013), and the TA (Ting & Wang, 2014) in terms of solution quality. Also, the running time of TSUSAHLP was less than HubTS's running time, on average. Since different algorithms have been implemented over different computers, it is difficult to compare their CPU times, generally. Here, the running times required to obtain the results of the MA and the TA have directly taken from Marić et al. (2013) and Ting and Wang (2014) respectively in order to get an idea about their effects in the solution quality. Unfortunately, the running times and the information of the implementations for the PSO and the GA haven't been reported by their authors. The MA was coded in C and its experiments were carried out on an Intel Core i7-860 2.8 GHz with 8 GB RAM memory under Windows 7 Professional. The TA was coded in C++ by Microsoft Visual Studio 2008 software. All numerical tests on the TA were carried out on a Pentium IV 3.0 GHz PC, with 768 MB RAM, running under Windows XP Professional.

The running times of the proposed MA and TA over AP data set, modified AP data set and new large AP data set with $n = 300$ and 400 are provided in Table 7 and 8, respectively. The result of MA (TA), reported before, is the best found solution over 20 runs and t_{MA} (t_{TA}) in Table 7 (8) represents the average of running times in second. While $\theta = 1$, the data set represents the original AP data set. A dash (−) indicates that the instance was not tested by the algorithms.

6. Conclusions

We here proposed an efficient tabu search (TS) for solving the uncapacitated single allocation hub location problem (USAHLP). To decrease the computational time, in the proposed tabu search, some new tabu rules were considered. Also, to compute the changes in the objective function's value in each move, some new results were given. The performance of the proposed tabu search was compared with recently proposed approaches on all standard ORLIB instances (CAB and AP data sets), modified AP data set and finally on four large instances with 300 and 400 nodes proposed by Silva and Cunha. The numerical experiments showed that the proposed algorithm obtained the optimal solution for instances which optimal solutions are proven. Furthermore, our proposed tabu search obtained the best known solution found in the literature or surpass them in a very short computational time. It is notable that our proposed tabu search obtained the same solution in each run against other methods proposed for USAHLP. Our proposed TS outperforms the recently proposed TS (Silva & Cunha, 2009) in both solution values and computational times.

Acknowledgements

We thank M. Krishnamoorthy, M. R. Silva, C. B. Cunha and M. Maric for providing us the test problems. Also, we are grateful to the two anonymous referees and the editor for their constructive comments and questions. Consideration of all improved the presentation. Finally, we thank the Research Council of Ferdowsi University of Mashhad for its support.

References

Abdinnour-Helm, S. (1998). A hybrid heuristic for the uncapacitated hub location problem. *European Journal of Operational Research, 106*, 489–499.

Abdinnour-Helm, S., & Venkataramanan, M. A. (1998). Solution approaches to hub location problems. *Annals of Operations Research, 78*, 31–50.

Alumur, S., & Kara, B. Y. (2008). Network hub location problems: The state of the art. *European Journal of Operational Research, 190*, 1–21.

Aversa, R., Botter, H. E., Haralambides, R. C., & Yoshizaki, H. T. Y. (2005). A mixed integer programming model on the location of a hub port in the east coast of South America. *Maritime Economics and Logistics, 7*, 1–18.

Aykin, T. (1995). Networking policies for hub-and-spoke systems with application to the air transportation system. *Transportation Science, 29*, 201–221.

Aykin, T., & Brown, G. F. (1992). Interacting new facilities and location–allocation problem. *Transportation Science, 26*, 212–222.

Bailey, A., Ornbuki-Berrnan, B., & Asobiela, S., (2013). Discrete pso for the uncapacitated single allocation hub location problem. In *Computational intelligence in production and logistics systems (CIPLS), 2013 IEEE workshop on* (pp. 92–98). (http://dx.doi.org/10.1109/CIPLS.2013.6595205).

Bania, N., Bauer, P., & Zlatoper, T. (1998). Us air passenger service: A taxonomy of route networks, hub locations, and competition. *Logistics and Transportation Review, 34*, 53–74.

Beasley, J. E. (2011). Or-library. <http://people.brunel.ac.uk/mastjjb/jeb/info.html> Accessed 1.06.11.

Campbell, J. F. (1990). Locating transportation terminals to serve an expanding demand. *Transportation Research, Part B, 24*, 173–192.

Campbell, J. F. (1994). Integer programming formulations of discrete hub location problems. *European Journal of Operational Research, 72*, 387–405.

