Benders/Gossip Methods for Heterogeneous Multi-Vehicle Routing Problems

Benders/Gossip Methods for Heterogeneous Multi-Vehicle Routing Problems

Sarmad Riazi
Chalmers University of Technology
Gothenburg, Sweden
sarmadr@student.chalmers.se

Oskar Wigström
Chalmers University of Technology
Gothenburg, Sweden
oskar.wigstrom@chalmers.se

Carla Seatzu
Dep. of Electrical and Electronic Eng.
Univ. of Cagliari, Italy
seatzu@diee.unica.it

Bengt Lennartson
Chalmers University of Technology
Gothenburg, Sweden
bengt.lennartson@chalmers.se

Abstract

In this paper, we propose a logic-based Benders decomposition (LBBD), as well as an LBBD/gossip method to solve the heterogeneous multi-vehicle routing problem (HMVRP). HMVRP is a newly formalized extension of the NP-hard multi-traveling salesman problem (mTSP). First, a hybrid algorithm based on LBBD is formulated that decomposes the HMVRP into an assignment problem and a cluster of sequencing problems. The former is solved by a mixed integer linear programming (MILP) solver, and the latter by a dedicated TSP solver. Then, a gossip algorithm is constructed which utilizes the mentioned LBBD for local optimization to achieve better computational efficiency. The use of LBBD remarkably reduces the CPU time. Furthurmore, integrating the three layers of gossip algorithm, LBBD and the TSP solver, results in a very efficient solution method.

1. Introduction

The traveling salesman problem (TSP) and vehicle routing problem (VRP) are among the most well known and well studied combinatorial NP-hard problems [12]. A generic TSP can be described as a set of places to visit, and the set of corresponding distances (weighted edges) between them. The goal is to find the shortest path connecting all the places, such that each place is visited exactly once, and then return to the starting point. A survey of eight different integer programming formulations of the TSP is presented in [13]. Moreover, Laporte [9] provides an extensive review of exact and heuristic methods for solving TSPs.

The VRP, on the other hand, could be stated as the problem of finding the optimal delivery routes from one or several depots to a number of geographically scattered cities, subject to side constraints such as capacity constraint on carrier vehicles [10]. Because of the close relationship between TSP and VRP, many exact and heuristic algorithms for TSP were adopted in VRP procedures, see for example [10], [11], [14] and [15].

An interesting extension of the TSP is obtained by considering more than one salesman; a generalization known as a multi-traveling salesman problem (mTSP). In [1], Bektas provides a detailed survey of applications and formulations of mTSP as well as possible solution procedures. In addition, one may conceive mTSP as a subset of the VRP; where in a VRP additional capacity constraints are enforced to vehicles (salesmen). Hence, mTSP can be imagined as a bridge between TSP and VRP. The interesting point is that for both mTSP and VRP, the main focus has been on the single-depot case, where the salesmen or vehicles start their journey from the same depot.

A recent publication by Bektas [2] presents several new flow-based formulations for a class of mTSPs called *fixed-destination multi-depot salesmen problem with load balancing* (FMSPB). It successfully utilizes the classic Benders decomposition method [3] for a number of the formulations, as an exact solution procedure.

Moreover, for the VRP, Carlsson et al. [4] for the first time introduce the notion of *minimizing the maximum length of a tour* for the multi-depot multi-vehicle routing problem (MDMVRP), as opposed to *minimizing the tour length* prevalent in the literature. They also propose two heuristic methods for solving this problem. One such algorithm is a load balancing heuristic, based on linear programming (LP), in which the number of nodes assigned to each depot is balanced in the beginning, and then for each depot a tour is found using a TSP solver.

Recently, Franceschelli et al. [5] extended Carlsson et al.'s work by considering a *heterogeneous multi-vehicle routing problem* (HMVRP), and proposed a distributed

This work was carried out at the Wingquist Laboratory VINN Excellence Center within the Area of Advance – Production at Chalmers, supported by the Swedish Governmental Agency for Innovation Systems (VINNOVA) and the Swedish Research Council. The support is gratefully acknowledged.

gossip algorithm to solve it. The structure of the HMVRP formulation provides an excellent opportunity to utilize a partitioning method such as the logic-based Benders decomposition (LBBD) [7] as an exact solution procedure, which is treated in this paper.

