Evolutionary Multiobjective Optimization for the Pickup and Delivery Problem with Time Windows and Demands (PDP-TW-D)

Dung H. Phan
Department of Computer Science
University of Massachusetts, Boston
Boston, MA 02125
phdung@cs.umb.edu

Junichi Suzuki
Department of Computer Science
University of Massachusetts, Boston
Boston, MA 02125
jxs@cs.umb.edu

***Abstract*—This paper studies an evolutionary algorithm to solve a new multiobjective optimization problem, the Pickup and Delivery Problem with Time Windows and Demands (PDP-TW-D), which extends PDP and PDP-TW. With respect to multiple optimization objectives, PDP-TW-D is to find a set of Pareto-optimal routes for a fleet of vehicles in order to serve given transportation requests. The proposed algorithm uses a population of individuals, each of which represents a solution candidate, and evolves them through generations to seek the Pareto-optimal solutions with respect to given multiple objectives. In addition to evolution, the proposed algorithm allows individuals to learn and improve themselves in each generation with a local search algorithm. Experimental results demonstrate that the evolutionary and learning processes complement with each other in the proposed algorithm and can effectively obtain quality solutions to PDP-TW-D.**

***Index Terms*—Evolutionary multiobjective optimization algorithms, Memetic algorithms, Pickup and Delivery Problem with Time Windows**

I. Introduction

This paper formulates a new multiobjective optimization problem, the Pickup and Delivery Problem with Time Windows and Demands (PDP-TW-D). PDP-TW-D is a combinatorial optimization problem of finding a set of optimal routes for a fleet of vehicles in order to serve given transportation requests. Each transportation request is defined by a pickup location, a delivery location, goods to be transported, a pickup time window and a delivery time window. Each of pickup and delivery locations is required to be visited within a certain time window by a vehicle. Each vehicle has a given capacity and follows an assigned route (i.e., a sequence of pickup/delivery locations) by loading goods at a pickup location and unloading them at a delivery location. PDP-TW-D is to find the optimal routes for vehicles with respect to multiple optimization objectives: the number of vehicles used to fulfill transportation requests, the total distance that vehicles travel, and the total amount of goods that vehicles transport.

In PDP-TW-D, all vehicles are required to satisfy *time window constraints*, *precedence constraints*, *pairing constraints* and *capacity constraints*. Time window constraints enforce vehicles to visit pickup and delivery locations within given time windows. Precedence constraints deal with the restriction that each pickup location has to be visited prior to its corresponding delivery location. Pairing constraints restrict that one vehicle has to do both the pickup and delivery of goods specified in a one transportation request. Capacity constraints enforce vehicles not to exceed its capacity by overloading goods.

PDP-TW-D extends two existing optimization problems, the Pickup and Delivery Problem (PDP) and the PDP with Time Windows (PDP-TW), which in turn extend the Vehicle Routing Problem (VRP). As a variant of PDP and PDP-TW, PDP-TW-D represents a number of logistics and transportation applications in airlift and sealift environments [1]–[7]. Other applications include parcel services, taxi dispatching [8], shared taxi services, school bus routing, dial-a-ride services (e.g., the transport of the elderly and handicapped among their homes, hospitals and other locations) [9]–[12], larvicide control program [13] and the transport of medical samples from medical offices to laboratories.

Given the facts that PDP and PDP-TW are NP-hard [14], PDP-TW-D is NP-hard as well. In the presence of time windows and other constraints, it is NP-complete to even examine the problem's feasibility. This means that it can take a significant amount of time, labor and costs to find the optimal solution(s) from a huge number of possible solution candidates. In fact, early studies in the area of PDP-TW, which used dynamic programming [9], [10], suffered from local optima and could solve only small-scale problem instances where less than 10 pickup and delivery locations are involved. Since then, various heuristic algorithms have been investigated to solve PDP and PDP-TW; however, the majority of them targeted small-scale problem instances as well. Currently, it is widely known that metaheuristic algorithms are more effective to solve larger-scale PDP and PDP-TW instances [15], [16].

This paper proposes and evaluates an evolutionary multiobjective optimization algorithm (EMOA) as a metaheuristic algorithm to solve PDP-TW-D. The proposed EMOA uses a population of individuals, each of which represents a solution candidate, and *evolves* them through generations to seek the Pareto-optimal solutions with respect to given multiple objectives. In addition to evolution, the proposed EMOA allows individuals to *learn* and improve themselves in each generation with a local search algorithm. Experimental results demonstrate that the evolutionary and learning processes complement with each other in the proposed EMOA and can effectively obtain quality solutions to PDP-TW-D in a relatively large-scale problem instances that have 100 pickup and delivery locations

II. Problem Statement

This paper uses the following notations to state the pickup and delivery problem with time windows and demands (PDP-TW-D).

- $P = \{1, 2, 3, ..., n\}$ is the set of pickup nodes. $D = \{n+1, n+2, n+3, ..., 2n\}$ is the set of delivery nodes. $\{0\}$ denotes the depot.
- $N = \{1, 2, 3, ..., n\}$ is the set of requests. A request i is represented by a pickup node i, a delivery node $i+n$ and a demand q_i. q_i denotes the amount of goods that are available at i to be picked up and delivered to $i+n$.
- K is the set of vehicles. $|K| = m$. Each vehicle has its capacity Q.
- For all nodes $i, j \in P \cup D$, d_{ij} denotes the travel distance from i to j. t_{ij} denotes the travel time from i to j.
- Each node $i \in P \cup D$ has an associated time window $[e_i, l_i]$, during which a vehicle is required to visit i to load or unload goods. Service time s_i is also associated with i. It is the time required to load/unload goods to/from a vehicle at i. A vehicle is allowed to arrive at i before e_i; however, it needs to wait for starting to load/unload goods until e_i.
- A pickup and delivery (PD) route for a vehicle k, R_k, is a sequence of nodes that k visits, starting and ending with the depot. Every PD route is required to satisfy three types of constraints: (1) *capacity constraints* (The amount of goods loaded on each vehicle should not exceed the capacity of the vehicle.), (2) *time window constraints* (Each vehicle should arrive at a node before the end of its time window.), and (3) *precedence constraints* (A pickup node i should be visited before its corresponding delivery node $i+n$).
- A pickup and delivery (PD) plan is a set of PD routes, $R = \{R_k | k \in K\}$. A PD plan may not fulfill all the n requests. No redundant nodes exist in R except the depot. (A non-depot node is not visited more than once.)

PDP-TW-D is to find the Pareto-optimal PD plans with respect to the following three objectives.

- The number of vehicles used in a PD plan (i.e., the number of routes in a PD plan). This objective is to be minimized. It is computed as follows:

$$f_{vehicle} = |R| \tag{1}$$

- The total travel distance. This objective is to be minimized. It is computed as follows:

$$f_{distance} = \sum_{k \in K} \sum_{i \in P \cup D} \sum_{j \in P \cup D} d_{ij} x_{ijk} \tag{2}$$

  x_{ijk} is true (1) iff the vehicle k travels from the node i to the node j; otherwise, x_{ijk} is false (0).
- The total demands. This objective is to be maximized. It represents the total amount of goods that are transported in a PD plan. It is computed as:

$$f_{demand} = \sum_{k \in K} \sum_{i \in N} q_i z_{ik} \tag{3}$$

  z_{ik} is true (1) iff the vehicle k fulfills the request i; otherwise, z_{ik} is true (0).

