GRASP with path relinking for the selective pickup and delivery problem

Sin C. Ho[a] and W. Y. Szeto[b*]

[a]Department of Economics and Business Economics
Aarhus University, Denmark
sinho@econ.au.dk

[b]Department of Civil Engineering
The University of Hong Kong, Hong Kong, China
ceszeto@hku.hk
phone: +852 28578552
[*]corresponding author

Abstract

Bike sharing systems are very popular nowadays. One of the characteristics is that bikes are picked up from some surplus bike stations and transported to all deficit bike stations by a repositioning vehicle with limited capacity to satisfy the demand of deficit bike stations. Motivated by this real world bicycle repositioning problem, we study the selective pickup and delivery problem, where demand at every delivery node has to be satisfied by the supply collected from a subset of pickup nodes. The objective is to minimize the total travel cost incurred from visiting the nodes. We present a GRASP with path-relinking for solving the described problem. Experimental results show that this simple heuristic improves the existing results in the literature with an average improvement of 5.72% using small computing times. The proposed heuristic can contribute to the development of effective and efficient algorithms for real world bicycle reposition operations.

Keywords: Pickup and delivery routing; GRASP; Path relinking

1 Introduction

The pickup and delivery problem (PDP) contributes one of the most important classes of the problems because its models have various logistic applications such as reverse logistics, shipping cargoes, dial-a-ride systems, the distribution of beverages and the collection of empty cans and bottles, bike repositioning operations, etc. It aims to determine routes to distribute the commodities between nodes to minimize the total transportation cost. Different variants of the PDP are studied in the literature.

Berbeglia, Cordeau, Gribkovskaia, and Laporte (2007) classified the variants of the PDP according to their structures, the number of deployed vehicles, and the pickup and delivery activities in the nodes. The structure criterion categorizes the PDP into one-to-one (1-1), one-to-many-to-one (1-M-1), and many-to-many (M-M) schemes, based on the number of origins and destinations of the commodities. In the one-to-many-to-one scheme, the commodities from the depot are delivered to delivery nodes and commodities from the pickup nodes are transported to the depot; in the one-to-one scheme, each commodity has exactly one pickup node and one delivery node. In the many-to-many scheme, any node can serve as a origin or as a destination for any commodity. The criterion of the number of deployed vehicles divided the PDP into single and multiple vehicle cases. The activity criterion classifies the PDP according to the way that pickup and delivery operations are performed at nodes. Under this classification, many PDP studies can be categorized explicitly.

Table 1 compares different variants of the PDP according to the schemes laid out by Berbeglia et al. (2007), the node features, and the vehicle characteristics and Table 2 gives the abbreviations of the variants and solution methods. The node features include the selectivity, depot supply/demand, time windows, the pickup and delivery operations, and the perfect balance requirement. The selectivity of nodes is about whether all nodes are required to be visited. Depot supply/demand is concerned with whether the depot supplies or receives commodities. Time windows define the time period during which the vehicle visit the nodes. The pickup and delivery operations are concerned with whether either a pickup or delivery activity is performed at a node and whether both activities are performed at a node separately or simultaneously. The perfect balance requirement is related to whether the total supply equals the total demand. The vehicle characteristics include the number and capacity of vehicles.

Clearly, the classification does not fully list out all elements in the PDP, such as the number of types of commodities or side constraints etc., but it should be enough to distinguish five features of the selective pickup and delivery problem (SPDP) which is also a variant of the PDP under the M-M scheme. First, different from existing 1-M-1 and M-M PDP variants, the SPDP does not require the depot to provide commodities. Second, the SPDP does not have time-window constraints related to the pickup and delivery nodes. Third, based on the problem setting, either pickup or delivery is performed at a customer node. Fourth, the perfect balance requirement does not need to be satisfied. Fifth, some nodes are not visited.

Unlike other PDP studies that consider selectivity, the SPDP requires that only *some*, *but not necessary all*, *pickup* nodes are visited by the vehicle to gather sufficient commodities for all delivery nodes. Moreover, despite the vehicle capacity constraint for each pickup

activity, there is another constraint for the delivery activity to ensure that the vehicle *must* have enough commodities to satisfy the demand of the delivery customers once the vehicle visits the corresponding nodes (i.e., split or incomplete deliveries are not allowed). These are the two distinguished features of this problem (Ting & Liao, 2013). The SPDP is related to applications that supply sufficient commodities to the customers using the cost minimizing principle while visiting all pickup nodes is not a requirement. An illustrative example is the single vehicle bicycle repositioning problem, where bikes are picked up from some surplus bike stations and transported to all deficit bike stations by a repositioning vehicle with limited capacity using the shortest route. Given that the total surplus is larger than the total deficit and all bikes are identical, it is not necessary to visit all the surplus bike stations to satisfy the demand of each deficit station. Hence, some surplus stations are not visited in order to lower the transportation cost.

Although there are realistic applications related to the SPDP, only Ting and Liao (2013) proposed and studied this problem as shown in Table 1. As recognized by them that the SPDP is a NP-hard problem, it is impractical to adopt exact methods to solve instances of realistic sizes. Hence, they adopted a metaheuristic approach and proposed a memetic algorithm for solving the SPDP. The performance was illustrated by comparing the results obtained from the memetic algorithm with the results from the genetic algorithm and tabu search. However, many existing heuristics could solve the variants of the PDP with great success as reflected by the last column of Table 1. These heuristics and their hybrids may give a much better performance.

Table 1 also shows that GRASP has been considered in very few PDP studies despite its success in solving combinatorial optimization problems and industrial applications (Festa & Resende, 2011). GRASP is a multi-start heuristic for producing diverse solutions to combinatorial optimization problems. There are two phases at each iteration of GRASP. In the first phase, a new solution is constructed based on some principles that rely on greediness and randomness. Then, the solution is improved by local search in the second phase. However, Resende and Ribeiro (2010) stated that the searching efficiency of GRASP can be improved via adopting fine tuning mechanisms, multiple neighborhoods, and path relinking.

Path relinking (PR), as an evolutionary method, generates solutions by combining elements from a pair of elite solutions (defined as initial and guiding solutions). The fundamental hypothesis is that good solutions of a problem should share some characteristics, and one could expect to obtain better ones when moving towards the elite solutions in a stepwise sequence. Many studies show that the hybridization of GRASP with path relinking can achieve competitive results to different combinatorial optimization problems such as vehicle routing (Campos, Marti, Sánchez-Oro, & Duarte, 2014; Souffriau, Vansteenwegen, Berghe, & Oudheusden, 2010), arc routing (Reghioui, Prins, & Labadi, 2007; Usberti, França, & França, 2013), capacitated clustering (Deng & Bard, 2011), job shop scheduling (Aiex, Binato, & Resende, 2003), rural road network development (Scaparra & Church, 2005), and network load balancing (D. Santos, de Sousa, & Alvelos, 2013).

In the current literature (2014-2015), we can find several applications of GRASP with path relinking. In Campos et al. (2014), the authors applied the hybrid heuristic on the orienteering problem and achieved competitive results. The authors tested out different strategies on how to construct greedy randomized initial solutions, where these

Table 1: Comparison of the SPDP with PDP variants

PDP variants	References	Structure	Node					Vehicle		Solution methods
			Sel	Dep	TW	Act	PB	Cap	Num	
SVRPPD	Gribkovskaia et al. (2007)	1-M-1		✓		P-D	Y	✓	1	Tabu search
FDPPTW	Wang and Chen (2013)	1-M-1		✓	✓	P-D	Y	✓	> 1	Coevolutionary algorithm
VRPSPD	Zachariadis et al. (2009)	1-M-1		✓		PD	Y	✓	> 1	Tabu search + GLS
	Çatay (2010)									ACO
	Zachariadis and Kiranoudis (2011)									Local search metaheuristic
	Li et al. (2015)									ILS
SVRPDSP	Gribkovskaia et al. (2008)	1-M-1	✓	✓		P-D	N	✓	1	Tabu search
	Bruck et al. (2012)									EA+VNS
VRPDSPTW	Gutiérrez-Jarpa et al. (2010)	1-M-1	✓	✓	✓	P-D	N	✓	> 1	Branch-and-price
TSPPDF	Cordeau, Dell'Amico, and Iori (2010)	1-1				P/D	Y		1	Branch-and-cut
TSPPDL	Cordeau, Iori, et al. (2010)	1-1				P/D	Y		1	Branch-and-cut
TSPPD	Dumitrescu et al. (2010)	1-1				P/D	Y		1	Branch-and-cut
TSPPDF	Erdoğan et al. (2009)	1-1				P/D	Y		1	PTS/ILS
PDPTW	Nanry and Barnes (2000)	1-1			✓	P/D	Y	✓	> 1	Reactive tabu search
	Bent and Van Hentenryck (2006)									SA+LNS
m-PDTSP	Hernández-Pérez and Salazar-González (2009)	1-1				P-D	Y	✓	1	Benders decomposition
	Hernández-Pérez and Salazar-González (2014)	M-M								Branch-and-cut
1-PDTSP	Hernández-Pérez and Salazar-González (2004)	M-M		✓		P/D	Y	✓	1	Greedy algorithm; branch-and-cut
	Hernández-Pérez et al. (2009)									Hybrid GRASP+VND
	Zhao et al. (2009)									Genetic algorithms
	Mladenović et al. (2012)									GVND
SP	Anily et al. (1999)	M-M		✓		P-D	Y	✓	1	Exact $\mathcal{O}(n^2)$ algorithm
	Anily et al. (2011)									1.5-approximation algorithm
MSP	Bordenave et al. (2010)	M-M		✓		P-D	Y	✓	1	Local search heuristics
NCSP	Erdoğan et al. (2010)	M-M		✓		P-D	Y	✓	1	Branch-and-cut
1-TSP-SELPD	Falcon et al. (2010)	M-M	✓	✓		P/D	N	✓	1	ACO
SPDP	Ting and Liao (2013)	M-M	✓			P/D	N	✓	1	Memetic algorithm
	This study									GRASP+PR