Campbell, J. F. (2010). Designing hub networks with connected and isolated hubs. In *43th Hawaii international conference system sciences (HICSS-43)* (pp. 1–10). koloa, Kauai: IEEE Computer Society.

Campbell, J. F., Ernst, A. T., & Krishnamoorthy, M. (2005a). Hub arc location problems: Part 1 – Introduction and results. *Management Science, 51*, 1540–1555.

Campbell, J. F., Ernst, A. T., & Krishnamoorthy, M. (2005b). Hub arc location problems: Part 2 – Formulations and optimal algorithms. *Management Science, 51*, 1556–1571.

Campbell, J. F., Ernst, A. T., & Krishnamoorthy, M. (2002). Hub location problems. In *Facility location: Applications and theory* (pp. 373–407).

Campbell, J. F., Stiehr, G., Ernst, A. T., & Krishnamoorthy, M. (2003). Solving hub arc location problems on a cluster of workstations. *Parallel Computing, 29*, 555–574.

Carello, G., Croce, F. D., Ghirardi, M., & Tadel, R. (2004). Solving the hub location problem in telecommunications network design: A local search approach. *Networks, 44*, 94–105.

Chen, J. F. (2007). A hybrid heuristic for the uncapacitated single allocation hub location problem. *Omega, 35*, 211–220.

Contreras, I., Cordeau, J. F., & Laporte, G. (2011). The dynamic uncapacitated hub location problem. *Transportation Science, 45*, 18–32.

Contreras, I., Cordeau, J. F., & Laporte, G. (2011). Stochastic uncapacitated hub location. *European Journal of Operational Research, 212*, 518–528.

Contreras, I., Cordeau, J. F., & Laporte, G. (2012). Exact solution of large-scale hub location problems with multiple capacity levels. *Transportation Science, 46*, 438–469.

Contreras, I., Fernández, E., & Marín, A. (2009). Tight bounds from a path based formulation for the tree of hub location problem. *Computers & Operations Research, 36*, 3117–3127.

Cunha, C. B., & Silva, M. R. (2007). A genetic algorithm for the problem of configuring a hub-and-spoke network for a LTL trucking company in Brazil. *European Journal of Operational Research, 179*, 747–758.

Don, T., Harit, S., English, J., & Whisker, G. (1995). Hub and spoke networks in truckload trucking: Configuration, testing, and operational concerns. *Logistics and Transportation, 31*, 209–237.

Ebery, J., Krishnamoorthy, M., Ernst, A., & Boland, N. (2000). The capacitated multiple allocation hub location problem: Formulations and algorithms. *European Journal of Operational Research, 120*, 614–631.

Eiselt, H. A., & Marianov, V. (2009). A conditional *p*-hub location problem with attraction functions. *Computers & Operations Research, 36*, 3128–3135.

Ernst, A. T., & Krishnamoorthy, M. (1996). Efficient algorithms for the uncapacitated single allocation *p*-hub median problem. *Location Science, 4*, 139–154.

Ernst, A. T., & Krishnamoorthy, M. (1999). Solution algorithms for the capacitated single allocation hub location problem. *Annals of Operations Research, 86*, 141–159.

Farahani, R. Z., Hekmatfar, M., Arabani, A. B., & Nikbakhsh, E. (2013). Hub location problems: A review of models, classification, solution techniques, and applications. *Computers & Industrial Engineering, 64*, 1096–1109.

Filipović, V., Kratica, J., Tošić, D., & Dugošija, D. (2009). Ga inspired heuristic for uncapacitated single allocation hub location problem. *Applications of Soft Computing, 58*, 149–158.

Gelareh, S., Nickel, S., & Pisinger, D. (2010). Liner shipping hub network design in a competitive environment. *Transportation Research Part E: Logistics and Transportation Review, 46*, 991–1004.

Glover, F. (1986). Future paths for integer programming and links to artificial intelligence. *Computer & Operations Research, 13*, 533–549.

Glover, F., & Laguna, M. (1997). *Tabu search*. Norwell, MA, USA: Kluwer Academic Publishers.

Hakimi, S. L. (1964). Optimum locations of switching centres and the absolute centres and medians of a graph. *Operations Research, 12*, 450–459.

Hamacher, H. W., Labbé, M., Nickel, S., & Sonneborn, T. (2004). Adapting polyhedral properties from facility to hub location problems. *Discrete Applied Mathematics, 145*, 104–116.

Han-yi, Y. (2010). An effective hybrid heuristic algorithm for the hub location problems of small-scale uncapacitated hub-and-spoke network. *Journal of Southwest Jiaotong University (English Edition), 18*, 225–230.