Objectives conflict with each other in PDP-TW-D. For example, the first two objectives conflict with the third one. A smaller number of vehicles and a shorter travel distance yields lower total demands. On the contrary, higher total demand yield a larger number of vehicles and a longer travel distance.

Constraint violations are computed as follows:

- Capacity violation of a vehicle k in a PD route R_k

$$g_c(R_k) = \sum_{i \in R_k} \Delta_i \tag{4}$$

  Δ_i denotes the amount of overloaded goods on k at i.
- Time window violation of a vehicle k in a PD route R_k

$$g_{tw}(R_k) = \sum_{i \in R_k} |t_{ik} - l_i| I_{ik} \tag{5}$$

  t_{ik} denotes the time when k arrives at i. $I_{ik} = 1$ if $t_{ik} > l_i$, otherwise $I_{ik} = 0$.

III. Related Work

Nanry and Barnes [17] is one of the first to present a meta-heuristic for PDPTW. The metaheuristic is based on a reactive tabu search. First, a feasible solution is constructed using greedy insertion method. Next, tabu search is used to improve the initial solution. Three neighborhood moves are proposed in this paper. They are: Single Pair Insertion (SPI), Swapping Pairs Between Route (SBR) and Within Route Insertion (WRI). In order to evaluate their work, the authors created PDPTW test instances from standard VRPTW problems proposed by Solomon [18].

Li and Lim [15] propose a tabu-embedded simulated annealing approach to solve PDPTW. The authors modify Solomon's insertion heuristic [18] by initializing each route with a pickup and delivery pair which satisfies some criteria (e.g. early time window interval, far distance from depot). Three different neighborhood moves (i.e. PD-Shift, PD-Swap, PD-Rearrange) are presented. The authors extend local search method to a descent local search (DLS) which tries to improve the current solution for a number of iterations. After a given number of iterations without improvement, the search is restarted from current best solution. To avoid cycling, a tabu list is used to keep track of the recently investigated solutions. Additionally, the authors generated 56 test instances for PDPTW problem based on all 56 Solomon VRPTW instances. The generated data set became the standard benchmark test data for PDPTW. Another tabu search based approach is presented by Lau and Liang [19] . Several construction heuristics (i.e.. insertion heuristic, sweep heuristic, portioned insertion heuristic) are investigated in this paper

Bent and Van Hentenryck [20] propose a two-stage hybrid algorithm for MV-PDPTW. At the first stage, a simple simulated annealing algorithm is applied to minimize the number of vehicles. A Large Neighborhood Search (LNS) is used to minimize the total travel cost of the solution at second stage. An extension of LNS, called adaptive LNS is also presented in [14]. The adaptive LNS uses several removal and insertion heuristics are used during the same search while normal LNS uses only one method for removal and one method for insertions. In each iteration, a removal or insertion heuristic is chosen based on its adaptive weight value. The experiment results on benchmark data [15] show the effectiveness the LNS based approaches since they were able to produce many new best solutions.

Besides Simulated Annealing, Tabu Search and LNS, genetic algorithm has also been applied to solve PDPTW in some studies. Crput et al [21] present an evolutionary algorithm to solve PDPTW. The individual is represented by a list of vehicles routes, where each route consists a sequence of pickup and delivery pairs. The fitness function is to minimize the number of vehicles and travel cost. Two crossover operators are investigated. The first one exchanges fragments of routes between parents, while the other exchanges complete routes. The mutation operators are designed to reduce the number of routes by merging two routes and improve a route by rearranging its nodes.

Pankratz [22] applies a Grouping Genetic Algorithm (GGA) to PDPTW. In this study, individual is represented as a set of genes, each gene represents a group of requests that are assigned to one vehicles. The crossover operator removes vehicles from one parent and inserts into the other parent. Then, a cleaning up procedure is executed to remove duplicate vehicles and the repeated requests. Ding et al [23] also uses GGA with three different routing adjustment strategies.

Recently, there are some studies that consider vehicle routing problem with time window (VRPTW) as a multi objectives optimization problem. Tan et al [24] consider both number of vehicles and total travelling time as 2 objectives in VRPTW. The authors propose a crossover called route-exchange crossover and a multi-mode mutation which contains swapping, splitting and merging of routes. Dominance ranking is used to assign fitness for individuals. After every 50 generations, one out of three local search heuristics is applied to improve quality of individuals. Ombuki et al [25] consider the same problem of VRPTW. The authors proposes best cost route crossover (BCRC) which is an improvement of uniform order crossover (UOC) and constrained route reversal mutation which is an adaption of well-known inversion mutation. Najera and Bullinaria [26] propose a method to measure route similarity and incorporate it into evolutionary algorithm to solve bi-objective VRPTW. The similarity measure, which is based on Jaccard's similarity coefficient is applied to select parents for the recombination process.

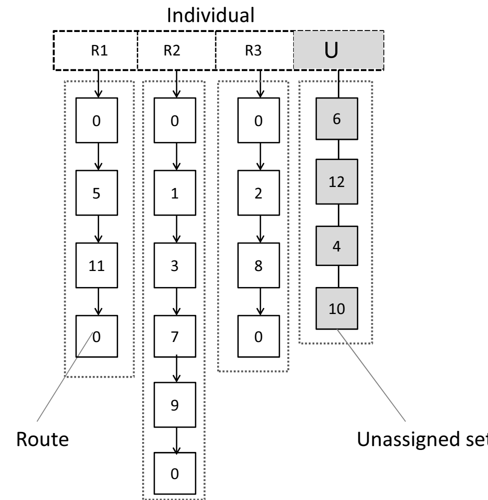


Fig. 1: An Example Individual

IV. The Proposed Evolutionary Multiobjective Optimization Algorithm for PDP-TW-D

A. Individual Representation

In this paper's EMOA, each individual is a variable-length representation of routes . It encodes the number of routes and the order of nodes visited by each vehicle. Each individual also has a set of unassigned nodes. Figure 1 shows an example individual. There are 6 requests in this example. The pickup-delivery (PD) pairs are: (1,7), (2,8), (3,9), (4,10), (5,11) and (6,12). The individual forms a routing plan which has three routes, starting and ending at the depot 0. There are two pairs (4,10), (6,12) are not served in this routing plan.

B. Algorithmic Structure

Algorithm 1 shows the algorithmic structure of evolutionary optimization in the proposed EMOA. It follows the optimization process in NSGA-II, a well-known existing EMOA [27].

At the *0*-th generation, N individuals are randomly generated as the initial population $\mathcal{P}_0$ (Line 2). Each of them has a random number of routes. Each routes in the individual contains a randomly-selected pickup-delivery pairs in a random order. A correction procedure is executed to swap the potions of pickup node and delivery node if the delivery nodes is visited before its corresponding pickup node is.

At each generation (g), two parent individuals (p_1 and p_2) are selected from the current population $\mathcal{P}_g$ with binary tournaments (Lines 6 and 7). A binary tournament randomly takes two individuals from $\mathcal{P}_g$, compares them based on α-dominance relationship, and chooses a superior one as a parent.