Sel: selectivity; Dep: depot supply/demand; TW: time window; Act: pickup and delivery activities; PB: perfect balance;
Cap: capacity; Num: number; P-D: two activities may be performed together or separately at each delivery node;
PD: each delivery node is visited exactly once for the combined pickup and delivery; P/D: either activity is performed at each delivery node but not both

Table 2: Abbreviations

1-PDTSP	One-commodity pickup-and-delivery travelling salesman problem
1-TSP-SELPD	The one-commodity traveling salesman problem with selective pickup and delivery
FDPPTW	Flexible delivery and pickup problem with time windows
m-PDTSP	Multi-commodity pickup-and-delivery traveling salesman problem
MSP	Mixed swapping problem
NCSP	Non-preemptive capacitated swapping problem
PDPTW	Pickup and delivery (vehicle routing problem) with time windows
SP	Swapping problem
SPDP	Selective pickup and delivery problem
SVRPDSP	Single vehicle routing problem with deliveries and selective pickups
SVRPPD	Single vehicle routing problem with pickups and deliveries
TSPPD	Traveling salesman problem with pickup and delivery
TSPPDF	Pickup and delivery traveling salesman problem with FIFO loading
TSPPDL	Pickup and delivery traveling salesman problem with LIFO loading
VRPDSPTW	Vehicle routing problem with deliveries, selective pickups and time windows
VRPSPD	Vehicle routing problem with simultaneous pickups and deliveries
ACO	Ant colony optimization
EA	Evolutionary algorithm
GLS	Guided local search
GRASP	Greedy randomized adaptive search procedure
GVNS	General variable neighborhood search
ILS	Iterated local search
LNS	Large neighborhood search
PR	Path relinking
PTS	Probabilistic tabu search
SA	Simulated annealing
VND	Variable neighborhood descent
VNS	Variable neighborhood search

solutions are improved by local search consisting of four different neighborhood operators. Path relinking was applied to the solutions obtained from GRASP. Morán-Mirabal, González-Velarde, and Resende (2014) proposed a GRASP with path relinking for the family traveling salesman problem. Path relinking is integrated within GRASP a little bit different than Campos et al. (2014). First, after the local search procedure path relinking is only applied on a pair of solutions consisting of a local optimal solution and one of the elite solutions. Second, a more evolved path relinking is applied to every pair of elite solutions. Due to the computational burden, the second path relinking is only applied once a while. D. O. Santos and Xavier (2015) presented a GRASP with path relinking for a dynamic dial-a-ride problem. The path relinking utilizes three different operators to generate the solutions. Computational experiments show that their heuristic performs better than GRASP. A non-routing application is found in Ríos-Mercado and Escalante (2015). The authors applied the hybrid heuristic on a commercial districting problem. They studied the effects of integrating path relinking within GRASP versus placing it after GRASP as a post-optimization phase. Their results indicate that the latter option achieved the best results.

Besides hybridizing path relinking with GRASP, one can also find path relinking hybridized with tabu search (Jia & Hu, 2014; Lai & Hao, 2015; Peng, Lü, & Cheng,

2015; Urrutia, Milanés, & Løkketangen, 2015), local search (Yang, Zhang, & Zhu, 2015), population-based metaheuristics (F. B. de Oliveira, Enayatifar, Sadaei, Guimarães, & Potvin, 2015; Hamdi-Dhaoui, Labadie, & Yalaoui, 2014; Marinakis & Marinaki, 2015; Martí, Corberán, & Peiró, 2015; Ribas, Companys, & Tort-Martorell, 2015) and mathematical programming based approaches (R. M. de Oliveira, Lorena, Chaves, & Mauri, 2014; Li, Chu, Prins, & Zhu, 2014). One important component with path relinking is the neighborhood operator for moving from the initial solution to the guiding solution. One common observation with the 16 recent publications is that the authors use simple problem-dependent operators to create the intermediate solutions (i.e., the path) that take the attributes of the guiding solution into consideration. For example, for the orienteering problem in Campos et al. (2014), an intermediate solution is formed by inserting nodes (that exist in the guiding solution) into the initial solution and by removing nodes (that do not exist in the guiding solution) from the solution. Another example is the commercial districting problem (Ríos-Mercado & Escalante, 2015), where new intermediate solutions are created by relocating a node from its territory to another territory without making the territories disconnected. Traditionally, the path ends when an intermediate solution coincides with the guiding solution. However, it is also possible to terminate the path before reaching the guiding solution (Jia & Hu, 2014; Morán-Mirabal et al., 2014; D. O. Santos & Xavier, 2015; Urrutia et al., 2015). Readers are referred to Ribeiro and Resende (2012) for more ideas/strategies on how to design a path relinking procedure.

This paper proposes a hybrid heuristic based on GRASP and path relinking for solving the selective pickup and delivery problem. One of the strengths of the proposed method is its simplicity: the heuristic relies on simple principles to construct a solution, to improve a solution, to update the solution pool and to generate paths based on initial and guiding solutions. In the proposed path relinking, a different strategy is utilized. Instead of terminating the path at the guiding solution (which is usually the chosen strategy in the current literature), the path might be truncated before reaching the guiding solution (i.e., truncated path relinking), or the path might be extended beyond the guiding solution (i.e., exterior path relinking). The algorithm does not decide a priori which form (truncated or exterior) to apply. It depends on the pair of initial and guiding solutions, the path relinking operator used to create the path and the termination condition. Overall, it is easy to implement the heuristic and the heuristic only requires three parameters. Despite its simplicity, the proposed method is also effective and efficient. The heuristic is assessed on a set of 90 benchmark instances used in the literature. This heuristic has improved the results obtained by a memetic algorithm on 88 of the instances, with an average improvement of 5.72% using on average less than 30 seconds of computing time. The contributions of our paper are as follows. 1) We propose a different path relinking strategy in our hybrid algorithm to solve the selective pickup and delivery problem and 2) the proposed algorithm is simple as well as effective and efficient compared with the memetic algorithm used in the literature. It is expected that our proposed algorithm can contribute to the development of efficient and effective algorithms for real world bike repositioning operations.

The paper is organized as follows. Section 2 introduces a complete mathematical formulation of the SPDP. Section 3 depicts the proposed heuristic. Section 4 illustrates the computational results and a comparison with the results in the literature. Section 5 presents the conclusions.

2 Mathematical formulation

2.1 Problem setting

A complete graph formed by nodes and edges is considered. The nodes can be classified into three categories: pickup nodes, delivery nodes, and the depot. Each pickup node supplies the same type of commodity but the supply varies from one pickup node to another. A demand is associated with every delivery node, and the demand may vary across the delivery nodes. The depot neither requires nor supplies any commodity, but the vehicle may return to the depot non-empty. A single vehicle with limited capacity is used to transport the vehicle load from some pickup nodes to all delivery nodes to satisfy the needs of each delivery node. Each pickup node is visited at most once. For every visit to a pickup node, all supply is required to be loaded on the vehicle and the resultant vehicle load must not exceed the vehicle capacity. Each delivery node is visited exactly once. For every visit to a delivery node, the vehicle load must be enough to satisfy the demand. There is a cost associated with each travel between nodes. The objective is to determine the minimum cost route starting from and ending at the depot to satisfy the above requirements.

2.2 Problem formulation

The notations of this study are given below:

$\mathcal{N}$	Set of nodes, indexed by $i = 0, 1, \ldots, \lvert\mathcal{N}\rvert$ with 0 representing the depot
$\mathcal{E}$	Set of edges
$\mathcal{P}$	Set of pickup nodes
$\mathcal{D}$	Set of delivery nodes
c_{ij}	Cost of edge $(i, j) \in \mathcal{E}$, $c_{ij} > 0$
d_i	Demand at node i. $d_i > 0$ if $i \in \mathcal{P}$. $d_i < 0$ if $i \in \mathcal{D}$. $d_0 = 0$
Q	Vehicle capacity
M	A very big constant

Decision variables:

$$y_{ij} = \begin{cases} 1, & \text{if the vehicle travels directly from node } i \text{ to node } j; \\ 0, & \text{otherwise.} \end{cases}$$

l_{ij}	The load on the vehicle when it travels directly from node i to node j. l_{ij} is zero if $y_{ij} = 0$.
g_i	Auxiliary variable associated with node i for the sub-tour elimination constraints.