The HMVRP formulation introduced in [5] is the basis of our work. The problem can be stated as a a set of tasks (nodes) to execute, and a set of depots, each having only one robot. Each robot has to travel between the assigned tasks, perform each task only once, and then return to the original depot. Costs of operations, robot task execution speeds, and robot traveling speeds are all heterogeneous, thus yielding a generalized form of an MVRP (or mTSP). The objective is to minimize the maximum execution time (make-span) of all robots, such that each task is performed only once.

In this paper we formulate the logic-based Benders method to decompose the HMVRP into an assignment master problem and a set of sequencing sub-problems. The sub-problems are solved either by CPLEX or by a TSP solver. Tran and Beck in [16] have adopted the same approach to address a parallel machine scheduling problem (PMSP), and our model could be considered as a subset of theirs. However, HMVRP has fewer constraints in the master problem than PMSP. In this case, having fewer constraints implies a harder problem.

Four main contributions of this paper are: (1) We extend the MILP-gossip algorithm in [5] with a TSP solver using LBBD; (2) we extend the gossip algorithm considering more than two robots in local problems; (3) we provide benchmarks for the extended versions of the gossip algorithm; (4) we provide numerical simulations and complexity evaluations for the decomposed HMVRP problem.

2. Problem definition

This paper concerns a particular formulation of VRP known as Heterogeneous Multi-Vehicle Routing Problem (HMVRP), which was first introduced in [5]. To state this NP-hard problem, we consider the two following sets whose members are scattered in the space:

- $\mathcal{N}$: The set of n robots
- $\mathcal{K}$: The set of k tasks

The movement speed and tasks execution speed of each robot may be different from another. Moreover, costs of the tasks vary depending on robots. This means that to assign the tasks to robots, a *load-balancing* strategy may not be as effective here as it was in [4]. To formulate the tours taken by robots we introduce the following sets:

- $\mathcal{V} = \mathcal{N} \cup \mathcal{K}$: The set of all nodes, i.e. $|\mathcal{V}| = n + k$.
- $\mathcal{E} = (\mathcal{N} \cup \mathcal{K}) \times (\mathcal{N} \cup \mathcal{K})$: The set of all directed edges, i.e. $|\mathcal{E}| = (n + k)^2$.

To formalize a MILP model, two sets of binary variables are required; one for task assignment and the other for sequence planning, as follows;

- $\mathcal{X}$: The set of all task assignment variables x_{ir}, where $i \in \mathcal{V}$ and $r \in \mathcal{N}$ with $|\mathcal{X}| = n \times (n + k)$. Also,
  If $i \in \mathcal{N}$ and $x_{ir} = 1$: Robot R_r starts its tour from depot (node) i,
  If $i \in \mathcal{K}$ and $x_{ir} = 1$: Task i is executed by robot R_r.
- $\mathcal{Y}$: The set of all sequencing variables y_{ijr}, where $(i, j) \in \mathcal{E}$ and $r \in \mathcal{N}$ with $|\mathcal{Y}| = n \times (n+k)^2$. Notice that $y_{ijr} = 1$ means that robot R_r goes directly from node i to node j. To prevent from taking self-loops, large costs should be assigned to edges with $i = j$.

Additionally, a continuous variable λ is defined to model the make-span. Finally, the following costs are defined:

- c_{ir}: The execution time of task i with the cost c_i ($i \in \mathcal{K}$) by robot R_r ($r \in \mathcal{N}$) with an execution speed w_r, i.e. $c_{ir} = c_i / w_r$.
- d_{ijr}: Represents the time spent by robot R_r to travel the Euclidean distance along the edge $(i, j) \in \mathcal{E}$ with speed v_r.

