A Three-Stage Matheuristic for Multi-vehicle Inventory Routing Problem

Shihao Huang,Yunhao Li,Zhipeng Lü,Zhouxing Su,Junwen Ding
School of Computer Science and Technology
Huazhong University of Science and Technology

Email of Corresponding Author: suzhouxing@hust.edu.cn

2022/01/30

Abstract: The multi-vehicle inventory routing problem (MIRP) tackles the combination of inventory management and vehicle routing, which is a challenging NP-hard optimization problem. It seeks a minimum-cost solution which utilizes a fleet to perform deliveries in multiple periods, ensuring that no customer runs out of stock. We propose a three-stage matheuristic (TSMH) consists of three optimization stages to solve the MIRP. The first stage optimizes the global delivery schedule and generates a feasible initial solution by a relax-and-repair method. The second stage improves the initial solution by iteratively improving the local structure of the delivery schedule. The last stage adopts a solution-based tabu search for the MIRP to implement a detailed optimization. The experimental results on commonly-used instances show that the proposed algorithm can find new upper bounds on 100 out of the 640 small instances and 192 out of the 240 large ones. These results demonstrate that the TSMH is competitive in solving the large-scale MIRP.

Team name: [SmartLab]
Solver name: [TSMHA]
VRP tracks: [IRP]

1 Introduction

The MIRP is defined on an complete undirected graph $G = (V, E)$. $V = \{0, 1, ..., n\}$ is the set of vertices, where vertex 0 stands for the depot and $C = V - \{0\}$ represents the client set. $E = \{(i, j) : i, j \in V, i \neq j\}$ denotes the edge set, where each edge (i, j) is associated with a traveling cost c_{ij}. There are p periods in the planning horizon. A fleet K of m homogeneous vehicles each with capacity Q is available to provide deliveries. The vehicles must stop at the depot, which means that each route begins and ends at the depot. In each period $t \in T = \{1, ..., p\}$, r_0^t units of commodity are produced at depot, while r_i^t units consumed by client $i \in C$. Note that the production and delivery happen before loading and consumption at each vertex, respectively. Throughout the planning horizon, each vertex $i \in V$ must maintain an inventory level of at least L_i and at most U_i, and will lead to a per-period cost of h_i for each unit of inventory held.

In order to represent a solution, we use variable q_i^{kt} to denote the quantity of the commodity delivered to vertex $i \in V$ by vehicle $k \in K$ in period $t \in T$. Binary variable x_{ij}^{kt} equals to 1 if edge (i, j) exists in the delivery route of vehicle $k \in K$ in period $t \in T$. Binary variable y_i^{kt} equals to 1 if vehicle $k \in K$ visits vertex $i \in V$ in period $t \in T$. Binary variable z_i^{kt} equals to 1 if vehicle $k \in K$ only visits vertex $i \in V$ in period $t \in T$. Let variable I_i^t be the inventory level of vertex $i \in V$ at the end of period $t \in T \cup \{0\}$, except that the initial inventory level I_i^0 is already known. Then, we formulate the MIRP as the following mixed-integer programming (MIP) model.