With the crossover rate P_c, two parents reproduce two offspring with a crossover operator (Lines 8 to 10). Each offspring performs mutation with the mutation rate P_m (Lines 11 to 16). After that, local search is applied for both offspring to improve their quality. The binary tournament, crossover, mutation and local search operators are executed repeatedly on

Algorithm 1 Optimization Process in the Proposed EMOA

1: g = 0;
2: $\mathcal{P}_g$ = Randomly generated N individuals;
3: **while** g < MAX-GENERATION **do**
4: $\quad \mathcal{O}_g = \emptyset$;
5: $\quad$ **while** $|\mathcal{O}_g| < N$ **do**
6: $\quad\quad p_1$ = tournament($\mathcal{P}_g$)
7: $\quad\quad p_2$ = tournament($\mathcal{P}_g$)
8: $\quad\quad$ **if** random() $\leq P_c$ **then**
9: $\quad\quad\quad \{o_1, o_2\}$ = crossover(p_1, p_2)
10: $\quad\quad$ **end if**
11: $\quad\quad$ **if** (random() $\leq P_m$) **then**
12: $\quad\quad\quad o_1$ = mutation(o_1)
13: $\quad\quad$ **end if**
14: $\quad\quad$ **if** random() $\leq P_m$ **then**
15: $\quad\quad\quad o_2$ = mutation(o_2)
16: $\quad\quad$ **end if**
17: $\quad\quad$ doLocalSearch(o_1)
18: $\quad\quad$ doLocalSearch(o_2)
19: $\quad\quad \mathcal{O}_g = \{o_1, o_2\} \cup \mathcal{O}_g$
20: $\quad$ **end while**
21: $\quad \mathcal{R}_g = \mathcal{P}_g \cup \mathcal{O}_g$
22: $\quad \mathcal{F}$ = sortByDominationRanking($\mathcal{R}_g$)
23: $\quad \mathcal{P}_{g+1} = \{\emptyset\}$
24: $\quad i = 1$
25: $\quad$ **while** $|\mathcal{P}_{g+1}| + |\mathcal{F}_i| \leq N$ **do**
26: $\quad\quad \mathcal{P}_{g+1} = \mathcal{P}_{g+1} \cup \mathcal{F}_i$
27: $\quad\quad i = i + 1$
28: $\quad$ **end while**
29: $\quad$ sortByCrowdingDistance($\mathcal{F}_i$)
30: $\quad \mathcal{P}_{g+1} = \mathcal{P}_{g+1} \cup \mathcal{F}_i[1 : (N - |\mathcal{P}_{g+1}|)]$
31: $\quad g = g + 1$
32: **end while**

$\mathcal{P}_g$ to reproduce N offspring. The offspring ($\mathcal{O}_g$) are combined with the parent population $\mathcal{P}_g$ to form $\mathcal{R}_g$ (Line 19).

The environmental selection process follows the reproduction process. N individuals are selected from $2N$ individuals in $\mathcal{R}_g$ as the next generation's population ($\mathcal{P}_{g+1}$). First, the individuals in $\mathcal{R}_g$ are ranked based on their constrained-dominance relationships. Non-dominated individuals are on the first rank. The i-th rank consists of the individuals dominated only by the individuals on the $(i-1)$-th rank. Ranked individuals are stored in $\mathcal{F}$ (Line 20). $\mathcal{F}_i$ contains the i-th rank individuals.

Then, the individuals in $\mathcal{F}$ move to $\mathcal{P}_{g+1}$ on a rank by rank basis, starting with $\mathcal{F}_1$ (Lines 23 to 26). If the number of individuals in $\mathcal{P}_{g+1} \cup \mathcal{F}_i$ is less than N, $\mathcal{F}_i$ moves to $\mathcal{P}_{g+1}$. Otherwise, a subset of $\mathcal{F}_i$ moves to $\mathcal{P}_{g+1}$. The subset is selected based on the crowding distance metric, which measures the distribution (or diversity) of individuals in the objective space [27] (Lines 27 and 28). The metric computes the distance between two closest neighbors of an individual in each objective and sums up the distances associated with all objectives. A higher crowding distance means that an individual in question is more distant from its neighboring individuals in the objective space. In Line 27, the individuals in $\mathcal{F}_i$ are sorted from the one with the highest crowding distance to the one with the lowest crowding distance. The individuals

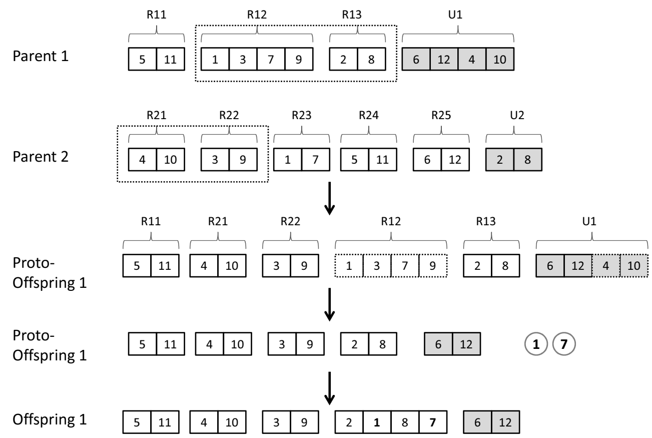


Fig. 2: An Example of Crossover Process

with higher crowding distance measures have higher chances to be selected to $\mathcal{P}_{g+1}$ (Line 28).

In this paper, the constrained dominance concept is defined as same as in [28]. An individual i is said to constrained-dominate an individual j, if any of the following three conditions is true:

- *i*ndividual i is feasible an j is not
- *i*ndividual i and j are both feasible and individual i dominates individual j in objective function
- *B*oth i and j are infeasible, but i dominates j in constraint space

C. Crossover

This paper adopts the group-oriented crossover operator proposed in [22] as its crossover operator. The crossover operator is illustrated in Figure 2.

The operator first selects two crossing points in each of the two parent individual at random. In the example in Figure 2, parent 1's crossing section is {R12,R13} and parent 2's is {R21,R22}. Next, the routes in the crossing section of the second parent are inserted in the parent 1 at the first crossing point. In figure 2, the routes {R22, R23} in parent 2 are inserted into parent 1 right before R12 of parent 1. As a result of this operation, a number PD pairs are duplicated in the proto-offspring (PD pairs: (3, 9), (4, 10)). In order to solve this problem, all parent 1's routes which contain duplicated PD pairs are removed (route R12). If duplicated PD pairs are found in unassigned set of parent 1, they will be deleted from the unassigned set. Note that, at this step all routes imported from parent 2 are left unchanged. Finally, the PD pairs which are belonged to the removed routes of parent 1 but are not contained in the imported routes (PD pair: (1, 7)) are reinserted in random order into existing routes (R11, R21, R22, R13). The reinsertion process can be described as follows: For each PD pair to be reinserted, two random positions at a random existing route are examined. If the insertion does not

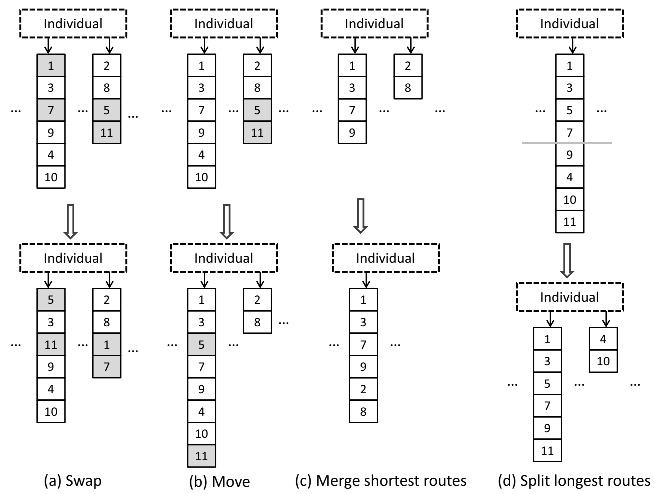


Fig. 3: Mutation operators

make any increment in time window violation, the insertion is accepted. Otherwise, other positions are examined. If all possible insertions make increment in time window violation, a new route is created for the current PD pair. In figure 2, the PD pair (1,7) is inserted into the R13 of parent 1.