Ultimately, the *centralized* MILP formulation is as follows:

$$
\begin{array}{lll}
min\ \lambda & & \\
\text{s.t.} & & \\
\displaystyle\sum_{i \in \mathcal{K}} x_{ir} c_{ir} + \sum_{(i,j) \in \mathcal{E}} d_{ijr} y_{ijr} < \lambda & \forall r \in \mathcal{N} & \text{(C.1)} \\
x_{rr} = 1 & \forall r \in \mathcal{N} & \text{(C.2)} \\
\displaystyle\sum_{r \in \mathcal{N}} x_{ir} = 1 & \forall i \in \mathcal{K} & \text{(C.3)} \\
\displaystyle\sum_{j \in \mathcal{V}} y_{ijr} = \sum_{j \in \mathcal{V}} y_{jir} = x_{ir} & \forall i \in \mathcal{V}, & \\
 & \forall r \in \mathcal{N} & \text{(C.4)} \\
\displaystyle\sum_{i \notin \mathcal{S}} \sum_{j \in \mathcal{S}} y_{ijr} \geq x_{qr} & \forall \mathcal{S} \subseteq \mathcal{K}, & \\
 & \forall q \in \mathcal{S}, & \\
 & \forall r \in \mathcal{N} & \text{(C.5)} \\
\lambda \in \mathbb{R} & & \text{(C.6)} \\
x_{ir} \in \{0, 1\} & \forall i \in \mathcal{V}, & \\
 & \forall r \in \mathcal{N} & \text{(C.7)} \\
y_{ijr} \in \{0, 1\} & \forall (i, j) \in \mathcal{E}, & \\
 & \forall r \in \mathcal{N} & \text{(C.8)}
\end{array}
$$

The solution to the above problem yields the optimal task assignment [5]. The constraints in (C.1) along with the objective function aim to minimize the maximum execution time of robots (make-span); (C.2) ensures that each robot is dispatched from its depot; (C.3) guarantees the assignment of each task to just one robot; (C.4) means that exactly one edge enters and exactly one edge leaves a node (depot or task), when the node is assigned to a robot. Finally, (C.5) represents the sub-tour elimination constraints (SEC). The SEC ensures that no sub-tour that does not include the starting node is taken. The number of SECs grows exponentially with the number of nodes. The reader is referred to [1] for more information regarding SECs.

3. Benders decomposition

The idea of the Benders decomposition is to take advantage of the problem structure to facilitate the solution procedure. As an example, consider a MILP with a mixture of continuous (easy) and integer (difficult) variables. To decompose the MILP, we construct an auxiliary problem by selecting the constraints including only the difficult variables. This auxiliary problem can be considered a relaxation to the original MILP, and is formally called the *master* problem. Next, the solution of the master problem is used to construct a second auxiliary problem, by fixing the difficult variables in the MILP to the solution values. This second auxiliary problem is called the *slave* problem (or sub-problem), which is now easier to solve due to lack of difficult variables (it is LP). The solution of the slave problem is then used to generate a special constraint called the Benders *cut*. The cut is then added to the master problem. The master problem is now re-optimized, and this process is repeated. Notice that the objective function value of the slave problem yields an upper bound for the main problem, while solving the master problem generates a lower bound. The optimality is achieved when the upper and lower bounds meet.

We utilize the logic-based Benders decomposition [7] in which sub-problems may take any other form besides LP. In particular, applying this method to HMVRP results in a master MILP problem, and a cluster of MILP sub-problems in form of TSPs. The master MILP treats the task allocation problem, while the sub-problems determine the sequencing of the allotted tasks for each robot.