Based on the above notations, the SPDP can be formulated mathematically as follows:

$$\min \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}} c_{ij} y_{ij} \tag{1}$$

$$\text{s. t.} \sum_{i \in \mathcal{N}} y_{ij} = \sum_{i \in \mathcal{N}} y_{ji} \leq 1 \qquad \forall j \in \mathcal{P} \tag{2}$$

$$\sum_{i \in \mathcal{N}} y_{ij} = \sum_{i \in \mathcal{N}} y_{ji} = 1 \qquad \forall j \in \mathcal{D} \cup \{0\} \tag{3}$$

$$l_{ij} \leq Q y_{ij} \qquad \forall i, j \in \mathcal{N} \qquad (4)$$

$$\sum_{i \in \mathcal{N}} l_{ji} - \sum_{i \in \mathcal{N}} l_{ij} = d_j \qquad \forall j \in \mathcal{N} \setminus \{0\} \qquad (5)$$

$$g_i + 1 - M(1 - y_{ij}) \leq g_j \qquad \forall i, j \in \mathcal{N} \qquad (6)$$

$$y_{ij} \in \{0, 1\} \qquad \forall i, j \in \mathcal{N} \qquad (7)$$

$$g_i \geq 0 \qquad \forall i \in \mathcal{N} \qquad (8)$$

$$l_{ij} \geq 0 \qquad \forall i, j \in \mathcal{N} \qquad (9)$$

Equation (1) states the objective of the SPDP, which is to minimize the total travel cost of the vehicle. Constraints (2) ensure that each pickup node is visited at most once. Constraints (3) require that each delivery node and the depot must be visited exactly once. Constraints (4) ensure that the vehicle load is not greater than the vehicle capacity. Constraints (5) depict the relation between the vehicle load before and after visiting a node, and their difference must equal to the demand or supply at the visited node. Constraints (6) eliminate the sub-tours among nodes. Constraints (7)-(9) define the decision variables to be binary or nonnegative. Note that l_{ij} takes an integer value at optimality due to the integrality of d_i and constraints (5).

Compared with the formulation from Ting and Liao (2013), our formulation represents the subtour elimination constraints differently and has more decision variables than theirs. More importantly, our formulation has explicitly related the vehicle load of each arc (i, j) and the corresponding indicator variable y_{ij} used to define whether arc (i, j) is on a tour, and hence our formulation can be directly used to solve for exact solutions by CPLEX for comparison with the results from heuristics if needed.

3 The hybrid algorithm

In this paper, a hybrid algorithm based on GRASP (Feo & Resende, 1995) and Path relinking (Glover & Laguna, 1993) is proposed to solve the selective pickup and delivery problem. At every iteration of GRASP, an initial solution is constructed and the initial solution is improved by a local search procedure. The best solution obtained from the different iterations is saved and returned to the user. The disadvantage with GRASP is that the iterations are independent from each other. Hence, as a remedy to overcome this independence, many researchers have included path relinking within the GRASP framework (e.g., Laguna & Martí, 1999). Path relinking generates new solutions by exploring paths that are connected by elite solutions found by GRASP. Path relinking has been applied successfully to different vehicle routing problems (e.g., Campos et al., 2014; Ho & Gendreau, 2006; Souffriau et al., 2010). Hence, path relinking embedded within GRASP is chosen to tackle the selective pickup and delivery problem as well.

A tour x is defined as $(i_0, i_1, \ldots, i_n, i_{n+1})$ where $i_0 = i_{n+1} = 0$ (i.e., the depot) and $i_h \in \mathcal{N}$, $h = 1, 2, \ldots, n$ where n represents the total number of pickup and delivery nodes in the tour. For a solution x, let $f(x)$ be its travel cost. The travel cost is obtained by adding up the costs $c_{i_u i_v}$ associated with traversing the edges (i_u, i_v). A tour to the selective pickup and delivery problem is incomplete (or infeasible) if at least one of the following

is met: 1) not every delivery node is visited; 2) no pickup node is visited; 3) the loading constraints of the vehicle are violated.

In Algorithm 1, a tour is constructed in `GreedyRandomized` (details are found in Algorithm 2). This subroutine ensures that each delivery node is added to the tour and appears in the tour exactly once. (i.e., this subroutine ensures that the first infeasibility condition mentioned in the previous paragraph cannot be met). However, the tour is still incomplete. The incomplete tour is improved by `LocalSearch`. New incomplete tours are generated by applying `PathRelinking` on incomplete tours from the reference set $\mathcal{R}$ and a local optimum (details can be found in Algorithm 3). However, no nodes are added or deleted from the tours. `UpdateR` determines whether an incomplete tour can be included in $\mathcal{R}$. Pickup nodes are added to the incomplete tour in `AddPickUpNodes` until the tour constructed satisfies the loading constraints (i.e., the violation of the third infeasibility condition does not occur). Note that to address the third infeasibility condition, the procedure `AddPickUpNodes` must add at least one pickup node to the incomplete tour consisting of delivery nodes only. This means that the procedure implicitly caters the second infeasibility condition as well.

Algorithm 1 `GRASP+PR`

Set $f^* = \infty$.
for $b = 1, \ldots, \kappa$ **do**
  $x =$`GreedyRandomized`()
  $\hat{x} =$`LocalSearch`(x)
  $\bar{x} =$`PathRelinking`$(\hat{x}, \mathcal{R})$
  $\mathcal{R} =$`UpdateR`$(\bar{x}, \mathcal{R})$
  $\check{x} =$`AddPickUpNodes`$(\bar{x})$
  if $f(\check{x}) < f^*$ **then**
    Set $f^* = f(\check{x})$ and $x^* = \check{x}$.
  end if
end for
return x^*.

In the selective pickup and delivery routing problem, every delivery node must be visited, but this requirement does not apply to the pickup nodes. This leads to the strategy where delivery nodes are inserted first, and thereafter pickup nodes are inserted. This is due to the fact that pickup nodes are typically fewer than the delivery nodes, and that inserting pickup nodes later can reduce the complexity of the search algorithm. Hence, in each iteration, the procedures `GreedyRandomized`, `LocalSearch`, `PathRelinking` and `UpdateR` are applied on tours with delivery nodes only. Pickup nodes are added to the incomplete tours in `AddPickUpNodes`. In other words, our proposed algorithm allows infeasible solutions (i.e., incomplete tours) to be generated in the procedure `GreedyRandomized` and maintained in procedures `LocalSearch`, `PathRelinking` and `UpdateR`, and the next procedure `AddPickUpNodes` is to recover solution feasibility. The best solution found is saved in each iteration and returned as the final result. The heuristic runs for a fixed number of iterations κ.

3.1 Construction heuristic

Initial tours are constructed using the value-based strategy from GRASP (Festa & Resende, 2011). This strategy allows randomness to be controlled through the parameter α where $\alpha \in (0, 1]$. The tour is constructed by inserting one delivery node at a time. The delivery node is chosen randomly from a set of unvisited delivery nodes whose incremental costs ($c(r) = c_{i_n r} + c_{r i_{n+1}} - c_{i_n i_{n+1}}$) are not larger than the best greedy selection by $\alpha \times 100\%$ of the difference between the maximum and minimum incremental costs.

Algorithm 2 `GreedyRandomized`

Set $x = (i_0, i_{n+1})$.
Set $\bar{\mathcal{D}} = \mathcal{D}$.
while $|\bar{\mathcal{D}}| > 0$ **do**
  Set $c_{min} = \min_{r \in \bar{\mathcal{D}}} \{c_{i_n r} + c_{r i_{n+1}} - c_{i_n i_{n+1}}\}$
  Set $c_{max} = \max_{r \in \bar{\mathcal{D}}} \{c_{i_n r} + c_{r i_{n+1}} - c_{i_n i_{n+1}}\}$
  Set $RCL = \{r \in \bar{\mathcal{D}} : c(r) \leq c_{min} + \alpha(c_{max} - c_{min})\}$.
  Randomly select $\hat{i}$ from RCL.
  Insert $\hat{i}$ between i_n and i_{n+1} in x.
  Set $\bar{\mathcal{D}} = \bar{\mathcal{D}} \setminus \{\hat{i}\}$.
end while
return x.

Algorithm 2 is a sequential construction heuristic where a tour is constructed by inserting one delivery node at a time at the end of the tour (i.e., between i_n and i_{n+1}) until all delivery nodes are added to the tour. This algorithm has a time complexity of $\mathcal{O}(|\mathcal{D}|^2)$. The output from this procedure is a tour starting and ending at the depot, and visiting each delivery node once.

3.2 Local search

The tour x is then improved by 2-opt using the best improving strategy. 2-opt is a classical neighborhood operator originally designed for the traveling salesman problem (Lin, 1965). A neighbor solution is obtained by removing non-adjacent arcs (i_u, i_{u+1}) and (i_v, i_{v+1}), and adding the arcs (i_u, i_v) and (i_{u+1}, i_{v+1}) to complete the tour. 2-opt is applied until no improvement can be found. This algorithm has a time complexity of $\mathcal{O}(|\mathcal{D}|^2)$.

3.3 Path relinking

The purpose with path relinking is to explore the paths connected by pairs of elite solutions. One of the solutions in the pair is labelled as an *initial solution*, while the other is labelled as a *guiding solution*. A path between the initial solution and the guiding solution is created by generating new solutions using a neighborhood operator to gradually introduce the attributes of the guiding solution into the newly generated solutions. As the procedure progresses, the new solutions possess more and more attributes of the guiding solution, and less attributes of the initial solution.

Path relinking (see Algorithm 3) with $\mathcal{O}(|\mathcal{D}|^2)$ is applied to $|\mathcal{R}|$ pairs of solutions consisting of the local minimum solution $\hat{x}$ and the elite solutions from the reference set $\mathcal{R}$. Every pair of solutions is subject to two rounds of path relinking. In the first round, $\hat{x}$ takes on the role as the initial solution x_I and the elite solution takes on the role as the guiding solution x_G. The roles are switched in the second round. Each round of the path relinking process is terminated after five consecutive iterations without introducing new attributes from the guiding solution into the newly generated solution $\bar{x}$. This means that the path from x_I to x_G may be truncated before reaching x_G, or this may also mean that the path is extended to beyond x_G.

The neighborhood operator 2-opt is used to generate new solutions. In order to make sure that the newly generated solutions possess attributes from the guiding solution, only *valid* solutions are accepted. In this case, an attribute is an arc. Arcs that exist in both x_I and x_G are locked, which means that valid solutions cannot be obtained by removing locked arcs.