$$
\begin{aligned}
min \quad & \textstyle\sum_{t\in T}\sum_{i\in V} h_i I_i^t + \sum_{t\in T}\sum_{k\in K}\sum_{(i,j)\in E} c_{ij} y_{ij}^{kt} + \sum_{t\in T}\sum_{k\in K}\sum_{i\in C} c_{0i} z_i^{kt} && (1)\\
s.t. \quad & I_0^t = I_0^{t-1} + r_0^t - \textstyle\sum_{k\in K}\sum_{i\in C} q_i^{kt}, \forall t \in T && (2)\\
& I_t^t = I_i^{t-1} + \textstyle\sum_{k\in K} q_i^{kt} - r_i^t, \forall i \in C, \forall t \in T && (3)\\
& \textstyle\sum_{k\in K} q_i^{kt} \leq U_i - I_i^{t-1}, \forall i \in C, \forall t \in T && (4)\\
& q_i^{kt} \leq U_i y_i^{kt}, \forall i \in C, \forall k \in K, \forall t \in T && (5)\\
& \textstyle\sum_{i\in C} q_i^{kt} \leq Q y_0^{kt}, \forall k \in K, \forall t \in T && (6)\\
& \textstyle\sum_{k\in K} y_i^{kt} \leq 1, \forall i \in C, \forall t \in T && (7)\\
& \textstyle\sum_{j\in V, i<j} x_{ij}^{kt} + \sum_{j\in V, j<i} x_{ji}^{kt} + 2z_i^{kt} = 2y_i^{kt}, \forall i \in C, \forall k \in K, \forall t \in T && (8)\\
& \textstyle\sum_{j\in C} x_{0j}^{kt} + 2\sum_{j\in C} z_j^{kt} = 2y_0^{kt}, \forall k \in K, \forall t \in T && (9)\\
& \textstyle\sum_{(i,j)\in S, i<j} x_{ij}^{kt} \leq |S| - 1, \forall S \text{ is a subtour}, |S| \geq 3, \forall k \in K, \forall t \in T && (10)\\
& q_i^{kt} \geq 0, i \in C, k \in K, t \in T && (11)\\
& L_i \leq I_i^t \leq U_i, i \in V, t \in T && (12)\\
& y_i^{kt} \in \{0, 1\}, i \in V, k \in K, t \in T && (13)\\
& x_{ij}^{kt} \in \{0, 1\}, i, j \in V, i < j, k \in K, t \in T && (14)\\
& z_i^{kt} \in \{0, 1\}, i \in C, k \in K, t \in T && (15)
\end{aligned}
$$

2 Three-Stage Matheuristic

2.1 Stage 1: Global Structural Optimization

We consider the MIRP from such a perspective that the delivery schedule (the timing of visits and the delivery quantities) is the skeleton of a solution: Once a feasible delivery schedule is determined, the remaining subproblems (routing for each period) are mutually independent. Thus, the goal of the first stage is to figure out a promising global structure of the delivery schedule, and generates a feasible initial solution so that the remaining stages can starts from a high-potential region of the solution space. Due to the factorial complexity of the subtour elimination constraints (10), relaxing them will greatly reduce the intractability of the induced subproblem. Therefore, we obtain an relatively easy subproblem which mainly optimizes the delivery schedule for each customer while considering the routing as accurate as possible. Namely, once a relaxed solution is found, for each period, we randomly pick one of the involved subtours with no more than α edges, and add a lazy constraint to eliminate this subtour. Besides, we relax constraints (6) that the commodity delivered to each customer is loaded by a specific vehicle in each period, so that the routing can be repaired as the classical capacitated vehicle routing problem (CVRP). We refer to this relaxed model as *GlobalRel* and it is as follows:

$$\begin{aligned}
min \quad & \textstyle\sum_{t\in T}\sum_{i\in V} h_i I_i^t + \sum_{t\in T}\sum_{k\in K}\sum_{(i,j)\in E} c_{ij} y_{ij}^{kt} + \sum_{t\in T}\sum_{k\in K}\sum_{i\in C} c_{0i} z_i^{kt} && (16)\\
s.t. \quad & I_0^t = I_0^{t-1} + r_0^t - \textstyle\sum_{i\in C} q_i^t, \forall t\in T && (17)\\
& I_t^t = I_i^{t-1} + q_i^t - r_i^t, \forall i\in C, \forall t\in T && (18)\\
& q_i^t \le U_i - I_i^{t-1}, \forall i\in C, \forall t\in T && (19)\\
& q_i^t \le U_i \textstyle\sum_{k\in K} y_i^{kt}, \forall i\in C, \forall t\in T && (20)\\
& \textstyle\sum_{i\in C} q_i^t \le Q\sum_{k\in K} y_0^{kt}, \forall t\in T && (21)\\
& \textstyle\sum_{k\in K} y_i^{kt} \le 1, \forall i\in C, \forall t\in T && (22)\\
& \textstyle\sum_{j\in V, i<j} x_{ij}^{kt} + \sum_{j\in V, j<i} x_{ji}^{kt} + 2z_i^{kt} = 2y_i^{kt}, \forall i\in C, \forall k\in K, \forall t\in T && (23)\\
& \textstyle\sum_{j\in C} x_{0j}^{kt} + 2\sum_{j\in C} z_j^{kt} = 2y_0^{kt}, \forall k\in K, \forall t\in T && (24)\\
& q_i^t \ge 0, i\in C, k\in K, t\in T && (25)\\
& L_i \le I_i^t \le U_i, i\in V, t\in T && (26)\\
& y_i^{kt} \in \{0,1\}, i\in V, k\in K, t\in T && (27)\\
& x_{ij}^{kt} \in \{0,1\}, i,j\in V, i<j, k\in K, t\in T && (28)\\
& z_i^{kt} \in \{0,1\}, i\in C, k\in K, t\in T && (29)
\end{aligned}$$