3.1. Master problem: the task assignment

The role of the master problem for HMVRP is to find a solution to the task assignment problem. Hence, the family of constraints solely composed of x_{ir} (constraints (C.2), (C.3) and (C.7)) are included there. Other constraints to add are the Benders cuts (discussed in Section 3.3), and relaxations of the sub-problems, if available. The solution of the master problem results in generating a set of sequencing TSP sub-problems (Section 3.2). As mentioned in Section 1, the objective function in the centralized formulation comprises minimizing the maximum make-span λ. So, the compatible master problem structure is chosen from [6]. This leads to the following MILP:

$$
\begin{array}{lll}
min\ z & & \\
\text{s.t.} & & \\
x_{rr} = 1 & \forall r \in \mathcal{N} & \text{(M.1)} \\
\sum\limits_{r \in \mathcal{N}} x_{ir} = 1 & \forall i \in \mathcal{K} & \text{(M.2)} \\
Benders\ Cuts & & \text{(M.3)} \\
Optional\ relaxations & & \text{(M.4)} \\
x_{ir} \in \{0,1\} & \forall i \in \mathcal{V}, \forall r \in \mathcal{N} & \text{(M.5)}
\end{array}
$$

Notice that z is an auxiliary continuous variable that appears in the cuts, and its optimal value in each iteration yields a lower bound for the make-span.

3.2. Slave problems: the sequencing TSPs

To formulate each sub-problem, one robot and the set of its allocated tasks are considered. Note that our HMVRP is assumed to be collision-free, so any assignment to the master problem leads to a number of fully decoupled sub-problems. A sub-problem corresponding to robot R_r is denoted by $\mathbb{S}_r$. Suppose that the solution set of the master problem in the l^{th} iteration is denoted by $\mathcal{X}^l$. Then, the following notations for $\mathbb{S}_r$ are defined:

- $\mathcal{N}_r$: The single-member set of the depot node of R_r.
- $\mathcal{K}_r^l$: The set of tasks assigned to robot R_r.
- $\mathcal{V}_r^l = \mathcal{N}_r \cup \mathcal{K}_r^l$: The set of all nodes.
- $\mathcal{E}_r^l = \mathcal{V}_r^l \times \mathcal{V}_r^l$: The set of directed edges.
- $x_{ir}^l = 1$: Solution to the assignment variable x_{ir} in the l^{th} iteration, where $(x_{ir}^l \in \mathcal{X}^l)$, $(i \in \mathcal{K}_r^l)$, $(r \in \mathcal{N}_r)$; i.e. the task i will be performed by R_r.

Then, the single TSP in $\mathbb{S}_r$ can be formulated as:

$$
\begin{array}{lll}
min\ \lambda_r^l & & \\
\text{s.t.} & & \\
\sum\limits_{i \in \mathcal{K}_r^l} x_{ir}^l c_{ir} + \sum\limits_{(i,j) \in \mathcal{E}_r^l} d_{ijr} y_{ijr} < \lambda_r^l & \forall r \in \mathcal{N}_r & \text{(S.1)} \\
\sum\limits_{j \in \mathcal{V}_r^l} y_{ijr} = \sum\limits_{j \in \mathcal{V}_r^l} y_{jir} = x_{ir}^l & \forall i \in \mathcal{V}_r^l, & \text{(S.2)} \\
 & \forall r \in \mathcal{N}_r & \\
\sum\limits_{i \notin \mathcal{S}} \sum\limits_{j \in \mathcal{S}} y_{ijr} \geq x_{qr}^l & \forall \mathcal{S} \subseteq \mathcal{K}_r^l, & \\
 & \forall q \in \mathcal{S}, & \\
 & \forall r \in \mathcal{N}_r & \text{(S.3)} \\
\lambda_r^l \in \mathbb{R} & & \text{(S.4)} \\
y_{ijr} \in \{0,1\} & \forall r \in \mathcal{N}_r, & \\
 & \forall (i,j) \in \mathcal{E}_r^l & \text{(S.5)}
\end{array}
$$

Notice that in constraint (S.1), the first left-hand side term is now a constant, and the same is true for the right-hand sides of (S.2) and (S.1). The sub-problem now resembles the classic asymmetric TSP for which an extensive body of research exists. It is later solved by a traditional branch and bound method, as well as by a dedicated TSP solver.