Let T be the set of common arcs between the initial solution x_I and the guiding solution x_G, and let A_I and A_G be the sets of arcs that make up the initial and guiding solutions, respectively. The neighborhood operator used to create the path from x_I is 2-opt. The difference between this 2-opt and the 2-opt used in `LocalSearch` is that tours obtained by removing arcs $(i_u, i_{u+1}) \in T$ and $(i_v, i_{v+1}) \in T$ are not considered.

The set of valid neighbor solutions is defined as the neighborhood $\mathcal{H}$. The tour $\bar{x}$ that yields the lowest total distance is chosen from $\mathcal{H}$. The reason for not choosing the tour with the highest number of attributes from x_G is that most likely the algorithm will end up with a short path. In order to explore the vicinity of the path, a long path is preferred. Any arcs in $\bar{x}$ common with the guiding solution are recorded in T. Due to computational burden and because long paths are preferred, each round of path relinking is terminated after five consecutive iterations without encountering solutions with an increasing similarity with x_G. The best tour encountered during the path relinking procedure is improved by local search.

3.4 Updating the reference set $\mathcal{R}$

A tour $\bar{x}$ is added to the reference set $\mathcal{R}$ if the reference set is not full (i.e., $|\mathcal{R}| < R_{max}$, where R_{max} is the maximum number of solutions that can be stored in $\mathcal{R}$). If the reference set is already full and tour $\bar{x}$ is better than the worst tour in the reference set, then the worst tour is replaced by tour $\bar{x}$. This procedure only takes $\mathcal{O}(1)$ time.

3.5 Adding pickup nodes

The tour $x = (i_0, i_1, \ldots, i_{n+1})$ so far consists of all delivery nodes only. In order to construct a feasible solution, pickup nodes need to be inserted in x and vehicle capacity constraints are considered.. Pickup nodes are inserted one at a time and the selection of pickup node $\hat{r}$ and the insertion position $\hat{p}$ are based on the cheapest insertion criterion, and the inclusion must not violate the vehicle capacity constraint. Pickup nodes are added

Algorithm 3 `PathRelinking`

Require: A local minimum $\hat{x}$ and the reference set $\mathcal{R}$

1: **if** $\mathcal{R} \neq \emptyset$ **then**
2: Set $f_1 = \infty$.
3: **for** $b = 1, \ldots, |\mathcal{R}|$ **do**
4: Set $x_I = \hat{x}$ and $x_G = \sigma_b$ (where σ_b is the bth element in $\mathcal{R}$).
5: Set $T = A_I \cap A_G$ and $x = x_I$.
6: Set $noImpr = 0$.
7: **while** $noImpr < 5$ **do**
8: Let $\mathcal{W}(x)$ be the neighborhood of x obtained by applying 2-opt.
9: Let $\mathcal{H} = \{z \in \mathcal{W}(x) : z$ is not obtained by removing common arcs from $x\}$.
10: Select a tour $\bar{x} \in \mathcal{H}$ that minimizes $f(\bar{x})$. $\bar{x}$ is obtained by removing (i_{u^*}, i_{u^*+1}) and (i_{v^*}, i_{v^*+1}), and adding (i_{u^*}, i_{v^*}) and (i_{u^*+1}, i_{v^*+1}).
11: **if** $(i_{u^*}, i_{v^*}) \in A_G$ and/or $(i_{u^*+1}, i_{v^*+1}) \in A_G$ **then**
12: Set $T = T \cup (i_{u^*}, i_{v^*})$ and/or $T = T \cup (i_{u^*+1}, i_{v^*+1})$.
13: Set $noImpr = 0$.
14: **else**
15: Set $noImpr = noImpr + 1$.
16: **end if**
17: **if** $\bar{x} \neq x_I$ and $\bar{x} \neq x_G$ and $f(\bar{x}) < f_1$ **then**
18: Set $f_1 = f(\bar{x})$ and $\ddot{x} = \bar{x}$.
19: **end if**
20: Set $x = \bar{x}$.
21: **end while**
22: Repeat lines 5-21 (once for every b), but with $x_I = \sigma_b$ and $x_G = \hat{x}$.
23: **end for**
24: $\bar{x} =$`LocalSearch`$(\ddot{x})$
25: **return** $\bar{x}$.
26: **else**
27: **return** $\hat{x}$.
28: **end if**

until the solution becomes feasible. The full procedure is depicted in Algorithm 4. It has the time complexity of $\mathcal{O}(|\mathcal{P}|^2)$.

Algorithm 4 `AddPickUpNodes`

Require: An infeasible tour x.

Set $\hat{\mathcal{P}} = \mathcal{P}$.
Set $l_{\min} = \min_{u=1,\ldots,n+1}\{l_{i_{u-1}i_u}\}$.
while $l_{\min} < 0$ **do**
  $(\hat{r}, \hat{p}) = \arg\min_{r\in\hat{\mathcal{P}};p=1,\ldots,t}\{c_{i_{p-1}r} + c_{ri_p} - c_{i_{p-1}i_p} : l_{i_{u-1}i_u} + d_r \leq Q; u = p, \ldots, t-1\}$, where t is the index of the first node i_t in the tour where $l_{i_t i_{t+1}} < 0$, $t = 1, \ldots, n$.
  Insert $\hat{r}$ in position $\hat{p}$ in x.
  Set $\hat{\mathcal{P}} = \hat{\mathcal{P}} \setminus \{\hat{r}\}$.
  Update $l_{i_{u-1}i_u}$, $u = \hat{p}+1, \ldots, n+1$.
  Set $l_{\min} = \min_{u=\hat{p}+1,\ldots,n+1}\{l_{i_{u-1}i_u}\}$.
end while
return x.

4 Computational experiments

The heuristic was coded in C++ and all computational experiments were carried out on a Dell notebook with an Intel Core i5-2520M CPU@2.5GHz. The efficiency of the hybrid algorithm was validated through a series of computational experiments. The experiments were conducted on the same set of test instances described in Ting and Liao (2013) and these instances are available from `https://db.tt/6ErosIg9`. The number of nodes in these instances range from 91 to 454. The vehicle capacity ranges from 400 to 1000. A total of 90 instances. These instances are modified from some pickup and delivery instances. An instance is denoted as X(Y)/Z, where X denotes the name of the original instance, Y denotes the number of nodes ($\mathcal{N}$), and Z denotes the number of pickup nodes ($\mathcal{P}$). γ is the additional supply added to the original supply of each pickup node (i.e., $d_i = d_i + \gamma$, $\forall i \in \mathcal{P}$). Ting and Liao (2013) compared the results obtained from a tabu search (TS), a genetic algorithm (GA), and a memetic algorithm (MA). Their results showed that MA outperformed both TS and GA. In the following, parameter tuning results will be given. Then, the results obtained from the hybrid heuristic (GRASP+PR) as well as the results from GRASP will be compared to the results from MA.

4.1 Parameter tuning

The heuristic has three parameters: α, R_{max} and κ. The tuning process of the first two parameters is performed on a selection of instances - n100mosA, n200mosA, n300mosA and n400mosA on different values of γ and Q, a total of 36 instances. The first parameter to be tuned was α and it governs the degree of randomness when creating initial solutions. In the tuning process, α can take on different values in the interval $[0.5, 0.9]$, while $R_{max} = 0$ and $\kappa = 100$. Each instance was run 30 times. On average, setting $\alpha = 0.5$ yields the best results. Hence, α is given the value 0.5. The next parameter in the tuning process is R_{max}. It is the maximal number of tours the reference set can hold for the purpose of the

path relinking procedure. The larger R_{max} is, the better the results will be, but at the expense of increasing computing time. While $\alpha = 0.5$ and $\kappa = 100$, R_{max} is set to take the following values: 5, 10, 15, 20, 25 and 30. The average results are shown in Figure 1. It is decided to set R_{max} to 10, as it shows a good trade-off between solution quality and computing time. The last parameter, κ, sets the total number of iterations for the entire algorithm. As with R_{max}, the higher value κ gets, the better the average results will be and this is also at the expense of soaring computing time. Hence, in the next few sections results from running GRASP and GRASP+PR with $\kappa \in \{50, 100, 200\}$ are shown.

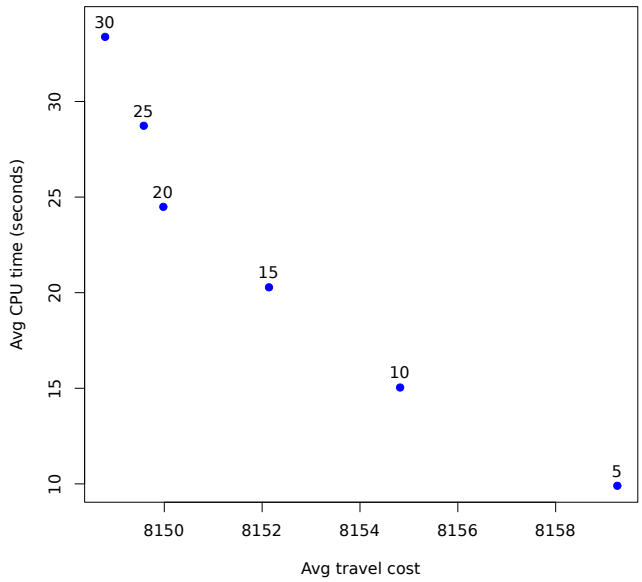

Figure 1: Results from tuning R_{max}

4.2 Comparing GRASP with GRASP+PR

To show the benefit of including a path relinking procedure within the GRASP framework, a comparison between the results obtained from GRASP and GRASP+PR is detailed below. The comparison is between GRASP (omitting lines 5-6 of Algorithm 1) and two variants of GRASP+PR. The first variant is the standard version as stated in Algorithm 1, while the second variant is a slightly faster version where `AddPickUpNodes` is only applied to the solutions in the reference set after the κ iterations have been completed (i.e., lines 7-10 of Algorithm 1 are moved out of the for-loop). The main results are reported in Tables 3 and 4. In these tables, the first three columns display the name of the instance and the values of γ and Q, while the remaining columns show the average travel cost (over 30 trials) of each instance of the different heuristics. From these tables, one can observe that the quality of the results depend on the values of γ and Q. For a fixed value of γ, the average travel cost tend to be slightly better as Q increases. The improvement can be explained by the fact that there are more feasible positions in the tour to insert pickup nodes into it when Q is increased. For a fixed value of Q, a substantial improvement of

the results is observed with an increasing γ. When γ is increased, fewer pickup nodes are needed. Hence, the total travel cost decreases.