We can can decide whether a customer i is visited or not in period t by examining $Z_i^t = \sum_{k\in K} y_i^{kt}$. Hence, in each period, when the delivery quantities and the visited customers are fixed, repairing the routes is equivalent to solving the CVRP on the subgraph composed of the visited customers and the depot.

2.2 Stage 2: Local Structural Optimization

When the quality of the best found solution gradually converges, a proper global structure of the delivery schedule is formed. However, there is usually large room of improvement for the best solution found so far, especially on the large instances. Thus, the TSMH algorithm moves on to the second stage which aims to iteratively improve the local structure of the delivery schedule.

In this stage, we improve only a part of the delivery schedule at a time while preserving the global structure of the delivery schedule. Specifically, the TSMH algorithm improves the best found solution by iteratively solving the relaxed model called *LocalRel*. The constraints of a specific model *LocalRel* consist of two parts. The first part inherits model *GlobalRel* for a series of consecutive periods $T_{con} = \{p_{start}, p_{start}+1, ..., p_{start}+l_{con}-1\}$. Here, parameter l_{con} controls the size of these consecutive periods. In the second part, for the periods not considered in T_{con}, we exclude the routing constraints but preserve the complete inventory management constraints (2)-(4), (11)-(12). In addition, we introduce the following constraints to enlarge the success rate of the CVRP-based repairing procedure, where Y_i^{kt} denotes if vertex i is visited by vehicle k in period t which is known for each specific (best found) solution.

$$q_i^{kt} \leq Y_i^{kt} U_i, \forall i \in V, \forall k \in K, \forall t \in T - T_{con} \tag{30}$$

$$\sum_{i \in C} Y_i^{kt} q_i^{kt} \leq Q, \forall k \in K, \forall t \in T - T_{con} \tag{31}$$

As a result, the objective of model *LocalRel* is:

$$min \quad \sum_{t \in T} \sum_{i \in V} h_i I_i^t + \sum_{t \in T_{con}} \sum_{k \in K} \sum_{(i,j) \in E} c_{ij} y_{ij}^{kt} + \sum_{t \in T_{con}} \sum_{k \in K} \sum_{i \in C} c_{0i} z_i^{kt} \tag{32}$$

In model *LocalRel* under a specific T_{con}, the delivery schedule of periods in T_{con} is completely re-optimized, as well as the involved routes. However, for the remaining periods, the vehicle routes and timing of visits are both fixed. The second stage performs $NumIter$ iterations. For each iteration, according to parameter l_{con}, the TSMH algorithm will progressively generate T_{con} from the first l_{con} periods to the last l_{con} ones. For each T_{con}, a corresponding model *LocalRel* is constructed and initialized with the best found solution. Then, we solve each model *LocalRel* by the same method as in the first stage within the time limit, including repairing routes for periods in T_{con}, adding subtour elimination constraints, and updating the best found solution.

2.3 Stage 3: Detailed Optimization

In this stage, we adopt a solution-based iterated tabu search to implement a detailed optimization for the MIRP. This search is similar to the one for the single-vehicle IRP [2]. Here, we adapt the neighborhood structures and their evaluation for MIRP. Specifically, we define four neighborhood structures which manipulate the timing of visits to customers (Z_i^t):

- Addition operation $M_a(t, i)$. Visit unvisited customer i in period t.