3.3. Benders cuts

The Benders cuts are arguably the most important ingredient of the Benders decomposition algorithm. The cuts are generated using the solution of the sub-problems, and then added to the master problem to direct the search. Notice that the cuts become available to the master problem *after* the first iteration, when the early solutions of the sub-problems have been obtained. In the classic Benders method [3] the cuts are generated using the solution of the dual variables corresponding to sub-problem constraints. However, in the Logic-based Benders method [7], sometimes the cuts can be formulated without them. We implement a cut formulation that is frequently used in planning and scheduling problems, [6], [7]. Defining $\mathcal{J}_r$ as the set of tasks assigned to robot R_r, the cut then becomes:

$$z \geq \lambda_r^* \Big(\sum_{i \in \mathcal{K}_r} x_{ir} - |\mathcal{J}_r| + 1 \Big)$$

where λ_r^* is the optimal objective function value of the r^{th} sub-problem, and $|\mathcal{J}_r|$ is the number of tasks assigned to robot R_r. It means that if exactly the same set of tasks are assigned to machine R_r, the make-span of this particular sub-problem will be at least λ_r^*. So, in order to obtain a shorter make-span, the solver should avoid allocating the same set of tasks to robot R_r.

3.4. The TSP solver

As mentioned earlier, the use of Benders methods for HMVRP results in sub-problems in form of asymmetric TSPs. We make use of a freely available TSP solver called *tsp_solve* [8], which was written in C++ and can efficiently solve problems of 1000 nodes using heuristic algorithms, and 150 nodes using exact methods. An advantage of Benders decomposition is that the sub-problems do not need to be solved to optimality (better to do so in general); this allows using heuristic solvers for large sub-problems, but at the cost of loosing the optimality guarantee.

4. Benders/Gossip algorithm

The gossip algorithm for HMVRP was first proposed in [5]. According to the gossip rule, after an initial task assignment, two robots and their tasks are randomly picked to form a local optimization problem. The solution to this problem yields either an equal or a better objective function value by retaining or changing the task assignments among the robots. Then, the same process is repeated for another couple of robots, until no further improvement is achieved.

Franceschelli et al. [5], used the centralized method for solving the local problems in their MILP-gossip algorithm. We, however, take advantage of the method developed in Section 3. We use LBBD as a tool to incorporate a TSP solver into the local optimization. Further, we investigate the effect of increasing the size of the local problems by selecting more than two robots. The results of this extension are presented in Section 5.2.

5. Numerical simulations

We have evaluated our exact and heuristic methods for different configurations of tasks and robots. The number of robots R was changed between 2 and 5, and for each R between 5 to 20 tasks T in increments of 3 tasks were considered. Each of these scenarios was randomly instantiated 10 times, and the average CPU time was computed. Costs of task execution for robots c_{ir}, costs of movements d_{ijr} and positions of depots and tasks in the space, were randomly generated as described in [5]. In addition, time limits of 9000 [s] and 600 [s] were considered for exact and heuristic methods respectively. The solver used was IBM ILOG CPLEX 12.4 (32-bit) in Microsoft Windows 7 Enterprise environment. The hardware used was an Intel Core2 Quad CPU (2.66 GHz), and 4 GB of RAM.

5.1. The exact methods

The three exact methods addressed in this paper are the centralized (*Centralized*) , pure Benders (*p-B*), and Benders with the TSP solver (*B-TSP*). The complexity results of the exact methods are presented in Table 1.

The comparison between *Centalized* and *p-B* reveals that the centralized method outperforms the pure Benders when problem size is relatively small. But as the number of tasks increases, the extra computational overhead of Benders is compensated by the noticeable gain in speed, and it starts performing better than the centralized method. In addition, it is clear that the use of Benders allows solving larger problems before running out of memory (O.M). For the case of B-TSP, it is observed that B-TSP is generally faster than the centralized and p-B, thanks to the use of *tsp_solve* for the sub-problems.

