An insertion-based heuristic for the constrained pickup and delivery problem

M. Houbraken, T. Walcarius, S. Demeyer, P. Audenaert, D. Colle, M. Pickavet, P. Demeester
Ghent University, Department of Information Technology
maarten.houbraken@intec.ugent.be

In this research, we consider a practical application of the pickup and delivery (P&D) problem incorporating several extensions such as time windows, capacitated non-homogeneous vehicles and on-line request specification. Pickup and delivery problems with each of these extensions are well-studied in literature, resulting in specialised and efficient algorithms. However, combining these techniques is not straightforward as the combined problem setting often prohibits the use of several modifications. In this research, we propose a practical heuristic algorithm for the P&D problem combined with all these extensions.

To show the need for a combined algorithm and test our heuristic, we focused on the practical viewpoint of a delivery company, resulting in the following use case scenario. Consider a set of packets that have to be picked up and delivered within predefined time windows. Each packet has a specified weight along with a pickup and a delivery location. The packets can only be picked up and dropped off during predefined time intervals. To deliver these packets, a set of heterogeneous transport drivers is available. Each transport driver is characterised by a transportation type, a maximum capacity and a base of operations. The transportation type of the transport driver determines the kind of transport (e.g. van driver, truck chauffeur,...) and how the travel times from one pickup/delivery event to another are calculated. The geographical pickup/delivery locations are first mapped, using a KD-tree, on the corresponding nodes in a graph representation of the Belgian road network (1.8 million nodes, 1.9 million links). All distances are then calculated by running Dijkstra’s algorithm from each node to all other nodes.

We started with a simple insertion-based algorithm that iteratively adds new tasks to the schedule found so-far. In each of these iterations, a new task is considered for inclusion in the schedule. This allows the tasks to be specified on-line. When multiple tasks are available for scheduling, the task to be scheduled next is determined based on a compatibility metric. For each combination of a scheduled task with an unscheduled one, the compatibility metric denotes the compatibility of both tasks based on the corresponding time windows and the pickup and delivery locations. This allows for similar tasks to be combined and scheduled simultaneously, speeding up calculation times.

Once the next task to be scheduled is selected, the best driver among all drivers needs to be determined. Per driver, a cost estimate is generated based on the current driver’s schedule. These estimates can be calculated in parallel as
the different driver schedules are independent of each other. To further reduce computation times, a simple heuristic can be used to limit the number of drivers considered.

To efficiently determine whether a task can be included in a schedule, we adopted the approach of [1]. For each event, we maintain quantities that indicate the amount of time the event can be shifted in time. This allows for fast insertion checks. To ensure the capacity constraints are not violated, similar quantities are defined to prevent exceeding the maximum capacity of the driver. Sequencing constraints (pickups before deliveries) are incorporated by maintaining dependency pointers to disallow invalid insertions.

Periodically, the generated schedules are locally optimised by shifting events in the sequence and inverting subtours using the lexicographical search strategy (see [1]). Tasks are also swapped between drivers to avoid suboptimal schedules. This reduces the influence of the initial cost estimates which are based on empty driver schedules. The heuristic algorithm has been implemented and tested on both randomly generated data and data supplied by an industrial company. The test instances varied in size, from 10 to 200 packets, as well as in scale with both city- as countrywide pickup and delivery locations. Experiments have shown that the heuristic produces good and feasible schedules and minimises the total travel time.

In future work, the algorithm will be used as the core of a larger genetic algorithm. The above procedure will generate the initial population after which the crossover and mutation operators will produce the final solution. These operators will reuse the quantities calculated by the heuristic to avoid generation of infeasible solutions. Furthermore, additional optimisation goals will be incorporated such as load balancing and order grouping.

In conclusion, we outlined a heuristic algorithm that produces locally optimised feasible solutions for the on-line capacitated multi-vehicle pickup and delivery problem with time windows. This heuristic was preliminarily tested in realistic test instances obtained from industry and will be further used to develop a genetic algorithm.

**Acknowledgement**  This research is funded by a Ph.D. grant of the Flemish Fund for Scientific Research (FWO – Vlaanderen).

**References**