The fourth and fifth columns show the results obtained from GRASP over 100 and 200 iterations, respectively. When running GRASP for more iterations, the average improvement is 0.27%. The next four columns show the results from the different variants of GRASP+PR over 50 and 100 iterations, respectively. As expected, the standard version is slightly better with an average improvement of 0.001%. In order to have a fair comparison between GRASP and GRASP+PR, the computing time consumed should be approximately the same between the heuristics. The average computing times of the heuristics are depicted in Table 5. It can be observed that running GRASP over 100 (200) iterations and GRASP+PR over 50 (100) iterations consumes about equal amount of computing time, but GRASP+PR has a better performance. The average gap between GRASP and GRASP+PR is 0.74% (0.77%).

We have shown that GRASP+PR produces better solutions than GRASP. In the following, we will further assess its performance by comparing it to MA proposed by Ting and Liao (2013) on the same set of instances.

4.3 Comparing GRASP+PR with MA

As the difference between the two variants of GRASP+PR is negligible (0.001%). The comparison is with the slightly faster version. Tables 6 and 7 provide a comparison on the results obtained from MA and GRASP+PR. The fourth column shows the average travel cost (over 30 trials) obtained from MA, while the remaining columns list the average and best results from GRASP+PR over 50 and 100 iterations, respectively. MA performs better than GRASP+PR (with $\kappa = 50$) on only six out of 90 instances, while GRASP+PR produces better solutions on the remaining instances. The average deviation between the two is 5.44%. GRASP+PR (with $\kappa = 100$) performs even better, with 88 instances being better than MA. The average deviation is increased to 5.72%. The results will be even better if κ is increased, but at the expense of the computing time. Ting and Liao have shown that by adding a local search component (2-opt) to GA (which is the MA), better results are obtained. Due to maintaining feasibility of a complete tour (where both pickup and delivery nodes exist), Ting and Liao modified the 2-opt operator where a tour is partitioned into segments and 2-opt is applied on each of the segments rather than on the whole tour. However, our 2-opt is applied on a tour consisting of delivery nodes only and few pickup nodes are inserted at a later stage. This two-stage search strategy shows to be beneficial for this problem as GRASP alone also outperforms MA. The average gap is 4.75% (with $\kappa = 100$) and 5% (with $\kappa = 200$).

The excellence of the proposed method is also statistically significant. The Wilcoxon signed rank test (Golden & Stewart, 1985) has been applied to compare MA with GRASP+PR, as well as MA with GRASP. The null hypothesis is that both MA and GRASP+PR (or GRASP) perform equally good, while the alternative hypothesis is that GRASP+PR (or GRASP) performs better than MA. The significance level is set to be 0.05. The sum of the signed ranks has been calculated as 4033 (or 3489). The corresponding test statistic is 8.113 (or 7.019), which is larger than the critical value of 1.645. Therefore, the null

hypothesis is rejected. As a matter of fact, we conclude that GRASP+PR (or GRASP) performs better MA under the current settings.

It is difficult to compare the CPU time between MA and GRASP+PR as Ting and Liao (2013) did not report the computing times for running their heuristic. They only included two plots of the solution value against the running time for two of the instances (n100mosA(91) and n500mosA(453)). The final running times on the plots for the two instances are about 10 and 40 seconds, respectively. Their experiments were conducted on an Intel core i7-920 computer, which is faster than our computer. Based on the two instances, GRASP+PR (with $\kappa = 50$) is definitely faster than MA. On the other hand, it is difficult to conclude which method is faster when GRASP+PR is set to run for $\kappa = 100$ iterations.

5 Conclusions

In this paper, we present a hybrid heuristic consisting of GRASP and path relinking for the selective pickup and delivery problem, which is motivated by a real-world bicycle repositioning problem. Path relinking makes it possible to further diversify and intensify the search around the local minima obtained by GRASP. Our proposed path relinking uses a different strategy compared with those in the literature. Instead of terminating the path at the guiding solution, the path might be truncated before reaching the guiding solution, or the path might be extended beyond the guiding solution. In addition, the search in our hybrid heuristic is based on two solution spaces: the infeasible space (where the tours consist of delivery nodes only) and the feasible space (where the tours are feasible consisting of both delivery and pickup nodes). The algorithm mainly looks for good tours in the infeasible space, and switches to the feasible space at the end of the algorithm. The heuristic is easy to understand, simple to implement, efficient and effective. These characteristics are important for managing real-time bike repositioning operations.

Experiments were conducted on 90 benchmark instances. Computational experiments show that including path relinking within the GRASP framework improves the GRASP results by 0.77% on average without additional computing time. Further, experiments also show that the hybrid heuristic improves the existing results in the literature with an average and maximum improvement of 5.72% and 6.79%, respectively. This improvement can be further accentuated by increasing the values of R_{max} and κ, but at the expense of increased computing time.

The proposed algorithm attains better results than MA because the proposed algorithm makes use of the feature that only few pickup nodes are visited eventually. More effort is thus spent on searching for the best sequence of delivery nodes before adding a few pickup nodes to obtain a more accurate solution. The main lesson learned is that it is important to incorporate the properties of the problem into the algorithm to obtain good solutions quickly.

Although the results show that our proposed heuristic performs better than MA, there are a few limitations related to the algorithmic aspect and findings. One limitation with the proposed heuristic is that there is still a 1% gap between the average improvement and

the maximum improvement of the results. This indicates that it is not always possible to achieve solutions of superior quality. Another limitation is that the proposed heuristic may not work very well when there is a need of a large amount of pickup nodes, because the proposed algorithm first focuses on generating good tours that consist of delivery nodes only and pickup nodes are then added at the end to make the tours complete and feasible. The third limitation is that our comparisons might be a bit limited as we only compared GRASP+PR with MA, which was in turn compared with GA and tabu search by Ting and Liao (2013). However, these are the only existing algorithms in the literature developed for the problem under study. One more limitation is that we only used the 90 instances provided by Ting and Liao (2013) for comparison and hence our experimental findings and conclusion are based on these instances. However, the instances do not capture the cases of small vehicle capacity, or varying the ratio between the numbers of pickup nodes and delivery nodes.

The following future works can be performed. First, as a remedy to the first limitation of the proposed algorithm, we could replace the local search component in Section 3.2 with a more sophisticated method (e.g., tabu search). Instead of a greedy randomized construction heuristic, we could have a ruin and recreate procedure where a solution is not constructed from scratch (Schrimpf, Schneider, Stamm-Wilbrandt, & Dueck, 2000). Currently, randomness can only be found in the construction heuristic. We could also apply randomness in the path relinking procedure. For example, instead of making greedy choices when creating the path, we could make random choices or greedy random choices equivalent to how the restricted candidate list RCL is made. Second, to address the second algorithmic limitation, we could pass the feasible tours to another procedure for post-optimization purposes where different operators can be applied in order to further improve the solutions. Third, more future works could be focused on designing other search methods to address the third limitation regarding very few algorithm comparisons. Fourth, we can generate more test instances in order to check the validity of the conclusions. Lastly, besides the above algorithmic research directions, we could also look at the problem settings to make the problem more realistic, especially for day-time operations (e.g., multiple routes/vehicles, split deliveries/pickups, stochastic/dynamic selective pickup and delivery problems).

Acknowledgments

This work was partially supported by a grant from National Natural Science Foundation of China (71271183). This support is gratefully acknowledged. Thanks are also due to the two anonymous reviewers for their constructive comments.

References

Aiex, R. M., Binato, S., & Resende, M. G. C. (2003). Parallel GRASP with path-relinking for job shop scheduling. *Parallel Computing*, *29*(4), 393-430.

Anily, S., Gendreau, M., & Laporte, G. (1999). The swapping problem on a line. *SIAM Journal on Computing*, *29*(1), 327-335.

Anily, S., Gendreau, M., & Laporte, G. (2011). The preemptive swapping problem on a tree. *Networks*, *58*(2), 83-94.

Bent, R., & Van Hentenryck, P. (2006). A two-stage hybrid algorithm for pickup and delivery vehicle routing problems with time windows. *Computers & Operations Research*, *33*(4), 875-893.

Berbeglia, G., Cordeau, J.-F., Gribkovskaia, I., & Laporte, G. (2007). Static pickup and delivery problems: a classification scheme and survey. *TOP*, *15*(1), 1-31.

Bordenave, C., Gendreau, M., & Laporte, G. (2010). Heuristics for the mixed swapping problem. *Computers & Operations Research*, *37*(1), 108-114.

Bruck, B. P., dos Santos, A. G., & Arroyo, J. E. C. (2012, June). Hybrid metaheuristic for the single vehicle routing problem with deliveries and selective pickups. In *2012 IEEE Congress on Evolutionary Computation (CEC)* (p. 1-8).