In the centralized case, it was not possible to solve instances larger than $(R = 2, K = 13)$, mainly due to explosion of the number of SECs, and insufficient memory. The number of SECs is upper bounded by $nk2^k$ as mentioned in [5]. In the pure Benders method, we were able to increase the size of the problem up to 19, where we reached the time limit of 2.5 hours. For larger instances, the same memory problem occurred, since each sub-problem was in form of a TSP with SECs. However, the use of the TSP solver in B-TSP effectively managed the SECs, as there was no further problem with explosion of the number of constraints. Although the *tsp_solve* was able to optimally solve the sub-problems in milliseconds, the limiting factor was in the master problem. This time, the master problem had difficulty with finding task assignments especially when k exceeded 20, and it was due to the increasing number of the cuts. This became more evident when n was also increased, which in turn, had a direct effect in size of the master problem. A similar issue in the master problem has been reported in [16]. This problem can probably be alleviated by using stronger Benders cuts.

Two of the scenarios are displayed in semi-logarithmic scale; the case of 2 robots, and 5 robots, in Figure 1. As seen in the figure, as the number of robots increases, the intersection point between the curves of the pure Benders and centralized solution occurs earlier (bottom). This is because more robots means having smaller TSPs in the sub-problems, which makes it easier for the CPLEX solver to handle them. Therefor, p-B starts outperforming the centralized earlier.

5.2. Heuristic methods

This section presents: (1) an analysis of the complexity of the Benders/gossip algorithm, and its comparison with the centralized gossip method; (2) the results for the extended Benders/gossip algorithm. The following notations have been used to denote different approaches: *G-Centralized*, *G-p-B*, and *GB-TSP-Lx*, which in turn indicate the centralized gossip, gossip with the pure Benders, and gossip with Benders and the TSP-solver. Note that x in GB-TSP-Lx shows the number of local problems, and

size		Centralized		p-B		B-TSP		GB-TSP-LP2		GB-TSP-LP3		GB-TSP-LP4	
R	T	#uns.	Sec	#uns.	Sec	#uns.	Sec	Gap%	Sec	Gap%	Sec	Gap%	Sec
2	5	0	0.29	0	1.95	0	0.14						
	8	0	1.15	0	19.99	0	0.39						
	11	0	18.58	0	78.38	0	2.09						
	14	10	O.M	0	228.16	0	47.89						
	17	10	O.M	0	3313.31	0	1303.78						
	20	10	O.M	10	O.M	10	O.T						
3	5	0	0.23	0	0.89	0	0.08	0.19	0.92				
	8	0	2.33	0	8.79	0	0.25	0.37	1.36				
	11	0	137.16	0	32.95	0	2.68	2.79	1.83				
	14	10	O.M	0	157.05	0	62.49	1.47	5.39				
	17	10	O.M	2	4351.64^{+}	1	2745.05^{+}	0.39	54.48				
	20	10	O.M	10	O.M	10	O.T	--	--				
4	5	0	0.21	0	0.50	0	0.08	0.00	3.78	0.00	0.79		
	8	0	5.68	0	2.61	0	0.22	0.22	5.89	0.22	1.12		
	11	0	581.19	0	14.77	0	2.41	2.09	5.01	0.00	2.64		
	14	10	O.M	0	117.64	0	76.33	1.05	8.65	0.26	12.84		
	17	10	O.M	2	2847.42^{+}	2	2801.89^{+}	2.12	10.04	0.97	59.39		
	20	10	O.M	10	O.M	10	O.T	--	--	--	--		
5	5	0	0.11	0	0.30	0	0.11	0.00	7.69	0.30	2.29	0.00	1.09
	8	0	7.11	0	1.49	0	0.18	1.16	11.56	0.14	2.83	0.14	0.92
	11	10	1427.20	0	7.49	0	1.26	4.54	9.82	0.57	9.16	0.00	2.00
	14	10	O.M	0	63.99	0	42.24	2.35	9.50	0.78	67.83	0.00	15.69
	17	10	O.M	1	2566.85^{+}	2	2699.08^{+}	1.03	27.09	0.00	64.03	0.00	156.63
	20	10	O.M	10	O.M	10	O.T	--	--	--	--	--	--

Table 1. Comparison of CPLEX, pure Benders (p-B), Benders with the TSP solver (B-TSP), gossip based on B-TSP with 2, 3 and 4 local problems (GB-TSP-LP2), (GB-TSP-LP3) and (GB-TSP-LP4): Number of unsolved instances (#uns.), artificial gap (Gap %), and average CPU time in seconds (Sec). O.M and O.T indicate out of memory and out of time respectively, and ”$^{+}$” indicates that the reported average time does not include the O.T cases.