- Removal operation $M_r(t, i)$. Cancel the existing visit to customer i in period t.
- Move operation $M_m(t_1, t_2, i) = M_a(t_1, i) + M_r(t_2, i)$. Visit customer i in period t_1 if it is not visited, and simultaneously cancel the existing visit to the same client in period t_2.
- Swap operation $M_s(t_i, t_j, i, j) = M_m(t_j, t_i, i) + M_m(t_i, t_j, j)$. Exchange the visits to a pair of customers in a pair of periods where $Z_i^{t_i} = Z_j^{t_j} = 1, Z_i^{t_j} = Z_j^{t_i} = 0$.

Next, the TSMH algorithm evaluate the holding costs for the top-π_e elite moves. For a specific move M, although Z_i^t does not indicate which vehicle is responsible for customer i, we can figure this out from the routes in period t. Thus, we can check the feasibility and determine the holding cost by solving the linear programming model LP which consists of constraints (2)-(4), (11)-(12), (14), and (33)-(35). Here, y_0^{kt} and z_i^{kt} are calculated according to the routes after making move M.

$$\min \quad \sum_{t\in T} \sum_{i\in V} h_i I_i^t \tag{33}$$

$$q_i^{kt} \le U_i(y_0^{kt} + \sum_{i\in C} z_i^{kt}), \forall i \in C, \forall k \in K, \forall t \in T \tag{34}$$

$$\sum_{i\in C} q_i^{kt} \le Q(y_0^{kt} + \sum_{i\in C} z_i^{kt}), \forall k \in K, \forall t \in T \tag{35}$$

Technically, Algorithm 1 describes the detailed optimization. It starts from the best found solution in the previous stages (line 1). For each iteration, the search constructs the neighborhood for S_{cur}, evaluates each neighboring solution, and moves to the best one (line 4-9). In order to speed up the search, we reformulate model LP to the minimum-cost flow problem to minimize the holding costs. Then, all the neighboring solutions whose routing and holding costs are both evaluated, i.e., the infeasible moves and the top π_e elite moves, are prohibited to be visited anymore (line 6). Next, we further refine the involved routes by canceling the zero-quantity deliveries and calling the CVRP solver (line 8). If S_{best} is not improved within π_{step} iterations, we perform perturbation on S_{best} (line 11) and repeat the above procedures until the time limit is reached (line 2).

3 Computational Experiments

3.1 Experimental Protocol

The proposed TSMH algorithm is evaluated on two sets of commonly-used Euclidean datasets, which consists of 800 small instances and 240 large ones. Our TSMH algorithm is programmed in C++ and tested on Intel Xeon E5-2698 v3 2.30GHz CPU. MIP models *GlobalRel* and *LocalRel* are solved by Gurobi 9.1.1[1], the minimum-cost flow model is solved by LEMON[2], and the CVRP subproblems are solved by EAX memetic algorithm [4] implemented by ourselves. Following the rule of the IRP track of the 12th DIMACS Implementation Challenge[3], according to the CPU we

[1]https://www.gurobi.com/
[2]https://lemon.cs.elte.hu/trac/lemon
[3]http://dimacs.rutgers.edu/programs/challenge/vrp/irp/

Algorithm 1 Detailed Optimization

Require: I: instance, S_{best}: best found solution in the previous stages

$S_{cur} \leftarrow S_{best}$, $step \leftarrow 1$
while time limit is not reached **do**
  while $step \leq \pi_{step}$ **do**
    $M_{all} \leftarrow constructNonTabuNeighrbood(S_{cur})$
    $M_{best} \leftarrow evaluate(M_{all})$
    $selectivelyTabu(M_{all})$
    update S_{cur} with M_{best}
    $refineInvolvedRoutes(S_{cur})$
    update S_{best} with S_{cur} if S_{cur} is better than S_{best}
  end while
  $S_{cur} \leftarrow perturb(S_{best}, \pi_a, \pi_r, \pi_m)$, $step \leftarrow 1$
end while

Ensure: S_{best}

used, the TSMH algorithm is tested on a single thread with 1817-second wall clock time limit on each instance. We perform 31 independent runs for each instance and record the best results. The time limit for models *GlobalRel* (in the first stage) and *LocalRel* (in the second stage) are 120 and 60 seconds, respectively.