D. Mutation

The proposed EMOA uses the following seven mutation operators. (See Fig. 3 for four of them.):

- *Add*: randomly chooses a PD pair from the unassigned set and inserts its pickup-node and delivery node to a randomly-selected positions in a randomly-selected route. This operator ensures that the pickup node is visited before the delivery node.
- *Delete*: removes a randomly-selected PD pair from a randomly-selected route and put it into the unassigned set.
- *Exchange*: randomly chooses a PD pair in a randomly-selected route and replaces it with a pair selected from the unassigned set randomly.
- *Swap*: exchanges the positions of two randomly-selected PD pairs in a route which is also selected randomly.
- *Move*: randomly removes a PD pair from a route and reinsert it into a randomly-selected routes at random positions.
- *Merge the shortest routes*: identifies the two shortest routes and appends one of them to the other.
- *Split the longest route*: identifies the longest route and splits it to two routes at a randomly-chosen point. If there are PD pairs of which pick up node and delivery node are contained in different routes, the delivery node will be moved to the route which contains its corresponding pickup node.

The proposed EMOA classifies these seven mutation operators to two categories. The first category consists of *Add*, *Delete* and *Exchange*. They exhibit the interactions between one route and the unassigned set. The remaining three operators are in the second category. They exhibit the interactions between two routes in the individual. These two categories have the same probably to be used. In each category, mutation operators are selected randomly.

E. Local Search

The local search is designed to improve the routes in the individual by swapping the positions of nodes within their respective route. The swapping has to ensure the precedence constraints of PD pairs. Algorithm 2, 3 show how local search improve an individual.

Algorithm 2 Pseudocode of Local Search

function doLocalSearch(o)
$c = true$
while c=true **do**
  for each route $r \in o$ **do**
    $c = improveRoute(r)$
  end for
end while
end function

Algorithm 3 improveRoute

function improveRoute(r)
N = a set of all nodes in r sorted in random order
$d = f_{distance}(r)$
$v = g_{tw}(r)$
for each $n \in N$ **do**
  for each node m, m is visited after n in r **do**
    Generate a route r' by swapping the positions of n and m
    if r' does not violate precedence constraint **then**
      $d' = f_{distance}(r')$
      $v' = g_{tw}(r')$
      if $d' < d$ and $v \leqslant v'$ **then**
        $r = r'$
        return true
      end if
    end if
  end for
end for
return false
end function

V. Experimental Evaluation

This section evaluate proposed EMOA through experiments with well-known Li and Lim's 100 nodes PDPTW benchmarks [15]. The data set consists of 60 instances and it is classified into two classes (class1 and 2). In class 1, vehicle have smaller capacity and nodes' time window intervals are shorter. Thus, class 1 instances are harder and require more vehicles than class 2.

Table I shows a set of parameter values used in experiments . All experiments were conducted with jMetal [29]. Every experimental result is obtained and shown based on 10 independent runs.

Parameter	Value
Population size	500
Max Generations	600
Crossover rate	0.9
Mutation rate	0.3

TABLE I: Parameter Configurations

A. Diversity Analysis

Tables II and III show the min, max and median values of three objectives at the last generation (600th generation) for class 1 and class 2 respectively. The values in parentheses show the standard deviations over 10 runs for each problem instance. The wide spread of objective values indicate the goodness of the proposed EMOA in diversity.

Figure 4 shows the distribution of objective values for lc101, lc201, lr101, lr201, lrc101, lrc201 by using box plot. The box plots show the distributions of the points graphically. The box in the box plot contains the middle 50% of the data. The upper edge of the box indicates the 75th percentile of the data, and the lower edge indicates the 25th percentile. The middle line in the box indicates the median value of the data. The ends of vertical lines (whiskers) indicate the maximum and minimum data values. If any whisker is more than 1.5 times as long as the length of the box, it is considered as an outlier.

B. Optimality Analysis

In order to illustrate the optimality of proposed EMOA, the best non-dominated solutions in which all requests are served are compared with the best known-solutions of PDPTW benchmark problems. The best-known solutions are available at http://www.sintef.no/projectweb/top/. Table IV, V present the best number of vehicles and total travel distances from proposed EMOA and the best known results for all instances of class 1 and class 2 problems respectively. Bold number represents the best result. The results from two tables indicate that the solutions obtained by the proposed EMOA are competitive with the best-known solution, especially in the class 1 (includes harder problems). In many cases, the proposed EMOA can find the solutions which are as good as the best-known solution. Moreover, it has a better solution in problem instance lrc107 and provides two non-dominated solutions in lc103, lc104 and lc109. In some cases, it has worst results than best-known results. However the differences are small.

C. More options for Decision Maker

The key point of the proposed EMOA is multi-objectives optimization. The interaction among different objective in the algorithm give a set of compromised solutions, i.e. trade-off non-dominated solutions. Therefore, consideration of multiple objectives provide more appropriate options for the planing and decision-making processes.

Figure 5 show non-dominated solutions that a decision maker can achieve when he/she uses the proposed algorithm with different preferences in problem instance lc103. A decision maker might have a fix number of vehicles and he/she want to use all vehicles to fulfill the requests in PDPTW problem. This case is illustrated in figure 5 (a), (b) in which the numbers of vehicles are 9, 6 respectively. It is easy to observe that the result of proposed EMOA is a set of non-dominated solutions with two objective: total demand and total travel distance. Other scenario is, the decision maker has flexible number of vehicles. What he/she considers is: the routing plan has to server at least 95% of total demand (figure 5 (c)). In this scenario, the algorithm can find two non-dominated solutions. The result also indicate that number of vehicles and total travel distance are conflicting in lc103. Lastly, the decision maker might want to restrict the total travel distance between 500 and 600. As shown in figure 5 (d), there are four non-dominated solutions for this scenario in lc103.

