# Metaheuristics for the Time-Dependent Orienteering Problem

**Cédric Verbeeck** *
*Department of Industrial Management. Ghent University. Belgium*, `Cedric.Verbeeck@ugent.be`

**Pieter Vansteenwegen**
*Department of Industrial Management. Ghent University. Belgium*, `Pieter.Vansteenwegen@ugent.be`

The orienteering problem (OP) integrates the Knapsack Problem (KP) and the Travelling Salesperson Problem (TSP). In contrast to the TSP not all vertices can be visited due to a limited travel time. The OP's goal is to maximize the total score collected by visiting a selection of vertices, while the TSP tries to minimize the travel time or distance. However, determining the shortest path between the selected vertices helps to visit more vertices and might increase the collected score. In addition to this, a feasible solution should start and end at a predetermined vertex.

Orienteering problems are typically used in logistic planning tools where each vertex resembles a customer and the score reflects the profit margin. Furthermore, they serve as the basic problem formulation of personalized touristic trip planners where a vertex stands for a point of interest and the score indicates the personal interest of the tourist in the point of interest.

This research focuses on time-dependent orienteering problems (TDOP) which means that the travel time between two vertices depends on the departure time at the first vertex. This specific problem formulation allows us to tackle congestion related issues in routing problems. To the best of our knowledge, this problem is only discussed by [1], [2], [3], [4]. This research focuses on developing metaheuristics to obtain solutions in real-time. The existing metaheuristics for the time-dependent vehicle routing problem (TD-VRP), a related problem, serve as an interesting starting point.

A first time dependent local search move and local evaluation metric have been successfully implemented. This move iteratively tries to insert non-included vertices into an existing solution, thus improving its total score. To prevent a full and time-consuming evaluation of a solution after every insertion attempt, we store for every included vertex the maximum amount of time that this vertex can be postponed before the solution becomes infeasible. This enables an efficient checking and updating mechanism. Three basic metaheuristic frameworks, implementing this local search move and evaluation technique, were designed as a first test: an artificial immune system (AIS), an iterated local search (ILS) and an ant colony system (ACS). The choice for these basic frameworks was motivated by the fact that generally very complex problems require simple solution mechanics.

The AIS emulates the human immune system and starts by creating initial solutions using a GRASP procedure. Thereafter a certain amount of best solutions are cloned a number of times. During this clone procedure a relatively large amount of mutation (exchange move) is allowed in order to create unique solutions. The new solutions are repaired and improved by the insertion move.

The ACS is based on the behavior of a foraging ant colony, using ethereal pheromones trails to mark travelled arcs (path between two locations). This heuristic starts by creating initial solutions mainly based on greedy information but the pheromone trails are also used. Greedy means relying on the ratio of the score and distance to select the next location. This way of working turns out to be less computational expensive and drastically enhances the performance.

After the creation of an ant, its included arcs are made less attractive for the following construction procedures (local pheromone update) to enhance diversification by depreciating the pheromone trails. Once all ants are created, they are improved by using the insert move and the ant with the highest score is stored. Finally the arcs that are used in the best solution are made more attractive by increasing these pheromone trails (global pheromone update).

The ILS framework starts by creating a random initial solution and improving its score by the insertion move. During a predetermined amount of iterations, a number of consecutive vertices are removed from a certain location within the solution (shake step), followed by a new attempt to insert non-included points. If no improvement can be found, more points are removed. Note that, due to the time dependent nature, a solution can become infeasible when a point is removed as the travel time of the solution increases, therefore additional vertices need to be removed when the shake step turns the solution infeasible.

Developing test instances was done by adopting an extension to the speed model of [5] for TDVRP, which simulates congestion patterns by using road categories and time slots. The travel time is determined by defining a speed level for every combination of a time slot and road category, requiring that every arc (link between two locations) is assigned to a road category. The datasets used for the TDVRP solution methods don't resemble a real live road network as the arcs were randomly assigned to a road category. Therefore we transformed the existing datasets of the orienteering problem to time dependent instances by manually inserting congestion patterns, imitating a real live road network (e.g. realistic morning and evening peaks between city centers and living areas).

Comparing the results between the three solution procedures proves that obtaining solutions in real-time is feasible as the maximum computation time for the largest datasets (102 vertices) is still smaller than 6 seconds and the average runtime is 2 seconds. Currently, the AIS and ACS clearly outperform the ILS framework, especially on larger instances, but require more computation time. The reason might be that, in these test instances, a lot of good solution characteristics are lost due to incremental vertex deletions as an infeasible solution needs to be repaired after a shake step. Another reason might be that too few vertices can be included (since the travel time limit is too constraining) in the developed test instances, significantly hampering the delete and insert procedure of the ILS. Moreover, the average gap between the best solution found and the metaheuristic is marginally lower for the AIS than the ACS (1.6% versus 1.8%). However, ACS performs better on the largest instances.

A preliminary analysis yields insights about the performance of each component, AIS relies heavily on the iterative insert move while the backbone of the ACS is formed by the ant construction procedure together with the pheromone memory effect. Furthermore, both AIS & ACS tend to get stuck in local optima as there is little improvement in performance when the number of iterations is increased.

Based on these preliminary results, adjustments need to be made to design an appropriate metaheuristic that strikes the optimal balance between the quality of the results and the computation time. To enhance the test procedure, instances will be developed so that the optimal solution is known, providing more insight into the obtained results.

## References

[1] Fomin, F. et al. Approximation algorithms for time-dependent orienteering *Information Processing Letters*, 2002.

[2] Abbaspour, A et al. Time-dependent personal tour planning and scheduling in metropolises *Expert Systems with Applications*, 2011.

[3] Garcia, A et al. Integrating public transportation in personalised electronic tourist guides. *Computers and Operations Research* , 2012, accepted.

[4] Li, J. et al. Study on the Time-dependent Orienteering Problem *International Conference on E-Product E-Service and E-Entertainment (ICEEE), 2010.*

[5] Donati, A. et al. Time dependent vehicle routing problem with a multi ant colony system European *European Journal of Operational Research*, 2008.