Campos, V., Marti, R., Sánchez-Oro, J., & Duarte, A. (2014). Grasp with path relinking for the orienteering problem. *Journal of the Operational Research Society*, *65*(12), 1800–1813.

Çatay, B. (2010). A new saving-based ant algorithm for the vehicle routing problem with simultaneous pickup and delivery. *Expert Systems with Applications*, *37*(10), 6809-6817.

Cordeau, J.-F., Dell'Amico, M., & Iori, M. (2010). Branch-and-cut for the pickup and delivery traveling salesman problem with FIFO loading. *Computers & Operations Research*, *37*(5), 970-980.

Cordeau, J.-F., Iori, M., Laporte, G., & Salazar González, J. J. (2010). A branch-and-cut algorithm for the pickup and delivery traveling salesman problem with LIFO loading. *Networks*, *55*(1), 46-59.

de Oliveira, F. B., Enayatifar, R., Sadaei, H. J., Guimarães, F. G., & Potvin, J.-Y. (2015). A cooperative coevolutionary algorithm for the multi-depot vehicle routing problem. *Expert Systems with Applications*. (In press)

de Oliveira, R. M., Lorena, L. A. N., Chaves, A. A., & Mauri, G. R. (2014). Hybrid heuristics based on column generation with path-relinking for clustering problems. *Expert Systems with Applications*, *41*(11), 5277-5284.

Deng, Y., & Bard, J. F. (2011). A reactive GRASP with path relinking for capacitated clustering. *Journal of Heuristics*, *17*(2), 119-152.

Dumitrescu, I., Ropke, S., Cordeau, J.-F., & Laporte, G. (2010). The traveling salesman problem with pickup and delivery: polyhedral results and a branch-and-cut algorithm. *Mathematical Programming*, *121*(2), 269-305.

Erdoğan, G., Cordeau, J.-F., & Laporte, G. (2009). The pickup and delivery traveling salesman problem with first-in-first-out loading. *Computers & Operations Research*, *36*(6), 1800-1808.

Erdoğan, G., Cordeau, J.-F., & Laporte, G. (2010). A branch-and-cut algorithm for solving the non-preemptive capacitated swapping problem. *Discrete Applied Mathematics*, *158*(15), 1599-1614.

Falcon, R., Li, X., Nayak, A., & Stojmenovic, I. (2010, July). The one-commodity traveling salesman problem with selective pickup and delivery: An ant colony approach. In *2010 IEEE Congress on Evolutionary Computation (CEC)* (p. 1-8).

Feo, T. A., & Resende, M. G. C. (1995). Greedy randomized adaptive search procedures. *Journal of Global Optimization*, *6*(2), 109-133.

Festa, P., & Resende, M. G. C. (2011). GRASP: basic components and enhancements. *Telecommunication Systems*, *46*(3), 253-271.

Glover, F., & Laguna, M. (1993). Tabu search. In C. R. Reeves (Ed.), *Modern Heuristic Techniques for Combinatorial Problems* (p. 70-150). Oxford: Blackwell Scientific Publishing.

Golden, B. L., & Stewart, W. R. (1985). Empirical analysis of heuristics. In E. L. Lawler, J. K. Lenstra, A. H. G. R. Kan, & D. B. Shmoys (Eds.), *The Traveling Salesman Problem* (p. 207-249). Wiley, Chichester.

Gribkovskaia, I., Halskau sr., Ø., Laporte, G., & Vlček, M. (2007). General solutions to the single vehicle routing problem with pickups and deliveries. *European Journal of Operational Research*, *180*(2), 568-584.

Gribkovskaia, I., Laporte, G., & Shyshou, A. (2008). The single vehicle routing problem with deliveries and selective pickups. *Computers & Operations Research*, *35*(9), 2908-2924.

Gutiérrez-Jarpa, G., Desaulniers, G., Laporte, G., & Marianov, V. (2010). A branch-and-price algorithm for the vehicle routing problem with deliveries, selective pickups and time windows. *European Journal of Operational Research*, *206*(2), 341-349.

Hamdi-Dhaoui, K., Labadie, N., & Yalaoui, A. (2014). The bi-objective two-dimensional loading vehicle routing problem with partial conflicts. *International Journal of Production Research*, *52*(19), 5565-5582.

Hernández-Pérez, H., Rodríguez-Martín, I., & Salazar-González, J.-J. (2009). A hybrid GRASP/VND heuristic for the one-commodity pickup-and-delivery traveling salesman problem. *Computers & Operations Research*, *36*(5), 1639-1645.

Hernández-Pérez, H., & Salazar-González, J.-J. (2004). Heuristics for the one-commodity pickup-and-delivery traveling salesman problem. *Transportation Science*, *38*(2), 245-255.

Hernández-Pérez, H., & Salazar-González, J.-J. (2009). The multi-commodity one-to-one pickup-and-delivery traveling salesman problem. *European Journal of Operational Research*, *196*(3), 987-995.

Hernández-Pérez, H., & Salazar-González, J.-J. (2014). The multi-commodity pickup-and-delivery traveling salesman problem. *Networks*, *63*(1), 46-59.

Ho, S. C., & Gendreau, M. (2006). Path relinking for the vehicle routing problem. *Journal of Heuristics*, *12*(1-2), 55-72.

Jia, S., & Hu, Z.-H. (2014). Path-relinking tabu search for the multi-objective flexible job shop scheduling problem. *Computers & Operations Research*, *47*, 11-26.

Laguna, M., & Martí, R. (1999). GRASP and path relinking for 2-layer straight line crossing minimization. *INFORMS Journal on Computing*, *11*(1), 44-52.

Lai, X., & Hao, J.-K. (2015). Path relinking for the fixed spectrum frequency assignment problem. *Expert Systems with Applications*, *42*(10), 4755 - 4767.

Li, J., Chu, F., Prins, C., & Zhu, Z. (2014). Lower and upper bounds for a two-stage capacitated facility location problem with handling costs. *European Journal of Operational Research*, *236*(3), 957 - 967.

Li, J., Pardalos, P. M., Sun, H., Pei, J., & Zhang, Y. (2015). Iterated local search embedded adaptive neighborhood selection approach for the multi-depot vehicle routing problem with simultaneous deliveries and pickups. *Expert Systems with Applications*, *42*(7), 3551-3561.

Lin, S. (1965). Computer solutions of the traveling salesman problem. *Bell System Technical Journal*, *44*, 2245-2269.

Marinakis, Y., & Marinaki, M. (2015). Combinatorial neighborhood topology bumble bees mating optimization for the vehicle routing problem with stochastic demands.

Soft Computing, *19*(2), 353-373.

Martí, R., Corberán, Á., & Peiró, J. (2015). Scatter search for an uncapacitated p-hub median problem. *Computers & Operations Research*, *58*, 53-66.

Mladenović, N., Urošević, D., Hanafi, S., & Ilić, A. (2012). A general variable neighborhood search for the one-commodity pickup-and-delivery travelling salesman problem. *European Journal of Operational Research*, *220*(1), 270-285.

Morán-Mirabal, L., González-Velarde, J., & Resende, M. (2014). Randomized heuristics for the family traveling salesperson problem. *International Transactions in Operational Research*, *21*(1), 41-57.

Nanry, W. P., & Barnes, J. W. (2000). Solving the pickup and delivery problem with time windows using reactive tabu search. *Transportation Research Part B*, *34*(2), 107-121.

Peng, B., Lü, Z., & Cheng, T. (2015). A tabu search/path relinking algorithm to solve the job shop scheduling problem. *Computers & Operations Research*, *53*, 154-164.

Reghioui, M., Prins, C., & Labadi, N. (2007). GRASP with path relinking for the capacitated arc routing problem with time windows. In M. Giacobini (Ed.), *Applications of Evolutionary Computing* (Vol. 4448, p. 722-731). Springer Berlin Heidelberg.

Resende, M. G. C., & Ribeiro, C. C. (2010). Greedy randomized adaptive search procedures: Advances, hybridizations, and applications. In M. Gendreau & J.-Y. Potvin (Eds.), *Handbook of Metaheuristics* (p. 283-319). Springer US.

Ribas, I., Companys, R., & Tort-Martorell, X. (2015). An efficient discrete artificial bee colony algorithm for the blocking flow shop problem with total flowtime minimization. *Expert Systems with Applications*, *42*(15-16), 6155-6167.

Ribeiro, C. C., & Resende, M. G. (2012). Path-relinking intensification methods for stochastic local search algorithms. *Journal of Heuristics*, *18*(2), 193-214.

Ríos-Mercado, R. Z., & Escalante, H. J. (2015). GRASP with path relinking for commercial districting. *Expert Systems with Applications*. (In press)

Santos, D., de Sousa, A., & Alvelos, F. (2013). A hybrid column generation with GRASP and path relinking for the network load balancing problem. *Computers & Operations Research*, *40*(12), 3147-3158.

Santos, D. O., & Xavier, E. C. (2015). Taxi and ride sharing: A dynamic dial-a-ride problem with money as an incentive. *Expert Systems with Applications*, *42*(19), 6728-6737.

Scaparra, M. P., & Church, R. L. (2005). A GRASP and path relinking heuristic for rural road network development. *Journal of Heuristics*, *11*(1), 89-108.

Schrimpf, G., Schneider, J., Stamm-Wilbrandt, H., & Dueck, G. (2000). Record breaking optimization results using the ruin and recreate principle. *Journal of Computational Physics*, *159*(2), 139-171.

Souffriau, W., Vansteenwegen, P., Berghe, G. V., & Oudheusden, D. V. (2010). A path relinking approach for the team orienteering problem. *Computers & Operations Research*, *37*(11), 1853-1859.