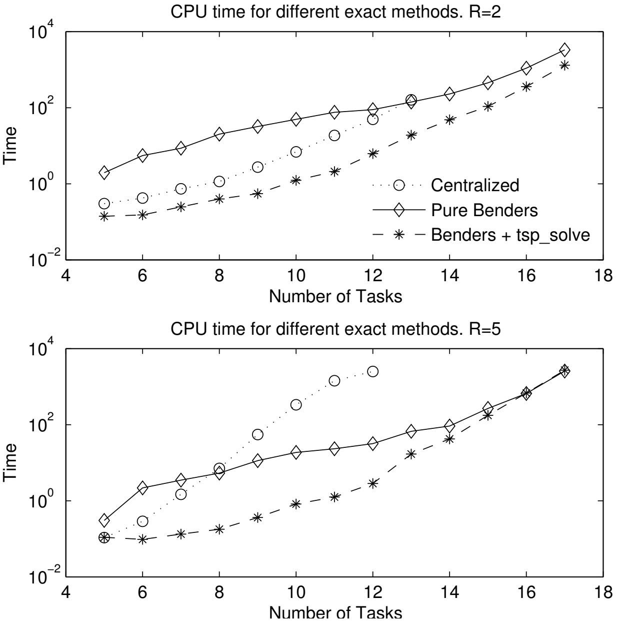

Figure 1. Results of the two scenarios: R=2 and R=5, with the the exact methods.

that it can be increased up to $R-1$ where R is the number of robots. Moreover, to indicate the quality of solutions, an artificial gap is defined using the exact solutions obtained in Section 5.1. The gap is the relative error (in %) between the optimal value and the best value from gossip in a time limit of 10 minutes.

The comparison between the gossip-based methods are given in Table 2. We used LBBD as a tool to integrate the TSP solver into the local problems. As it is evident in G-Centralized and G-p-B columns, the pure usage of LBBD has no advantage over the centralized gossip method in terms of CPU time, but when the TSP solver is added as a third layer, the GB-TSP-LP2 performs faster than both G-Centralized and G-p-B, up to several orders of magnitude. Notice that in G-Centralized, for the case of $(R = 3, K = 17)$, most of the instances could not converge to their optimal or near optimal values in the defined time limit of 600 [s] (10 minutes), and the average gap was 11.58%.

Another important advantage of GB-TSP-LP2 over G-Centralized besides its speed is that it made it possible to solve larger problems without any memory problems (although no guarantee on the optimality). But with the G-Centralized, the size of the manageable problem was very

size		G-Centralized	G-p-B	GB-TSP-LP2
R	T	Sec	Sec	Sec
3	5	1.61	2.91	0.92
	8	61.65	22.42	1.36
	11	6.22	85.37	1.83
	14	32.33	183.89	5.39
	17	OT	304.65	54.48
4	5	3.93	4.99	3.78
	8	7.24	15.68	5.89
	11	64.15	84.35	5.01
	14	13.22	166.71	8.65
	17	210.46	418.92	10.04
5	5	6.28	9.46	7.69
	8	15.52	16.88	11.56
	11	73.38	29.46	9.82
	14	72.31	117.07	9.50
	17	37.69	267.50	27.09

Table 2. Comparison of Centralized-Gossip (Centralized-G), Gossip with pure Benders (G-p-B), Gossip with Benders and the TSP solver (GB-TSP-LP2): Average CPU time in seconds (Sec), and OT for out of time.

sensitive to the size of the local problems. This resembles the memory issue with the exact centralized method, as described in Section 5.1.

