Adapting the Large Neighborhood Search to Effectively Solve Pickup and Delivery Problems with Time Windows

Vincent Tam and M.C. Kwan

Department of Electrical and Electronic Engineering
University of Hong Kong, Pokfulam, Hong Kong.
vtam@eee.hku.hk

Subject areas : heuristic search, scheduling.

Abstract

Pickup and delivery problems with time windows (PDP-TW) are challenging combinatorial optimization problems widely occurring in modern distribution and transportation industry. A previous proposal successfully adapted and combined the well-known push forward insertion heuristic (PFIH) with a new repair-based swap operator as a computationally reduced version of the Large Neighborhood Search (LNS) to efficiently solve the PDP-TWs. In this paper, we focus on a more systematic scheme to adapt the LNS in order to effectively solve PDP-TWs. Up to our knowledge, this work represents the first attempt to systematically adapt LNS for solving PDP-TWs. Besides, the empirical results obtained by our search proposals affirmed the attractive capability of the LNS approach to effectively reduce the total distance traveled in solving PDP-TWs, with similar results also achieved by the original LNS method in tackling the conventional vehicle routing problems with time windows. Lastly, our proposal of adapting LNS to solve hard combinatorial problems opens up many possible directions for future investigations.

1 Introduction

The paper is organized as follows. Section 2 clearly gives a formal definition of PDP-TWs, and describes in greater detail about the related works such as the LNS and *Swap* operator to repair the current routing plan when solving PDP-TWs. Section 3 carefully considers our LNS based search proposal and the new and systematic removal guiding scheme to effectively solve PDP-TWs. Section 4 carefully evaluates the empirical results of our adapted LNS based search framework against those of Li and Lim's meta-heuristic approach [7] on the modified Solomon's benchmarks. Finally, we conclude our work in Section 5.

2 The Background and Related Works

2.1 The Pickup and Delivery Problems with Time Windows

Pickup and delivery problems with time windows (PDP-TWs) are intrinsically constrained optimization problems [8, 13]. Following [6, 7], a formal definition of a PDP-TW is clearly stated as follows.

Given a node set $N = \{n_0, n_1, n_2, n_3, \ldots, n_m\}$ where n_0 always denotes the depot, n_1 to n_m denote delivery or pickup locations for customers' requests, and the last index m is always an even number, a PDP-TW contains

- individual customer request as denoted by a pair of delivery and pickup locations: each delivery or pickup location n_i where $i > 0$ is associated with a customer demand q_i such that $q_i > 0$ for a pickup location whereas $q_i < 0$ for a delivery location, a service time s_i, that is the duration required to effectively

Proceedings of the 16th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2004)
1082-3409/04 $20.00 © 2004 **IEEE**

service the customer demand at that location, and an associated service time window $[e_i, l_i]$ where e_i and l_i denote the earliest and latest time to start the service as similarly defined in VRP-TWs;

- the delivery and pickup demand q_i and q_j belonging to the same customer are integers of the same magnitude but opposite signs such that $q_i + q_j = 0$ for ease of analysis;
- for any possible edge $\langle n_i, n_j \rangle$, both the non-negative distance d_{ij} and required travel time t_{ij} are specified;
- a set of problem constraints specified as follows.
    1. the *capacity* constraints: each vehicle has a fixed capacity C that cannot be exceeded. Each vehicle must carry an amount less than or equal to C;
    2. the *individual's time-window* constraints: a customer location can only be serviced within the associated service time window $[e_i, l_i]$. That is, only those edges $\langle n_i, n_j \rangle$ that satisfy their corresponding time-window constraints as $t_{0i} + s_i + t_{ij} \leq l_j$, restricting the concerned vehicle to arrive at or before the latest service time l_j after traveling from the depot to n_i to n_j with its completion of service at n_i, should be considered. On the contrary, if a vehicle reached the customer location earlier than e_i, the vehicle has to wait until e_i;
    3. the *global time-window* constraint: all vehicles depart from and return to the same depot n_0, and share the same constraints time window $[E, L]$, where E denotes the time a vehicle must have left the depot, and L denotes the time a vehicle must have returned to the depot;
    4. the *coupling* constraints: request that every pair of pickup and delivery locations must be serviced by the same vehicle;
    5. lastly, the *precedence* constraints: specify that the pickup location of a coupled customer requests must be serviced before the corresponding delivery location in the same route.