VI. Conclusions

This paper formulates a new multiobjective optimization problem, the Pickup and Delivery Problem with Time Windows and Demands (PDP-TW-D), which extends PDP and PDP-TW. With respect to multiple optimization objectives, PDP-TW-D is to find a set of Pareto-optimal routes for a fleet of vehicles in order to serve given transportation requests. This paper proposes and evaluates an evolutionary multiobjective optimization algorithm (EMOA) as a metaheuristic algorithm to solve PDP-TW-D. The proposed EMOA uses a population of individuals, each of which represents a solution candidate, and evolves them through generations to seek the Pareto-optimal solutions with respect to given multiple objectives. In addition to evolution, the proposed EMOA allows individuals to learn and improve themselves in each generation with a local search algorithm. Experimental results demonstrate that the evolutionary and learning processes complement with each other in the proposed EMOA and can effectively obtain quality solutions to PDP-TW-D in a relatively large-scale problem instances that have 100 nodes.

Several future extensions are planned for the proposed EMOA. First, its crossover and mutation operators will be examined further so that the the operators can be better fit to PDP-TW-D. Its local search algorithm will be enhanced too. Second, the proposed EMOA will be evaluated in larger-scale problem instances.

References

[1] S. F. Baker, D. P. Morton, R. E. Rosenthal, and L. M. Williams, "Optimizing military airlift," *Operations Research*, vol. 50, no. 4, pp. 582–602, 2002.

[2] R. S. Solanki and F. Shouthworth, "An execution planning algorithm for military airlift," *Interfaces*, vol. 21, pp. 121–131, 1991.

[3] H. K. Rappoport, L. S. Levy, B. L. Golden, and K. Toussaint, "A planning heuristic for military airlift," *Interfaces*, vol. 22, pp. 73–87, 1992.

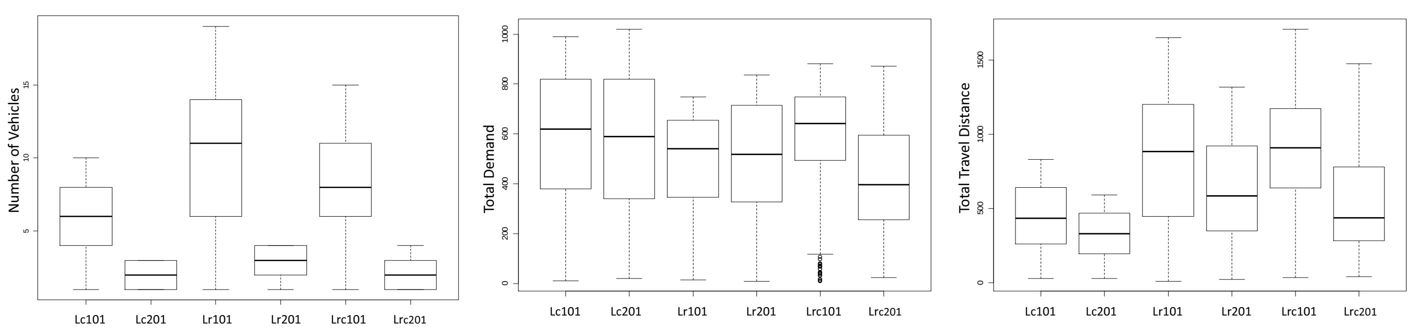
[4] H. K. Rappoport, L. S. Levy, K. Toussaint, and B. L. Golden, "A transportation problem formulation for the mac airlift planning problem," *Annals of Operations Research*, vol. 50, pp. 505–523, 1994.

[5] M. Christiansen, "Decomposition of a combined inventory routing and time constrained ship routing problem," *Transportation Science*, vol. 33, pp. 3–16, 1999.

[6] M. L. Fisher and M. B. Rosenwein, "An interactive optimization system for bulk-cargo ship scheduling," *Naval Research Logistics Quarterly*, vol. 35, pp. 27–42, 1989.

TABLE II: Objective Values of Class1 Problems

Problems	# of Vehicles			Total Demand			Total travel distance		
	Min	Median	Max	Min	Median	Max	Min	Median	Max
lc101	1.9 (1.449)	5.7 (0.823)	10 (0)	107 (119.355)	576 (85.042)	990 (0)	107.4 (113.921)	413.78 (85.023)	828.94 (0)
lc102	1.9 (1.853)	5.6 (0.966)	10 (0)	128 (194.182)	659.5 (84.737)	1070 (0)	109.91 (157.896)	460.44 (88.541)	952.22 (41.496)
lc103	1.5 (1.269)	5.3 (0.675)	10 (0)	68 (105.071)	583 (38.384)	940 (0)	70.81 (108.656)	468.79 (53.722)	1083.36 (21.471)
lc104	1.6 (1.35)	4.65 (1.055)	10 (0.471)	75 (147.742)	555 (73.106)	880 (0)	82.11 (118.399)	453.17 (83.14)	968.23 (39.915)
lc105	2.1 (1.524)	5.6 (1.075)	10 (0)	126 (148.937)	623.5 (102.769)	1030 (0)	113.68 (121.632)	412.63 (108.424)	836.75 (17.189)
lc106	1.1 (0.316)	4.4 (0.699)	10 (0)	23 (13.375)	510.5 (26.082)	970 (0)	31.19 (18.821)	378.19 (25.707)	855.39 (56.199)
lc107	1.5 (0.85)	5.35 (0.747)	10 (0)	84 (109.057)	605 (48.132)	1010 (0)	82.07 (89.041)	432.64 (53.379)	858.03 (38.948)
lc108	1.7 (1.567)	5.1 (1.287)	10 (0)	119 (191.512)	655.5 (92.329)	1100 (0)	105.46 (150.884)	410.14 (108.946)	917.22 (102.083)
lc109	1.9 (1.729)	5.3 (1.337)	10.1 (0.316)	133 (199.836)	607 (103.204)	970 (0)	112.71 (142.662)	461.97 (105.274)	1038.96 (77.962)
lr101	2.2 (1.687)	10.05 (1.423)	19 (0)	83.2 (97.607)	510.65 (48.112)	748 (0)	111.88 (145.846)	828.75 (104.505)	1650.8 (0)
lr102	1.8 (1.229)	7.95 (0.956)	17 (0)	124.9 (130.852)	638.15 (50.573)	892 (0)	119.29 (136.407)	758.8 (95.675)	1527.93 (8.479)
lr103	1.8 (1.476)	6.75 (1.034)	13.1 (0.316)	80.6 (93.599)	523.25 (51.355)	741 (0)	100.92 (131.084)	685.36 (110.478)	1305.03 (11.881)
lr104	1.8 (1.476)	5.4 (0.966)	9.6 (0.516)	92.3 (120.129)	486.4 (54.29)	701 (0)	107.84 (159.919)	528.13 (96.804)	1058.91 (46.166)
lr105	1.6 (1.578)	7.2 (1.135)	14 (0)	70.2 (141.833)	554.1 (53.507)	834 (0)	77.92 (158.727)	644.04 (103.606)	1377.11 (0)
lr106	2.1 (1.449)	6.8 (0.919)	12 (0)	123 (120.167)	572.1 (71.808)	823 (0)	138.7 (148.546)	647.18 (111.11)	1256.65 (8.5)
lr107	1.6 (1.075)	5.8 (1.033)	10 (0)	87.3 (115.588)	493.8 (53.776)	709 (0)	108.81 (129.232)	583.05 (107.032)	1117.03 (12.864)
lr108	1.7 (1.494)	5.4 (0.843)	9 (0)	79.7 (116.171)	472.95 (47.762)	714 (0)	107.91 (144.021)	535.78 (74.316)	971.38 (7.618)
lr109	2.7 (3.164)	7.2 (1.751)	11.1 (0.316)	151.2 (237.675)	575.05 (98.315)	806 (0)	214.34 (351.61)	724.26 (198.127)	1227.05 (36.639)
lr110	1.2 (0.632)	5.9 (0.568)	11.4 (0.699)	29.1 (64.495)	518.85 (38.171)	770 (0)	59.83 (81.914)	590.48 (67.803)	1207.97 (36.139)
lr111	2.3 (1.889)	6.65 (1.001)	10 (0)	116.9 (144.732)	527.35 (48.739)	696 (0)	152.53 (199.112)	638.82 (108.996)	1110.23 (2.77)
lr112	1.5 (0.85)	5.1 (0.876)	10.2 (0.422)	86.8 (108.227)	524.75 (54.699)	794 (0)	90.45 (102.798)	479.58 (88.767)	1088.17 (54.279)
lrc101	2.2 (1.687)	8.2 (1.033)	14.1 (0.316)	129.7 (135.277)	642.65 (58.084)	881 (0)	206.78 (195.893)	916.02 (123.272)	1708.8 (0)
lrc102	2.2 (1.687)	7 (1.054)	12 (0)	144.6 (173.915)	674.65 (74.13)	934 (0)	205.92 (219.412)	844.09 (136.629)	1561.9 (4.563)
lrc103	2 (1.333)	6.3 (0.949)	11.1 (0.316)	130.2 (121.493)	591.21 (62.797)	909 (0)	175.55 (161.836)	697.94 (85.824)	1276.51 (19.938)
lrc104	1.9 (1.912)	5.6 (1.265)	10 (0)	93.3 (176.709)	515.45 (93.776)	837 (0)	135.16 (236.197)	588.36 (142.394)	1146.2 (13.95)
lrc105	1.6 (1.578)	7.1 (1.287)	13 (0)	88.5 (146.088)	657.5 (65.733)	923 (0)	120.58 (214.633)	811.79 (139.311)	1638.42 (1.295)
lrc106	3 (2.449)	7 (1.054)	12 (0)	174.8 (175.182)	612.9 (85.656)	862 (0)	272.3 (279.412)	794.9 (143.026)	1424.73 (0)
lrc107	1.7 (1.16)	6.4 (0.699)	11 (0)	104.2 (150.851)	659.05 (52.185)	939 (0)	118.83 (157.619)	666.93 (85.744)	1235.22 (16.074)
lrc108	2.5 (2.461)	6.7 (1.059)	10.8 (0.422)	150.6 (188.93)	611.25 (80.599)	875 (0)	214.24 (267.391)	701.48 (127.139)	1174.97 (15.851)