Ting, C.-K., & Liao, X.-L. (2013). The selective pickup and delivery problem: Formulation and a memetic algorithm. *International Journal of Production Economics*, *141*(1), 199-211.

Urrutia, S., Milanés, A., & Løkketangen, A. (2015). A dynamic programming based local search approach for the double traveling salesman problem with multiple stacks. *International Transactions in Operational Research*, *22*(1), 61-75.

Usberti, F. L., França, P. M., & França, A. L. M. (2013). GRASP with evolutionary

path-relinking for the capacitated arc routing problem. *Computers & Operations Research*, *40*(12), 3206-3217.

Wang, H.-F., & Chen, Y.-Y. (2013). A coevolutionary algorithm for the flexible delivery and pickup problem with time windows. *International Journal of Production Economics*, *141*(1), 4-13.

Yang, Z., Zhang, G., & Zhu, H. (2015). Multi-neighborhood based path relinking for two-sided assembly line balancing problem. *Journal of Combinatorial Optimization*, 1-20. (In press)

Zachariadis, E. E., & Kiranoudis, C. T. (2011). A local search metaheuristic algorithm for the vehicle routing problem with simultaneous pick-ups and deliveries. *Expert Systems with Applications*, *38*(3), 2717-2726.

Zachariadis, E. E., Tarantilis, C. D., & Kiranoudis, C. T. (2009). A hybrid metaheuristic algorithm for the vehicle routing problem with simultaneous delivery and pick-up service. *Expert Systems with Applications*, *36*(2, Part 1), 1070-1081.

Zhao, F., Li, S., Sun, J., & Mei, D. (2009). Genetic algorithm for the one-commodity pickup-and-delivery traveling salesman problem. *Computers & Industrial Engineering*, *56*(4), 1642-1648.

Table 3: Average travel costs for GRASP and GRASP+PR

Instance	γ	Q	GRASP $\kappa = 100$	GRASP $\kappa = 200$	GRASP+PR $\kappa = 50$	GRASP+PR[1] $\kappa = 50$	GRASP+PR $\kappa = 100$	GRASP+PR[1] $\kappa = 100$
n100mosA(91)/42	100	400	5169.63	5167.50	5167.50	5167.50	5167.50	5167.50
		600	5169.63	5167.50	5167.50	5167.50	5167.50	5167.50
		1000	5169.63	5167.50	5167.50	5167.50	5167.50	5167.50
	200	400	5169.12	5166.99	5166.99	5166.99	5166.99	5166.99
		600	5169.12	5166.99	5166.99	5166.99	5166.99	5166.99
		1000	5169.12	5166.99	5166.99	5166.99	5166.99	5166.99
	400	600	5168.96	5166.83	5166.83	5166.83	5166.83	5166.83
		800	5168.96	5166.83	5166.83	5166.83	5166.83	5166.83
		1000	5168.96	5166.83	5166.83	5166.83	5166.83	5166.83
n100mosB(92)/47	100	400	5153.76	5153.76	5153.76	5153.76	5153.76	5153.76
		600	5153.76	5153.76	5153.76	5153.76	5153.76	5153.76
		1000	5153.76	5153.76	5153.76	5153.76	5153.76	5153.76
	200	400	5153.52	5153.52	5153.52	5153.52	5153.52	5153.52
		600	5153.52	5153.52	5153.52	5153.52	5153.52	5153.52
		1000	5153.52	5153.52	5153.52	5153.52	5153.52	5153.52
	400	600	5153.41	5153.41	5153.41	5153.41	5153.41	5153.41
		800	5153.41	5153.41	5153.41	5153.41	5153.41	5153.41
		1000	5153.41	5153.41	5153.41	5153.41	5153.41	5153.41
n200mosA(181)/94	100	400	7461.99	7438.99	7415.55	7415.55	7389.73	7390.03
		600	7461.99	7438.99	7415.55	7415.55	7389.73	7390.03
		1000	7461.99	7438.99	7415.55	7415.55	7389.73	7390.03
	200	400	7460.89	7437.88	7414.63	7414.63	7388.79	7389.07
		600	7460.18	7437.29	7414.19	7414.19	7388.08	7388.40
		1000	7460.18	7437.29	7414.19	7414.19	7388.08	7388.40
	400	600	7460.71	7437.72	7414.55	7414.55	7388.74	7389.02
		800	7459.83	7436.96	7413.93	7413.93	7387.73	7388.06
		1000	7459.83	7436.96	7413.93	7413.93	7387.73	7388.06
n200mosB(184)/88	100	400	8240.86	8204.49	8149.48	8150.15	8121.35	8121.35
		600	8240.86	8204.48	8149.48	8150.15	8121.35	8121.35
		1000	8240.86	8204.48	8149.48	8150.15	8121.35	8121.35
	200	400	8240.16	8204.14	8148.58	8149.25	8120.48	8120.48
		600	8239.81	8203.66	8147.97	8148.64	8119.92	8119.92
		1000	8239.81	8203.66	8147.97	8148.64	8119.92	8119.92
	400	600	8240.01	8204.02	8148.37	8149.03	8120.33	8120.33
		800	8239.64	8203.48	8147.81	8148.47	8119.73	8119.73
		1000	8239.64	8203.48	8147.81	8148.47	8119.73	8119.73
n300mosA(181)/94	100	400	9188.24	9161.32	9112.40	9112.40	9048.53	9048.74
		600	9188.24	9161.32	9112.40	9112.40	9048.53	9048.74
		1000	9188.24	9161.32	9112.40	9112.40	9048.53	9048.74
	200	400	9187.43	9160.00	9111.39	9111.39	9047.80	9047.86
		600	9187.42	9159.99	9111.33	9111.33	9047.78	9047.83
		1000	9187.42	9159.99	9111.33	9111.33	9047.78	9047.83
	400	600	9187.34	9159.88	9111.20	9111.20	9047.70	9047.75
		800	9187.30	9159.85	9111.17	9111.17	9047.66	9047.71
		1000	9187.30	9159.85	9111.17	9111.17	9047.66	9047.71

[1] `AddPickUpNodes` is only applied to the tours in the reference set $\mathcal{R}$ after κ iterations have been completed.

Table 4: Average travel costs for GRASP and GRASP+PR (cont'd)

Instance	γ	Q	GRASP $\kappa = 100$	GRASP $\kappa = 200$	GRASP+PR $\kappa = 50$	GRASP+PR[1] $\kappa = 50$	GRASP+PR $\kappa = 100$	GRASP+PR[1] $\kappa = 100$
n300mosB(279)/138	100	400	9318.91	9289.10	9237.74	9237.74	9189.72	9189.82
		600	9318.88	9289.05	9237.74	9237.74	9189.70	9189.80
		1000	9318.88	9289.05	9237.74	9237.74	9189.70	9189.80
	200	400	9317.34	9287.16	9235.96	9235.96	9188.18	9188.22
		600	9317.13	9286.97	9235.78	9235.78	9187.94	9187.97
		1000	9317.03	9286.89	9235.67	9235.67	9187.82	9187.85
	400	600	9316.91	9286.66	9235.54	9235.54	9187.70	9187.75
		800	9316.69	9286.54	9235.37	9235.37	9187.55	9187.59
		1000	9316.69	9286.54	9235.37	9235.37	9187.55	9187.59
n400mosA(358)/172	100	400	11095.70	11072.80	11030.90	11030.90	11016.20	11016.20
		600	11095.70	11072.80	11030.90	11030.90	11016.20	11016.20
		1000	11095.70	11072.80	11030.90	11030.90	11016.20	11016.20
	200	400	11094.50	11071.50	11029.40	11029.40	11014.80	11014.80
		600	11094.30	11071.30	11029.20	11029.20	11014.60	11014.60
		1000	11094.30	11071.30	11029.20	11029.20	11014.60	11014.60
	400	600	11094.20	11071.30	11029.00	11029.00	11014.50	11014.50
		800	11094.10	11071.10	11028.80	11028.80	11014.30	11014.30
		1000	11094.00	11071.10	11028.80	11028.80	11014.30	11014.30
n400mosB(364)/183	100	400	10868.20	10828.50	10757.50	10758.10	10740.60	10741.40
		600	10868.10	10828.20	10757.40	10758.10	10740.50	10741.30
		1000	10868.10	10828.20	10757.40	10758.10	10740.50	10741.30
	200	400	10865.80	10826.10	10754.70	10755.30	10737.80	10738.50
		600	10865.10	10825.50	10754.10	10754.70	10737.20	10738.00
		1000	10865.10	10825.50	10754.00	10754.70	10737.10	10737.90
	400	600	10865.10	10825.60	10753.80	10754.50	10737.00	10737.80
		800	10864.60	10825.00	10753.40	10754.00	10736.50	10737.30
		1000	10864.60	10824.90	10753.30	10754.00	10736.50	10737.30
n500mosA(453)/232	100	400	11823.80	11777.10	11712.20	11712.20	11661.00	11661.00
		600	11823.70	11777.00	11712.10	11712.10	11660.90	11660.90
		1000	11823.70	11777.00	11712.10	11712.10	11660.90	11660.90
	200	400	11822.00	11774.50	11709.80	11709.80	11658.60	11658.60
		600	11821.60	11774.20	11709.50	11709.50	11658.30	11658.30
		1000	11821.60	11774.20	11709.40	11709.50	11658.30	11658.30
	400	600	11821.60	11774.30	11709.50	11709.60	11658.50	11658.50
		800	11821.30	11773.80	11709.20	11709.20	11658.10	11658.10
		1000	11821.30	11773.80	11709.20	11709.20	11658.10	11658.10
n500mosB(454)/228	100	400	12306.90	12267.60	12136.00	12136.00	12097.30	12097.30
		600	12306.80	12267.50	12135.90	12135.90	12097.10	12097.10
		1000	12306.70	12267.50	12135.90	12135.90	12097.10	12097.10
	200	400	12305.70	12266.30	12134.50	12134.50	12095.90	12095.90
		600	12305.40	12265.90	12134.00	12134.00	12095.30	12095.30
		1000	12305.40	12265.90	12134.00	12134.00	12095.30	12095.30
	400	600	12305.40	12265.70	12133.90	12133.90	12095.30	12095.30
		800	12305.20	12265.60	12133.70	12133.70	12095.00	12095.00
		1000	12305.10	12265.60	12133.60	12133.60	12095.00	12095.00

[1] `AddPickUpNodes` is only applied to the tours in the reference set $\mathcal{R}$ after κ iterations have been completed.