Following [9, 12], we considered consistently throughout this paper the common objective function of $TV \times TD$ where TV denotes the number of vehicles used and TD is the total distance traveled, as widely adopted in the conventional VRP-TWs. However, the objective function can vary across many different real-life applications. Taking the dial-a-ride application as an example, a common goal is usually to minimize the customers' inconvenience, often measured in term of the customers' total waiting time, as induced by any possible delay to cause the subsequent services occurred after the expected time.

2.2 The Large Neighborhood Search

To effectively solve the vehicle routing problems (VRPs), Shaw [10] proposed the large neighborhood search (LNS) that is actually an iterative process of relaxation and re-optimisation to continually improve on the existing routing plan until the convergence to a local minimum or resource exhaustion occurs. Relaxation refers to the removal of some selected customer requests from the current solution whereas re-optimisation is achieved by re-inserting the removed requests back into the routing plan for a smaller objective value using contraint propagation and heuristics such as the branch and bound (BnB) technique [13], and/or incomplete search methods [4, 8] that are aimed to explore only part of a search tree to hopefully find a better solution. Nevertheless, in LNS, one iteration of removal and re-insertion is considered as a *powerful move* for advancing the search to certain far-reaching and potential states that are otherwise possible by many more localized steps, thus the name *Large Neighborhood Search*. In essence, the performance of LNS largely depends on two key components: i) how to effectively choose customer requests for relaxation/removal; ii) how to effectively re-insert the removed customers to obtain a

better solution. It should be noted that due to the space limitation, it is impossible to have a very detailed discussion of the LNS here. Interested readers are strongly recommended to refer to [10] for more detail.

2.3 The *Swap* Operator

Tam and Tseng [12] independently proposed a heuristic-based *Swap* operator for solving PDP-TWs. The *Swap* operator works by a substantial modification of the current solution. From each vehicle in the fleet, *Swap* will randomly pick up a few pairs of pickup and delivery requests, remove them from the vehicle and add the pairs into a relocation pool. The major consideration is to keep the relocation pool having roughly the same number of customers in each iteration. As an example strategy, Tam and Tseng had arbitrarily set the *Swap* operator to remove around 1/5 of total number of customers into the relocation pool in the prototype implementation. Whenever the relocation pool is non-empty, the operator will randomly choose a pair of customers from relocation pool and insert them into any vehicle based on the common objective function defined in Section 2. In case there is no possible position in any route to insert the customer pair(s) in the relocation pool, a new vehicle would be created. Any empty vehicle resulting from the removal of customer pairs, should be removed from the fleet, thus showing the major strength of the *Swap* operator as a reduced version of the LNS approach to minimize the total number of routes.

3 Our Adapted LNS Proposal and Exploration Scheme

Here, we proposed an improved variant of the modified PFIH (mPFIH) as the best-fit PFIH (bPFIH) initialisation method described in the following steps:

step (1) - Start a new route to insert the 1st "request pair" from a list of request-pairs sorted in decreasing order of their *static* objective values (as defined previously);

step (2) - After examining all feasible positions in *all* existing route(s), insert the selected request-pair into positions giving the lowest *dynamic* objective value;

step (3) - When there is no feasible positions in any route, starts a new route for the selected request-pair;

step (4) - Goto step (1) until all request pairs are assigned.