3.2 Computational Results

In this section, we compare the proposed TSMH algorithm with the best known solution (BKS) obtained by existing algorithms including both exact methods and heuristics. Besides, we compare the results of TSMH with both the best objective values obtained by heuristics, and the objective values of the state-of-the-art matheuristic KS-MIRP [3]. All the best known results are directly collected from [3]. Since their statistics do not include the small instances with 6 periods and over 30 customers, the comparison involves only the instances reported in previous studies.

The overall computational results of the small instances are shown in Table 2 as the way in [1]. The results are divided into two parts according to the holding cost. The statistics of each instance group comprises four columns. The first column reports the value of period number p or vehicle number m of the instances aggregated in the row. The second column reports the number of new upper bounds (UB) found by TSMH in 31 independent runs. The third (fourth) column gives the percentage gaps of the best (average) objective values of TSMH with respect to the objective values of the BKS, the best results of heuristics (BestHeu), and the best results of KS-MIRP (KS), respectively. Furthermore, we report the average (Avg), minimum (Min), and maximum (Max) gaps among the instances in each instance group. We denote our best and average objective values as BestTSMH and AvgTSMH respectively. The values of the percentage gaps in Table 2 are calculated as $(v_{TSMH} - v_{other})/v_{other} \times 100$, where v_{TSMH} is BestTSMH or AvgTSMH, and v_{other} is the objective value of BKS, BestHeu, or KS. The TSMH algorithm improves 100 of the 640 small instances (15.6%), and most of the improved instances involve 6 periods and more than 2 vehicles. For BestTSMH, the average gaps among all instances are no more than 0.05%. For AvgTSMH, the average gaps among all instances are no more 0.71%.

The overall computational results of large instances are shown in Table 3. One can observe that TSMH improves 192 of the 240 (80%) instances, including all the instances with 50 customers. For BestTSMH, the average gaps among all instances are no more than -0.54%. For AvgTSMH, the average gaps among all instances are no more than 0.49%. Especially, compared with the state-of-the-art matheuristic KS-MIRP, the average gaps of AvgTSMH are less than 0.1% for the small instances, while being all negative for the large ones.

4 Conclusion

We proposed a three-stage matheuristic which includes a global structural optimization stage, a local structural optimization, and a detailed optimization stage. Tested on commonly-used instances, the TSMH algorithm improves the best known results on 292 out of 880 instances. These statistics show that the TSMH algorithm is very competitive on solving large-scale instances.

References

[1] C. Archetti, N. Boland, and M. G. Speranza, "A matheuristic for the multivehicle inventory routing problem," *INFORMS Journal on Computing* 29(3), 377-387, 2017.

[2] Z. Su, S. Huang, C. Li, and Z. Lü, "A two-stage matheuristic algorithm for classical inventory routing problem," in Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence, 2021, 3430–3436.

[3] C. Archetti, G. Guastaroba, D. L. Huerta-Muñoz, and M. G. Speranza, "A kernel search heuristic for the multivehicle inventory routing problem," *International Transactions in Operational Research* 28, 2984-3013, 2021.

[4] Y. Nagata and O. Bräysy, "Edge assembly-based memetic algorithm for the capacitated vehicle routing problem," *Networks* 54(4), 205–215, 2009.

A Appendix: Parameter Settings

Parameter	Description	Value
$NumIter$	Number of iterations in local structural optimization	3
l_{con}	Size of consecutive periods for each iteration in local structural optimization	2, 2, 2
π_e	Maximal number of feasible elite neighborhood moves	$p\sqrt{n}\sqrt{m}$
α	Maximal number of involved edges of a subtour elimination constraint	10
β	Maximal time of repairing the routing for a relaxed solution	10
π_{step}	Maximal number of non-improving moves	20
π_{range}	Range of hash values	10^8
$\gamma_1, \gamma_2, \gamma_3$	Parameters for the hash functions	1.8, 2.4, 3.0
π_a	Number of addition operations in perturbation	$\{4, 5\}$
π_r	Number of removal operations in perturbation	$\{0, 1, 2\}$
π_m	Number of move operations in perturbation	$\{4, 5\}$
π_{pop}	Size of population in CVRP Solver	10
$\pi_{closest}$	Number of closest vertices to consider in CVRP Solver	10

Table 1: Parameter settings.