(a) Number of vehicles

(b) Total demand

(c) Total travel distance

Fig. 4: Boxplots for three objectives

[7] H. N. Psaraftis, J. B. Orlin, D. Bienstock, and P. M. Thompson, “Analysis and solution algorithms of sealift routing and scheduling problems,” Massachusetts Institute of Technology, Sloan School of Management, Tech. Rep. 1700-85, 1985.

[8] H. Wang, D-H-Lee, and R. Cheu, “Pdptw based taxi dispatch modeling for booking service,” in *Proc. of Int’l Conference on Natural Computation*, 2009.

[9] H. N. Psaraftis, “A dynamic programming solution to the single-vehicle many-to-many dial-a-ride problem with time windows,” *Transportation Science*, vol. 14, pp. 130–154, 1980.

[10] ——, “An exact algorithm for the single-vehicle many-to-many dial-a-ride problem with time windows,” *Transportation Science*, vol. 17, pp. 351–357, 1983.

[11] O. B. G. Madsen, H. F. Ravn, and J. M. Rygaard, “A heuristic algorithm for a dial-a-ride problem with time windows, multiple capacities and multiple objectives,” *Annals of Operations Research*, vol. 60, pp. 193–208, 1995.

[12] P. Toth and D. Vigo, “Heuristic algorithms for the handicapped persons transportation problem,” *Transportation Science*, vol. 31, pp. 60–71, 1997.

[13] M. M. Solomon, A. Chalifour, J. Desrosiers, and J. Boisvert, “An application of vehicle routing methodology to large-scale larvicide control programs,” *Interfaces*, vol. 22, pp. 88–99, 1992.

[14] S. Ropke and D. Pisinger, “An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows,” *Transportation Science*, vol. 40, no. 4, pp. 455–472, 2006.

[15] H. Li and A. Lim, “A metaheuristic for the pickup and delivery problem with time windows,” in *Proc. of IEEE Int’l Conference on Tools with*