To solve the VRPs, the original LNS approach relied on the number of failures to improve on the current solution to increase the size of neighborhood for exploration, including both removal and re-optimisation of visits. However, in handling PDP-TWs, the consecutive-failures-triggered strategy was not "responsive" enough to bring in significant improvement in the solution quality. Thus, we adopted an *active and resource-dependent* exploration scheme. The key idea is: the usually larger increase in the size of neighborhood is triggered by a predefined number of iterations as specified in an update schedule, rather than certain *dynamic and unpredictable* runtime behavior such as the occurrence of x consecutive failures to improve the current solution. To allow a systematic investigation, we suggested to introduce two parameters including a for the number of iterations required to increase the neighborhood size, and $\pm b$ denoting the actual change to the neighborhood size at every a iterations. As a result, each update schedule is best specified in a tabular form as the following example. Table 1

Iterations	a	$2a$	$3a$	$4a$	$5a$	$6a$	$7a$
Changes	$+b$	$+b$	$+b$	$+b$	$+b$	$+b$	$+b$

Table 1: An example update schedule/strategy for an iteratively increasing neighborhood size

shows an example update strategy, namely the *iINCR*, producing a continuing and iterative increase of $+b$ to the neighborhood size at every multiple of a iterations. The opposite *iDECR* strategy is unambiguously with a constant decrease of $-b$ at every a iterations.

4 An Empirical Evaluation

To show the effectiveness of our proposed improvement on the previous local search framework [4, 12], our improved LNS proposals using the best-fit push forward insertion heuristic (bPFIH) and the two opposite update strategies of *iINCR* and *iDECR* for aggressive exploration of large-sized neighborhood, forming our interested search prototypes: Adapted_LNS(bPFIH,*iINCR*) and Adapted_LNS(bPFIH,*iDECR*) were compared against the previous *Swap*-based optimizer *Swap*(mPFIH) [12], and Li and Lim's metaheuristic approach [7] integrating tabu search and simulated annealing on a set of 56 modified Solomon's benchmarks [6, 7] in which each problem instance has 100 customers. There are totally 6 distinct problem classes, namely LC1, LC2, LR1, LR2, LRC1, and LRC2. 'LC' refers to cases with clustered distribution of customers, 'LR' refers to cases with uniform distribution of customers, and 'RC' refers to mixed customers types. '1' refers to small vehicle capacity whereas '2' refers to large vehicle capacity.

All our search prototypes were implemented using the Microsoft Visual C++ Version 7.0 compiler, and tested on a desktop computer with the Intel Pentium IV processor of 2.8 GHz, 1,024 MB RAM, and a hard disk of 80 GB space. The operating system used was the Microsoft Windows XP. It should be noted that the *Swap*(mPFIH) optimizer with a constant relocation pool size of 10 request-pairs would halt after no improvement over 30 consecutive iterations whereas we set $a = 120$ and $b = \pm 5$ for both improved optimisers when changing from 0 to 720 iterations, or vice versa. Both adapted LNS optimizers would go through 2 rounds of such changes, giving a total of 1,440 iterations.

Table 2 compares their overall results, in terms of the total number of vehicles used (TV), the total distance traveled (TD) and their product as $TV * TD$ as the common objective function, of the different search proposals involved over all 56 modified Solomon's test cases. The smallest figure in each column was boldfaced for ease of comparison. Clearly, both of our

Optimizers	Overall Results		
	TV	TD	$TV * TD$
Swap(mPFIH)	417	58,410	481,426
Adapted_LNS(bPFIH,*iINCR*)	410	57,932	467,345
Adapted_LNS(bPFIH,*iDECR*)	410	**57,766**	467,197
Li & Lim's approach	**405**	58,185	**462,873**

Table 2: Overall results of different optimizers on all the 56 modified Solomon's benchmarks

adapted LNS optimizers excelled the original *Swap*(mPFIH) optimizer overwhelmingly on both TV and TD of overall results revealed in Table 2, demonstrating the effectiveness of our proposed improvements over the original heuristics and exploration strategies used. Besides, Li & Lim's metaheuristic approach [7] achieved the best TV and $TV * TD$ results while our Adapted_LNS(bPFIH,*iINCR*) optimiser *obtained the best overall result* in TD, followed by our other Adapted_LNS(bPFIH,*iDECR*) with the 2nd best overall result in TD. Amazingly, these interesting results achieved by our adapted LNS proposals on PDP-TWs completely agreed with the similarly attractive ability of reducing TD demonstrated by the original LNS proposal in solving the conventional VRP-TWs. Furthermore, it is worth considering that the Li & Lim's metaheuristic approach is very complicated to implement with many parameters for tuning. Yet it only excelled our over-

all result in $TV * TD$ slightly by less than 1% less. On the other hand, our adapted LNS proposals is easy to implement or tune for better performance. Lastly, the surprisingly stable performance of both adapted LNS proposals using opposite exploration strategies (*iINCR* versus *iDECR*) is another very attractive strength of our search proposal that prompts for further investigation and a more careful analysis.