B Appendix: Statistics of Computational Experiments

			BestTSMH % gap									AvgTSMH % gap								
			w.r.t. BKS			w.r.t. BestHeu			w.r.t. KS			w.r.t. BKS			w.r.t. BestHeu			w.r.t. KS		
		UB	Avg	Min	Max	Avg	Min	Max	Avg	Min	Max	Avg	Min	Max	Avg	Min	Max	Avg	Min	Max
Low inventory cost	p=3	17	0.05	-1.35	6.21	-0.05	-4.03	6.21	-0.85	-10.03	6.21	0.46	-0.10	6.74	0.35	-1.45	6.74	-0.45	-10.01	6.74
	p=6	36	-0.02	-3.24	2.76	-0.59	-4.60	2.27	-0.77	-5.96	1.20	1.14	-1.48	4.11	0.57	-3.18	4.11	0.38	-5.21	4.11
	m=2	2	0.11	-0.59	6.21	-0.03	-1.40	6.21	-0.40	-10.03	6.21	0.55	0.00	6.21	0.41	-0.66	6.21	0.04	-10.01	6.21
	m=3	14	-0.06	-2.40	1.28	-0.23	-2.58	1.28	-0.70	-8.17	1.28	0.70	-0.38	6.74	0.53	-1.43	6.74	0.05	-7.88	6.74
	m=4	21	-0.01	-1.20	6.21	-0.34	-4.18	0.66	-1.01	-7.84	0.41	0.68	0.00	6.21	0.34	-2.84	3.46	-0.32	-7.15	3.46
	m=5	16	0.11	-0.59	2.76	-0.40	-4.60	2.27	-1.16	-7.90	0.92	0.55	0.00	3.87	0.45	-3.18	3.76	-0.32	-6.65	2.76
	All	53	0.03	-3.24	6.21	-0.25	-4.60	6.21	-0.82	-10.03	6.21	0.71	-1.48	6.74	0.43	-3.18	6.74	-0.14	-10.01	6.74
High inventory cost	p=3	12	0.06	-0.69	3.38	0.00	-2.03	3.38	-0.35	-3.96	3.38	0.32	-0.18	3.38	0.27	-0.79	3.38	-0.09	-3.60	3.38
	p=6	35	0.05	-1.37	1.69	-0.22	-2.41	1.69	-0.31	-2.58	0.96	0.74	-0.68	2.78	0.46	-1.15	2.71	0.38	-1.37	2.71
	m=2	2	0.07	-0.05	3.38	0.03	-0.84	3.38	-0.08	-1.61	3.38	0.37	0.00	3.38	0.33	-0.18	3.38	0.22	-1.53	3.38
	m=3	12	0.00	-0.87	0.79	-0.09	-1.81	0.78	-0.26	-2.33	0.78	0.44	-0.49	2.87	0.34	-1.03	2.87	0.17	-1.67	2.87
	m=4	19	0.03	-1.09	3.38	-0.14	-2.41	1.21	-0.48	-2.58	1.21	0.50	-0.37	3.38	0.32	-1.15	2.71	-0.01	-1.92	2.71
	m=5	14	0.07	-0.05	1.69	-0.12	-2.03	1.69	-0.53	-3.96	0.96	0.37	0.00	2.73	0.38	-0.79	2.54	-0.03	-3.60	2.12
	All	47	0.05	-1.37	3.38	-0.08	-2.41	3.38	-0.34	-3.96	3.38	0.48	-0.68	3.38	0.34	-1.15	3.38	0.09	-3.60	3.38