Next, we compare the performance of Benders/gossip algorithm with the exact methods, and investigate the performance of the extended gossip algorithm. See Table 1 for the results. Extending the gossip algorithm has an interesting effect on the quality of the solutions. The bigger the number of local problems, the better the approximation of the original B-TSP. In other words, if $x \to R$, then $\lambda^*_{GB-TSP-LPx} \to \lambda^*_{B-TSP}$, where x is the number of local problems in GB-TSP-Lx. This can be seen by comparing the gap in GB-TSP-LP3, LP4 and LP5 for the case of $R = 5$. As more robots are included in the local problems, the optimality gap decreases, and the solution becomes more reliable. Of course, opting for more reliability, naturally affects the CPU time, but the increased cost could be reasonable. Therefore, one can adopt some strategy to find a balance between the desired solution reliability, and the computation time.

6. Conclusion

We used the logic-based Benders decomposition to reformulate the HMVRP into an assignment master problem, and a set of sequencing sub-problems. A TSP solver was integrated in the sub-problem layer, and improved the efficiency of the LBBD further. We also addressed the gossip method and improved its efficiency in two ways; (1) we achieved superior performance in the Benders/gossip algorithm, compared to the centralized MILP-gossip method. Hence, the improved procedure may be used for finding an approximation of the optimal value quickly. This was carried out by using the LBBD as a tool to include the TSP solver in the local problems; (2) by extending the number of local problems in gossip, we generated better approximation of the exact solutions. This has the potential to yield solutions trading off between the reliability and computation time. In the future, we will investigate heuristic methods for selecting the gossip local problems, instead of choosing them randomly.

References

[1] T. Bektas. The multiple traveling salesman problem: an overview of formulations and solution procedures. *Omega*, 34(3):209–219, June 2006.

[2] T. Bektas. Formulations and benders decomposition algorithms for multidepot salesmen problems with load balancing. *European Journal of Operation Research*, 216:83–93, 2012.

[3] J. Benders. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4:238–252, 1962.

[4] J. Carlsson, D. Ge, A. Subramaniam, A. Wu, and Y. Ye. Solving min-max multi-depot vehicle routing problem. *Lectures on global optimization*, 55:31–46, 2009.

[5] M. Franceschelli, D. Rosa, C. Seatzu, and F. Bullo. Gossip algorithms for heterogeneous multi-vehicle routing problems. *Nonlinear Analysis: Hybrid Systems*, 2013. http://dx.doi.org/10.1016/j.nahs.2013.03.001.

[6] J. Hooker. *Integrated methods for optimization*. Springer, New York, NY, 2010.

[7] J. Hooker and G. Ottosson. Logic-based benders decomposition. *Math. Program.*, 96:33–60, 2003.

[8] C. Hurwitz and R. Craig. *GNU tsp_solve version 1.4*. Computer software, http://www.or.deis.unibo.it/research_pages/tspsoft.html.

[9] G. Laporte. The traveling salesman problem: An overview of exact and approximate algorithms. *European Journal of Operational Research*, 59(2):231–247, 1992.

[10] G. Laporte. The vehicle routing problem: An overview of exact and approximate algorithms. *European Journal of Operational Research*, 59(3):345–358, 1992.

[11] G. Laporte and Y. Nobert. Models, relaxations and exact algorithms for the capacitated vehicle routing problem. *Annals of Discrete Mathematics*, 31:147–184, 1987.

[12] J. K. Lenstra and A. H. G. Rinnooy Kan. Complexity of vehicle routing and scheduling problem. *Networks*, 11:221–228, 1981.

[13] A. J. Orman and H. P. Williams. A survey of different integer programming formulations of the travelling salesman problem. *Advances in Computational Management Science*, 9:91–104, 2007.

[14] P. Toth and D. Vigo. Models, relaxations and exact algorithms for the capacitated vehicle routing problem. *Discrete Applied Mathematics*, 123:487–512, 1992.

[15] P. Toth and D. Vigo. *The Vehicle Routing Problem*. SIAM, Monographs on Discrete Mathematics and Applications, Philadelphia, 2002.

[16] T. Tran and J. Christopher Beck. Logic-based benders decomposition for alternative resource scheduling with sequence dependent setups. *ECAI*, pages 774–779, 2012.