5 Concluding Remarks

In this paper, we considered a formal definition of the pickup and delivery problems with time windows (PDP-TWs) [6, 7]. Based on the improved best-fit push forward insertion heuristic (bPFIH) and the adapted LNS proposal, we proposed two systematic and resource-directed exploration strategies that aggressively look for opportunistic improvements so as to solve PDP-TWs more effectively. Our results obtained from the adapted LNS optimizers using the two systematic exploration strategies compared favorably against those of Li & Lim's tabu-embedded metaheuristic search proposal [7] and the previous *Swap*-based optimizer [12] on a set of 56 modified Solomon's test cases. This work, as the first attempt to systematically adapt LNS for solving PDP-TWs, affirmed the attractive capability of the LNS approach to effectively reduce the total distance traveled in solving a set of 56 PDP-TWs. More importantly, our proposal of adapting LNS to solve hard combinatorial problems opens up many possible directions for future investigations.

References

[1] Braysy, O.: Genetic Algorithms for the Vehicle Routing Problem with Time Windows. Special issue on Bioinformatics and Genetic Algorithms, Arpakannus 1/2001.

[2] Glover, F.: Tabu Search - Part I. ORSA Journal on Computing, **Volume 1, Number 3**, Summer, 1989.

[3] Holland, J.: Adaptation in Naturaland Artificial Systems. University of Michigan Press, Ann Arbor, 1975.

[4] Jee, J.: Solving Vehicle Routing Problems with Time Windows using Micro-Genetic Algorithms. Undergraduate Research Opportunity Project (UROP) report, School of Computing, The National University of Singapore, 1999/2000.

[5] Kancko, K., Yoshikawa, M., Nakakuki, Y.: Improving a Heuristic Repair Method for Large-Scale School Timetabling Problems. Principles and Practice of Constraint Programming - CP99, 5th International Conference, Alexandria, Virginia, USA, October 11-14, 1999.

[6] Lau, H., Liang, Z.: Pickup and Delivery Problems with Time Windows: Algorithms and Test Case Generation. in *Proceedings* of the 13th IEEE International Conference on Tools with Artificial Intelligence, Nov 7-9, 2001.

[7] Li, H., Lim, A.: A Metaheuristic for the Pickup and Delivery Problem with Time Windows. in *Proceedings* of the 13th IEEE International Conference on Tools with Artificial Intelligence, Nov 7-9, 2001.

[8] Minton, S., Johnston, M., Philips, A., Laird, P.: Minimizing conflicts: a heuristic repair method for constraint satisfaction and scheduling problems. Artificial Intelligence, 1992.

[9] Prosser, P., Shaw, P.: Study of greedy search with multiple improvement heuristics for vehicle routing problems. Technical Report, **RR/96/201**, Department of Computer Science, University of Strathclyde, Glasgow, January 1997.

[10] Shaw, P.: Using Constraint Programming and Local Search Methods to Solve Vehicle Routing Problems. Technical Report, **RR/98**, Department of Computer Science, University of Strathclyde, Glasgow, April 1998.

[11] Solomon, M.M.: Algorithms for the vehicle routing and scheduling problem with time windows. Operations Research, **35:254-265**, 1987.

[12] Tam, V., Tseng, L.: Effective Heuristics to Solve Pickup and Delivery Problems with Time Windows. in *Proceedings* of the 15th IEEE International Conference on Tools with Artificial Intelligence, Nov 3-5, 2003.

[13] Tsang, E.: Foundations of Constraint Satisfaction. Academic Press, 1993.