Table 2: Computational Results on 640 Tested Small Instances

			BestTSMH % gap									AvgTSMH % gap								
			w.r.t. BKS			w.r.t. BestHeu			w.r.t. KS			w.r.t. BKS			w.r.t. BestHeu			w.r.t. KS		
		UB	Avg	Min	Max	Avg	Min	Max	Avg	Min	Max	Avg	Min	Max	Avg	Min	Max	Avg	Min	Max
Low inventory cost	n=50	40	-2.86	-4.73	-1.07	-2.86	-4.73	-1.07	-4.93	-10.36	-1.07	-1.55	-3.68	1.07	-1.55	-3.68	1.07	-3.66	-9.07	1.07
	n=100	38	-2.65	-5.72	0.38	-2.65	-5.72	0.38	-7.16	-13.45	-2.58	-0.46	-3.26	2.95	-0.46	-3.26	2.95	-5.07	-11.96	0.21
	n=200	24	-0.07	-4.01	3.66	-0.07	-4.01	3.66	-9.49	-15.01	-4.28	3.49	-0.16	6.58	3.49	-0.16	6.58	-6.28	-11.08	-1.00
	m=2	29	-2.02	-4.59	0.33	-2.02	-4.59	0.33	-7.22	-13.45	-1.07	-0.07	-3.68	3.16	-0.07	-3.68	3.16	-5.38	-11.96	1.07
	m=3	27	-2.16	-4.40	0.89	-2.16	-4.40	0.89	-7.42	-13.80	-1.75	0.16	-2.92	3.90	0.16	-2.92	3.90	-5.27	-11.08	0.21
	m=4	24	-1.89	-4.65	0.33	-1.89	-4.65	1.27	-7.67	-15.01	-1.74	0.64	-2.83	3.16	0.64	-2.83	6.04	-5.32	-11.00	-0.77
	m=5	22	-2.02	-4.59	3.66	-1.36	-5.72	3.66	-6.48	-12.38	-2.07	-0.07	-3.68	6.58	1.23	-3.46	6.58	-4.05	-9.97	-0.16
	All	102	-1.86	-5.72	3.66	-1.86	-5.72	3.66	-7.20	-15.01	-1.07	0.49	-3.68	6.58	0.49	-3.68	6.58	-5.00	-11.96	1.07
High inventory cost	n=50	40	-1.27	-2.41	-0.19	-1.28	-2.41	-0.19	-2.28	-4.20	-0.47	-0.59	-1.83	0.27	-0.60	-1.83	0.27	-1.61	-3.78	0.27
	n=100	36	-0.71	-2.30	0.43	-0.71	-2.30	0.43	-2.73	-5.13	-0.73	0.22	-1.34	1.61	0.22	-1.34	1.61	-1.82	-4.53	0.65
	n=200	14	0.36	-1.54	1.62	0.36	-1.54	1.62	-2.74	-4.66	-1.54	1.30	-0.59	2.62	1.30	-0.59	2.62	-1.83	-3.58	-0.46
	m=2	28	-0.61	-1.74	0.50	-0.62	-1.74	0.50	-2.39	-3.63	-0.76	0.15	-0.91	1.16	0.13	-0.91	1.16	-1.65	-2.89	-0.27
	m=3	24	-0.73	-2.30	0.24	-0.73	-2.30	0.24	-2.85	-4.66	-0.47	0.09	-1.34	1.41	0.09	-1.34	1.41	-2.05	-3.78	0.65
	m=4	18	-0.28	-1.82	0.50	-0.28	-1.82	1.62	-2.69	-5.13	-1.06	0.57	-1.25	1.16	0.57	-1.25	2.28	-1.86	-4.53	-0.30
	m=5	20	-0.61	-1.74	1.61	-0.53	-2.41	1.61	-2.41	-4.45	-1.04	0.15	-0.91	2.62	0.44	-1.83	2.62	-1.46	-3.54	-0.10
	All	90	-0.54	-2.41	1.62	-0.54	-2.41	1.62	-2.59	-5.13	-0.76	0.31	-1.83	2.62	0.31	-1.83	2.62	-1.75	-4.53	0.65

Table 3: Computational Results on 240 Large Instances