Table 5: Average CPU times (in seconds) for the different classes of instances for the different heuristics

Instance	GRASP $\kappa = 100$	GRASP $\kappa = 200$	GRASP+PR $\kappa = 50$	GRASP+PR[1] $\kappa = 50$	GRASP+PR $\kappa = 100$	GRASP+PR[1] $\kappa = 100$
n100mosA(91)/42	0.30	0.58	0.67	0.66	1.43	1.31
n100mosB(92)/47	0.25	0.48	0.61	0.57	1.32	1.16
n200mosA(181)/94	1.88	3.67	2.62	2.34	5.63	4.88
n200mosB(184)/88	2.31	4.42	3.42	3.11	7.27	6.56
n300mosA(181)/94	7.20	13.97	9.12	8.00	19.26	16.80
n300mosB(279)/138	8.08	15.65	9.18	8.11	19.82	16.65
n400mosA(358)/172	20.12	40.18	21.56	18.28	45.07	37.18
n400mosB(364)/183	19.04	37.97	20.52	17.24	42.41	34.94
n500mosA(453)/232	37.74	75.32	35.75	28.60	73.28	59.34
n500mosB(454)/228	38.44	77.07	36.98	30.53	76.33	63.77
Average time	13.54	26.93	14.04	11.74	29.18	24.26

[1] `AddPickUpNodes` is only applied to the tours in the reference set $\mathcal{R}$ after κ iterations have been completed.

Table 6: Average travel costs for MA and GRASP+PR

Instance	γ	Q	MA	GRASP+PR $\kappa = 50$	GRASP+PR $\kappa = 50$	GRASP+PR $\kappa = 100$	GRASP+PR $\kappa = 100$
			Average	*Average*	*Best*	*Average*	*Best*
n100mosA(91)/42	100	400	5274.23	5167.50	5167.50	5167.50	5167.50
		600	5232.19	5167.50	5167.50	5167.50	5167.50
		1000	5243.14	5167.50	5167.50	5167.50	5167.50
	200	400	5268.42	5166.99	5166.99	5166.99	5166.99
		600	5215.64	5166.99	5166.99	5166.99	5166.99
		1000	5189.47	5166.99	5166.99	5166.99	5166.99
	400	600	5170.98	5166.83	5166.83	5166.83	5166.83
		800	5168.91	5166.83	5166.83	5166.83	5166.83
		1000	5167.87	5166.83	5166.83	5166.83	5166.83
n100mosB(92)/47	100	400	5272.99	5153.76	5153.76	5153.76	5153.76
		600	5261.24	5153.76	5153.76	5153.76	5153.76
		1000	5263.58	5153.76	5153.76	5153.76	5153.76
	200	400	5231.55	5153.52	5153.52	5153.52	5153.52
		600	5236.76	5153.52	5153.52	5153.52	5153.52
		1000	5185.84	5153.52	5153.52	5153.52	5153.52
	400	600	5172.61	5153.41	5153.41	5153.41	5153.41
		800	5156.03	5153.41	5153.41	5153.41	5153.41
		1000	5160.39	5153.41	5153.41	5153.41	5153.41
n200mosA(181)/94	100	400	7646.87	7415.55	7338.39	7390.03	7338.39
		600	7560.72	7415.55	7338.39	7390.03	7338.39
		1000	7549.46	7415.55	7338.39	7390.03	7338.39
	200	400	7715.88	7414.63	7337.06	7389.07	7337.06
		600	7515.59	7414.19	7336.01	7388.40	7336.01
		1000	7457.90	7414.19	7336.01	7388.40	7336.01
	400	600	7581.82	7414.55	7336.01	7389.02	7336.01
		800	7471.26	7413.93	7335.37	7388.06	7335.37
		1000	7418.96	7413.93	7335.37	7388.06	7335.37
n200mosB(184)/88	100	400	8329.73	8150.15	8024.34	8121.35	8024.34
		600	8289.96	8150.15	8024.34	8121.35	8024.34
		1000	8211.91	8150.15	8024.34	8121.35	8024.34
	200	400	8433.09	8149.25	8022.70	8120.48	8022.70
		600	8233.01	8148.64	8021.54	8119.92	8021.54
		1000	8187.13	8148.64	8021.54	8119.92	8021.54
	400	600	8292.87	8149.03	8021.54	8120.33	8021.54
		800	8178.86	8148.47	8021.42	8119.73	8021.42
		1000	8149.09	8148.47	8021.42	8119.73	8021.42
n300mosA(181)/94	100	400	9613.65	9112.40	8978.63	9048.74	8944.07
		600	9321.98	9112.40	8978.63	9048.74	8944.07
		1000	9239.73	9112.40	8978.63	9048.74	8944.07
	200	400	9612.50	9111.39	8978.33	9047.86	8943.58
		600	9289.82	9111.33	8978.33	9047.83	8943.58
		1000	9148.37	9111.33	8978.33	9047.83	8943.58
	400	600	9228.90	9111.20	8978.30	9047.75	8943.51
		800	9156.36	9111.17	8978.22	9047.71	8943.50
		1000	9104.33	9111.17	8978.22	9047.71	8943.50

Table 7: Average travel costs for MA and GRASP+PR (cont'd)

Instance	γ	Q	MA	GRASP+PR $\kappa = 50$	GRASP+PR $\kappa = 50$	GRASP+PR $\kappa = 100$	GRASP+PR $\kappa = 100$
			Average	*Average*	*Best*	*Average*	*Best*
n300mosB(279)/138	100	400	9807.17	9237.74	9132.86	9189.82	9055.51
		600	9476.11	9237.74	9132.61	9189.80	9055.42
		1000	9412.23	9237.74	9132.61	9189.80	9055.42
	200	400	9816.74	9235.96	9132.53	9188.22	9054.67
		600	9377.53	9235.78	9132.08	9187.97	9054.43
		1000	9270.51	9235.67	9131.58	9187.85	9054.23
	400	600	9376.53	9235.54	9131.58	9187.75	9054.14
		800	9189.02	9235.37	9131.56	9187.59	9054.14
		1000	9189.70	9235.37	9131.56	9187.59	9054.14
n400mosA(358)/172	100	400	13303.94	11030.90	10822.30	11016.20	10822.30
		600	11802.30	11030.90	10822.30	11016.20	10822.30
		1000	11912.59	11030.90	10822.30	11016.20	10822.30
	200	400	12940.68	11029.40	10821.30	11014.80	10821.30
		600	11447.40	11029.20	10821.00	11014.60	10821.00
		1000	10972.00	11029.20	10821.00	11014.60	10821.00
	400	600	11781.90	11029.00	10821.00	11014.50	10821.00
		800	11259.50	11028.80	10820.70	11014.30	10820.70
		1000	11018.23	11028.80	10820.70	11014.30	10820.70
n400mosB(364)/183	100	400	13338.54	10758.10	10518.80	10741.40	10516.30
		600	11687.35	10758.10	10518.80	10741.30	10516.30
		1000	10996.21	10758.10	10518.80	10741.30	10516.30
	200	400	12937.36	10755.30	10515.50	10738.50	10513.00
		600	11311.26	10754.70	10514.30	10738.00	10511.90
		1000	10694.08	10754.70	10514.30	10737.90	10511.90
	400	600	11652.05	10754.50	10514.30	10737.80	10511.90
		800	11030.08	10754.00	10514.00	10737.30	10511.50
		1000	10772.36	10754.00	10514.00	10737.30	10511.50
n500mosA(453)/232	100	400	19272.46	11712.20	11550.50	11661.00	11417.30
		600	14664.66	11712.10	11550.50	11660.90	11417.30
		1000	13965.66	11712.10	11550.50	11660.90	11417.30
	200	400	18218.14	11709.80	11550.10	11658.60	11413.40
		600	14321.33	11709.50	11550.10	11658.30	11412.50
		1000	12151.28	11709.50	11550.10	11658.30	11412.50
	400	600	13691.69	11709.60	11550.10	11658.50	11412.50
		800	12444.84	11709.20	11550.10	11658.10	11411.90
		1000	12347.79	11709.20	11550.10	11658.10	11411.90
n500mosB(454)/228	100	400	18430.17	12136.00	11956.80	12097.30	11956.80
		600	14450.89	12135.90	11956.80	12097.10	11956.80
		1000	13477.41	12135.90	11956.80	12097.10	11956.80
	200	400	18041.28	12134.50	11955.30	12095.90	11955.30
		600	14363.94	12134.00	11955.10	12095.30	11955.10
		1000	12455.25	12134.00	11955.10	12095.30	11955.10
	400	600	13752.26	12133.90	11955.10	12095.30	11955.10
		800	12883.73	12133.70	11955.00	12095.00	11955.00
		1000	12350.01	12133.60	11955.00	12095.00	11955.00