TABLE III: Objective Values of Class 2 Problems

Problems	# of Vehicles			Total Demand			Total travel distance		
	Min	Median	Max	Min	Median	Max	Min	Median	Max
lc201	1 (0)	3 (0)	4 (0)	36 (43.022)	568.7 (34.998)	871 (0)	74.32 (65.936)	756.61 (70.759)	1500.72 (65.477)
lc202	1 (0)	2 (0)	3 (0)	38 (19.322)	557 (12.293)	990 (0)	42.52 (31.575)	324.5 (13.909)	601.21 (18.351)
lc203	1.1 (0.316)	2.1 (0.316)	3 (0)	102 (148.159)	512 (71.655)	870 (0)	92.08 (108.137)	347.8 (45.043)	594.38 (10.141)
lc204	1 (0)	2 (0)	3.1 (0.316)	19 (3.162)	560.5 (19.784)	1020 (0)	29.7 (1.744)	315.29 (9.682)	635.15 (51.018)
lc205	1.2 (0.422)	2.2 (0.422)	3 (0)	85 (122.043)	505 (67.412)	880 (0)	90.34 (91.774)	329.92 (54.474)	593.18 (13.598)
lc206	1 (0)	2 (0)	3 (0)	58 (80.939)	498.5 (39.161)	890 (0)	46.95 (46.659)	312.63 (20.821)	590.8 (7.308)
lc207	1.5 (0.85)	2.3 (0.483)	3 (0)	210 (337.738)	573.5 (177.029)	890 (0)	160.69 (219.969)	374.26 (124.552)	592.06 (11.915)
lc208	1.3 (0.675)	2.2 (0.422)	3 (0)	124 (278.855)	522.5 (137.503)	890 (0)	107.22 (175.844)	342.46 (93.59)	621.94 (68.642)
lr201	1.6 (0.966)	3.2 (0.422)	4.2 (0.422)	163.6 (188.085)	590.85 (97.186)	837 (0)	237.65 (287.238)	734.93 (190.836)	1309.7 (66.298)
lr202	1 (0)	2 (0)	3.7 (0.675)	29.4 (9.082)	562.3 (27.629)	791 (0)	33.82 (8.109)	623.51 (53.149)	1314.08 (103.884)
lr203	1 (0)	2 (0)	3.2 (0.422)	44.3 (82.472)	480.7 (57.162)	754 (0)	78.6 (90.839)	513.46 (73.064)	1109.53 (97.304)
lr204	1 (0)	1.3 (0.483)	3 (0)	28 (0)	464.9 (32.372)	721 (0)	36.2 (0)	422.08 (47.329)	1050.19 (49.968)
lr205	1.1 (0.316)	2.1 (0.316)	3.2 (0.422)	74.9 (100.296)	570.95 (49.552)	829 (0)	94.96 (99.749)	598.89 (71.49)	1153.8 (151.331)
lr206	1 (0)	2 (0)	3.3 (0.483)	27.3 (33.15)	471.05 (49.095)	756 (0)	56.02 (43.15)	480.27 (81.44)	1081.08 (123.364)
lr207	1 (0)	1.5 (0.527)	3 (0)	16 (0)	425.1 (25.206)	671 (0)	32.85 (0)	449.63 (58.932)	1042.84 (44.651)
lr208	1 (0)	1.05 (0.158)	2.3 (0.483)	9.3 (2.669)	448 (39.29)	710 (0)	34.05 (5.562)	382.94 (47.37)	873.87 (87.007)
lr209	1 (0)	2 (0)	3.7 (0.483)	16.1 (4.677)	446.5 (34.893)	737 (0)	20.03 (3.208)	462.9 (46.274)	1166.9 (141.483)
lr210	1 (0)	2 (0)	3.1 (0.316)	67.5 (83.729)	484.45 (67.498)	735 (0)	88.89 (112.168)	558.14 (110.332)	1121.46 (104.592)
lr211	1 (0)	2 (0)	3 (0)	25.1 (6.008)	431.8 (14.262)	718 (0)	36.85 (3.694)	422.42 (25.801)	1065.45 (66.869)
lrc201	1 (0)	3 (0)	4 (0)	36 (43.022)	568.7 (34.998)	871 (0)	74.32 (65.936)	756.61 (70.759)	1500.72 (65.477)
lrc202	1.1 (0.316)	2.2 (0.422)	4 (0)	57.6 (95.954)	569.85 (62.224)	827 (0)	95.2 (205.698)	800.58 (115.085)	1468.65 (94.446)
lrc203	1.1 (0.316)	2 (0)	3.6 (0.516)	68.1 (104.559)	514.7 (58.878)	791 (0)	107.47 (158.286)	648.14 (121.169)	1243.89 (130.498)
lrc204	1.1 (0.316)	2 (0)	3.1 (0.316)	62.1 (137.05)	573.4 (60.637)	871 (0)	84.58 (129.55)	476.96 (82.351)	967.49 (72.302)
lrc205	1.1 (0.316)	2.2 (0.422)	4 (0)	85 (132.128)	562.25 (49.203)	858 (0)	125.88 (141.957)	705.81 (114.002)	1492.13 (87.315)
lrc206	1 (0)	2.1 (0.316)	3 (0)	39.1 (78.393)	595.5 (17.005)	908 (0)	86.66 (99.561)	670.48 (37.11)	1167.26 (6.075)
lrc207	1.3 (0.675)	2.2 (0.422)	3 (0)	144.5 (248.316)	550.95 (93.79)	796 (0)	202.55 (325.995)	671.37 (152.767)	1076.71 (15.319)
lrc208	1 (0)	2 (0)	3.5 (0.527)	26.3 (30.258)	574.45 (37.569)	909 (0)	54.57 (39.201)	479.91 (36.127)	1017.62 (99.593)

Artificial Intelligence.

[16] C. D'Souza, S. N. Omkar, and J. Senthilnath, "Pickup and delivery problem using metaheuristic techniques," *Expert Systems with Applications: An International Journal*, vol. 39, no. 1, pp. 328–334, 2012.

[17] J. B. W. P. Nanry, "Solving the Pickup and Delivery Problem with Time Windows Using Reactive Tabu Search," in *Transportation Research Part B: Methodological*, 2000.

[18] M. M. Solomon, "Algorithms for the vehicle routing and scheduling problems with time window constraints," in *Operation Research 35*, 1987.

[19] H. C. Lau and Z. Liang, "Pickup and delivery with time windows: Algorithms and test case generation," in *Proc. of IEEE Int'l Conference on Tools with Artificial Intelligence*, 2001.

[20] P. V. H. R. Bent, " A two-stage hybrid algorithm for pickup and delivery vehicle routing problems with time windows," in *Computers and Operations Research*, 2006.

[21] J. K. J. Crput, A. Koukam and J. Lukasik, " An Evolutionary Approach to Pickup and Delivery Problem with Time Windows ," in *Proceedings of International Conference on Computational Science*, 2004.

[22] G. Pankratz, "A grouping genetic algorithm for the pickup and delivery problem with time windows," in *OR Spectrum*, 2005.

[23] Y. J. G. Ding, L. Li, "Multi-strategy grouping genetic algorithm for the pickup and delivery problem with time windows," in *Proceedings of the first ACM/SIGEVO Summit on Genetic and Evolutionary Computatio*, 2009.

[24] L. H. L. K. C. Tan, Y. H. Chew, " A Hybrid Multiobjective Evolutionary Algorithm for Solving Vehicle Routing Problem with Time Windows," in *Computation optimization and applications*, 2006.

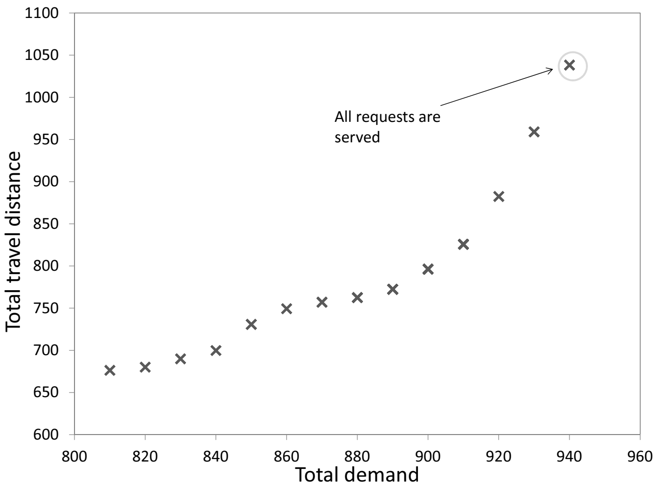
[25] F. H. B. Ombuki, B. J. Ross, " Multi-objective Genetic Algorithms for Vehicle Routing Problem with Time Windows ," in *Applied Intelligence*, 2006.

[26] J. A. B. A. G. Najera, " Bi-objective Optimization for the Vehicle Routing Problem with Time Windows: Using Route Similarity to Enhance Performance," in *Proceedings of 5th International Conference on Evolutionary Multi-Criterion Optimization*, 2009.

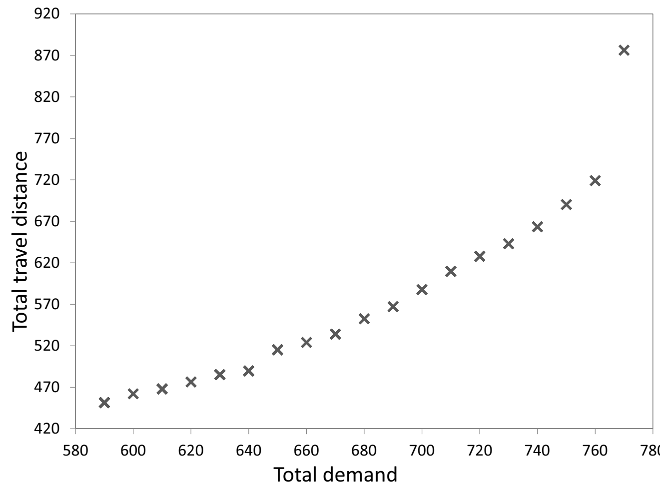
[27] A. P. T. M. K. Deb, S. Agrawal, "A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization:NSGA-II," in *Proc. of Int'l Conference on Parallel Problem Solving from Nature*, 2001.