Helme, M. P., & Magnanti, T. L. (1989). Designing satellite communication networks by zero-one quadratic programming. *Networks, 19*, 427–450.

Jaeggi, D., Parks, G., Kipouros, T., & Clarkson, P. (2008). The development of a multi-objective tabu search algorithm for continuous optimisation problems. *European Journal of Operational Research, 185*, 1192–1212.

Karimi, H., & Bashiri, M. (2011). Hub covering location problems with different coverage types. *Scientia Iranica, 18*, 1571–1578.

Kim, H., & OKelly, M. E. (2009). Reliable *p*-hub location problems in telecommunication networks. *Geographical Analysis, 41*, 283–306.

Klencewicz, J. (1998). Hub location in backbone tributary network design: A review. *Location Science, 6*, 307–335.

Korte, B., & Vygen, J. (2008). *Combinatorial optimization. Theory and algorithms* (4th ed.). Springer.

Kuby, M., & Gray, R. (1993). Hub network design problems with stopovers and feeders: Case of Federal Express. *Transportation Research, 27*, 1–12.

Labbé, M., & Yaman, H. (2004). Projecting flow variables for hub location problems. *Networks, 44*, 84–93.

Labbé, M., Yaman, H., & Gourdin, E. (2005). A branch and cut algorithm for the hub location problems with single assignment. *Mathematical Programming, 102*, 371–405.

Lium, A. G., Crainic, T. G., & Wallace, S. W. (2009). A study of demand stochasticity in service network design. *Transportation Science, 43*, 144–157.

Love, R., Morris, J., & Wesolowsky, G. (1988). *Facilities location: Models & methods*. North-Holland: Publications in Operations Research.

Marianov, V., Serra, D., & ReVelle, C. (1999). Location of hubs in a competitive environment. *European Journal of Operational Research, 114*, 363–371.

Marić, M., Stanimirović, Z., & Stanojević, P. (2013). An efficient memetic algorithm for the uncapacitated single allocation hub location problem. *Soft Computing, 17*, 445–466.

Marín, A. (2005). Formulating and solving splittable capacitated multiple allocation hub location problems. *Computers & Operations Research, 32*, 3093–3109.

Mayer, G., & Wagner, B. (2002). Hub locator: An exact solution method for the multiple allocation hub location problem. *Computers & Operations Research, 29*, 715–739.

Naeem, M., & Ombuki-Berman, B., (2010). An efficient genetic algorithm for the uncapacitated single allocation hub location problem. In *IEEE congress on evolutionary computation, CEC 2010*.

OKelly, M. E. (1987). A quadratic integer program for the location of interacting hub facilities. *European Journal of Operational Research, 32*, 393–404.

OKelly, M. E. (1992). A clustering approach to the planar hub location problem. *Annals of Operations Research, 40*, 339–353.

OKelly, M. E. (1992). Hub facility location with fixed costs. *Papers in Regional Science, 71*, 293–306.

OKelly, M. E., Bryan, D., Skorin-Kapov, D., & Skorin-Kapov, J. (1996). Hub network design with single and multiple allocation: A computational study. *Location Science, 4*, 125–138.

Pirim, H., Eksioglu, B., & Bayraktar, E. (2008). *Tabu search: A comparative study*. INTECH Open Access Publisher.

Silva, M. R., & Cunha, C. B. (2009). New simple and efficient heuristics for the uncapacitated single allocation hub location problem. *Computers & Operations Research, 36*, 3152–3165.

Sim, T., Lowe, T., & Thomas, B. (2009). The stochastic *p*-hub center problem with service-level constraint. *Computers & Operations Research, 36*, 3166–3177.

Skorin-Kapov, D., Skorin-Kapov, J., & OKelly, M. E. (1996). Tight linear programming relaxations of uncapacitated *p*-hub median problems. *European Journal of Operational Research, 94*, 582–593.

Sun, M. (2006). Solving the uncapacitated facility location problem using tabu search. *Computers & Operations Research, 33*, 2563–2589.

Ting, C.-J., & Wang, H.-J. (2014). A threshold accepting algorithm for the uncapacitated single allocation hub location problem. *Journal of the Chinese Institute of Engineers, 37*, 300–312.

Topcuoglu, H., Corut, F., Ermis, M., & Yilmaz, G. (2005). Solving the uncapacitated hub location problem using genetic algorithms. *Computers & Operations Research, 32*, 967–984.

Yang, T. H. (2009). Stochastic air freight hub location and flight routes planning. *Applied Mathematical Modelling, 33*, 4424–4430.