[28] K. F. A. Oyama, K. Shimoyana, "New constraint-handling method for multi-objective multi-constraint evolutionary optimization and its application to space plane design," in *Evolutionary and Deterministic Methods for Design, Optimization and Control with Applications to Industrial and Societal Problems EUROGEN 2005*, 2005.

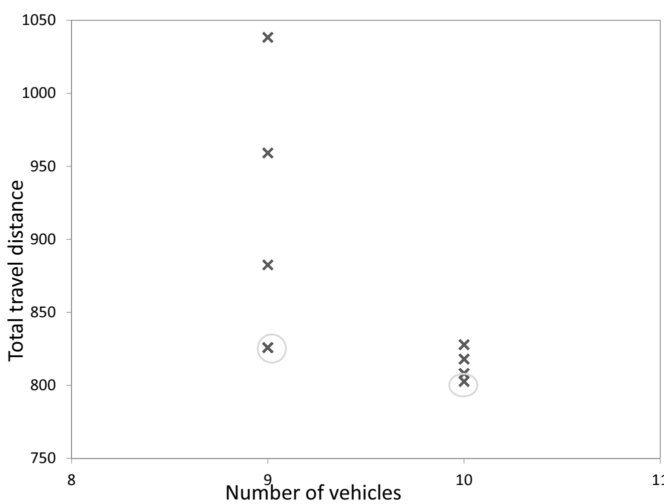
[29] J. Durillo, A. Nebro, and E. Alba, "The jMetal framework for multi-objective optimization: Design and architecture," in *Proc. IEEE Congress on Evolutionary Computation*, 2010.



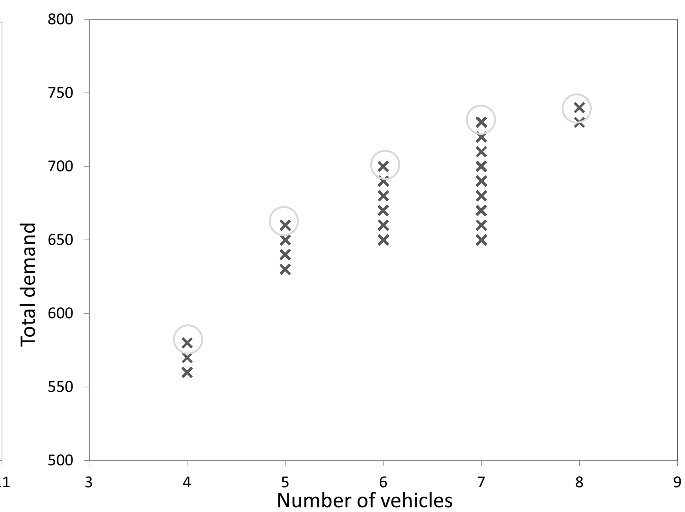
(a) Number of vehicles = 9



(b) Number of vehicles = 6



(c) Total demand is equal or greater than 900



(d) Travel distance between 500 and 600

Fig. 5: Objective spaces of lc103

TABLE IV: Comparison with best known results in class1 problems

Problems	Best known		proposed EMOA	
	# of Veh.	Total dist.	# of Veh.	Total dist.
lc101	**10**	**828.94**	**10**	**828.94**
lc102	**10**	**828.94**	**10**	**828.94**
lc103	**9**	1035.35	**9 10**	1038.35, **827.86**
lc104	**9 860.01**	**9,10**	917.7, **818.6**	
lc105	**10**	**828.94**	**10**	**828.94**
lc106	**10**	**828.94**	**10**	**828.94**
lc107	**10**	**828.94**	**10**	**828.94**
lc108	**10**	**826.44**	**10**	**826.44**
lc109	**9**	1000.6	**9,10**	1068.59, **827.82**
lr101	**19**	**1650.8**	**19**	**1650.8**
lr102	**17**	**1487.57**	**17**	**1489.69**
lr103	**13**	**1292.68**	**13**	**1292.68**
lr104	**9**	**1013.39**	**9**	**1013.99**
lr105	**14**	**1377.11**	**14**	**1377.11**
lr106	**12**	**1252.62**	**12**	**1252.62**
lr107	**10**	**1111.31**	**10**	**1111.31**
lr108	**9**	**968.97**	**9**	**968.97**
lr109	**11**	**1208.96**	**11**	**1208.96**
lr110	**10**	**1159.35**	**10**	**1159.35**
lr111	**10**	**1108.9**	**10**	**1108.9**
lr112	**9**	**1003.77**	10	1035.52
lrc101	**14**	**1708.8**	**14**	**1708.8**
lrc102	**12**	**1558.07**	**12**	**1558.07**
lrc103	**11**	**1258.74**	**11**	**1258.74**
lrc104	**10**	**1128.4**	**10**	**1128.4**
lrc105	**13**	**1637.62**	**13**	**1637.62**
lrc106	**11**	**1424.73**	**11**	**1424.73**
lrc107	**11**	**1230.15**	**11**	**1230.14**
lrc108	**10**	**1147.43**	**10**	**1147.43**

TABLE V: Comparison with best known results in class 2 problems

Problems	Best known		proposed EMOA	
	# of Veh.	Total dist.	# of Veh.	Total dist.
lc201	**3**	**591.56**	**3**	**591.56**
lc202	**3**	**591.56**	**3**	**591.56**
lc203	**3**	**585.56**	**3**	591.17
lc204	**3**	**590.6**	**3**	**590.6**
lc205	**3**	**588.88**	**3**	**588.88**
lc206	**3**	**588.49**	**3**	**588.49**
lc207	**3**	**588.29**	**3**	**588.29**
lc208	**3**	**588.32**	**3**	**588.32**
lr201	**4**	**1253.23**	**4**	**1253.23**
lr202	**3**	**1197.67**	**3**	1206.56
lr203	**3**	**949.4**	**3**	1003.17
lr204	**2**	**849.05**	3	962.87
lr205	**3**	**1054.02**	**3**	1055.9
lr206	**3**	**931.63**	**3**	939.7
lr207	**2**	**903.06**	3	989.98
lr208	**2**	**734.85**	**2**	787.27
lr209	**3**	**930.59**	4	1016.72
lr210	**3**	**964.22**	**3**	1028.68
lr211	**2**	**911.52**	3	958.25
lrc201	**4**	**1406.94**	**4**	**1406.94**
lrc202	**3**	**1374.27**	4	1385.25
lrc203	**3**	**1089.07**	**3**	1092.23
lrc204	**3**	**818.66**	**3**	828.49
lrc205	**4**	**1302.2**	**4**	**1302.2**
lrc206	**3**	**1159.03**	**3**	**1159.03**
lrc207	**3**	**1062.05**	**3**	**1062.05**
lrc208	**3**	**852.76**	**3**	861.31