Available online at www.sciencedirect.com

ScienceDirect

Procedia Social and Behavioral Sciences 2 (2010) 6109–6117

Procedia Social and Behavioral Sciences

The Sixth International Conference on City Logistics

Dynamic optimisation of urban intermodal freight transport with random transit times, flexible tasks and time windows

Alejandro Escudero[a*], Raluca Raicu[b], Jesús Muñuzuri[a], María del Carmen Delgado[a]

[a]*Organization Engineering Group, University of Seville, Camino de los Descubrimientos, s/n, Seville, 41092, Spain*
[b]*Transport Systems Centre, University of South Australia, City East Campus, North Terrace, Adelaide, 5000, Australia*

Abstract

An improvement on drayage operations is necessary for intermodal freight transport to become competitive. When drayage takes place in cities or urban centres transit times are usually random, as a consequence finding the optimal fleet schedule is very difficult, and this schedule can even change during the day. The work we present here is a dynamic optimisation model which uses real-time knowledge of the fleet's position, permanently enabling the planner to reallocate tasks as the problem conditions change. Stochastic trip times are considered, both in the completion of each task and between tasks. Tasks can also be flexible or well-defined. We describe the algorithm in detail for a test problem and then apply it to a set of random drayage problems of different size and characteristics, obtaining significant cost reductions with respect to initial estimates.

Keywords: Urban drayage; intermodality; stochastic transit time

1. Introduction

Road transport has been and continues to be prevalent for the movement of freight on land. However, increasing road congestion and the necessity to find more sustainable means of transport has encouraged governments to promote inter-modality as an alternative. For inter-modality to become viable for trips shorter than 700 km a cost reduction is necessary. Final road trips (drayage) represents 40% of intermodal transport costs. It is possible to overcome this disadvantage and make intermodal transport more competitive through proper planning of the drayage operation.

Traditionally, optimisation efforts focused on drayage operations concentrate on improving the cost and quality of service through the collaboration between drayage companies. Along this line, Morlok and Spasovic (1994) develop an integer programming model to plan truck and container movements in a centralised manner. They contemplate different payment options for drayage services and conclude that centralised management of drayage operations would result in savings between 43% and 63%, as well as an improvements in the quality of service.

* Corresponding author. Tel.: +34-95-448-6042; fax: +34-95-448-7248.
E-mail address: aescudero@esi.us.es.

1877-0428 © 2010 Published by Elsevier Ltd. Open access under CC BY-NC-ND license.
doi:10.1016/j.sbspro.2010.04.023

Following the initial work by De Meulemeester et al (1997) and Bodin et al. (2000), the number of references on centralised drayage management has increased significantly over the last years, but most of them consider the problem only from a static and deterministic perspective. The main objective is normally the assignment of transportation tasks to the different vehicles, often with the presence of time windows (Wang and Regan, 2002). The first part of the work by Cheung and Hang (2003) develops a deterministic model with time windows, which is then solved by means of the making each task's start and end time discrete, and by incorporating the concept of dummy tasks for the beginning and the end of the vehicle's day. Ileri et al. (2006) cover a large number of task types, both simple and combined, as well as the costs involved in drayage operations, and solve the problem with a column generation method. Smilovik (2006) and Francis et al. (2007) incorporate flexible tasks where only the origin or the destination is precisely known.

Much research also includes randomness in the generation of tasks (Bent and Van Hentenryck, 2004; Bertsimas, 1992; Gendreau et al., 1995) or dynamic assignment (Bent and Van Hentenryck, 2004; Psaraftis, 1995; Wang et al., 2007). However, it is hard to find randomness in trip times (Laporte et al., 1992), which is appropriate when the intermodal terminal requiring drayage operations is close to a large urban centre. Cheung and Hang (2003) and Cheung et al. (2005) do consider the dynamic and stochastic characteristics of the drayage problem and solve it with a rolling window heuristic, but this randomness only affects the duration of the task, and not the displacement time between different tasks.

The work we present here considers random trip times both in the completion of each task and between tasks. It also incorporates real-time knowledge of the vehicles' position, which permanently enables the planner to reassign the tasks in case the problem conditions change. Section 3 generalizes the drayage problem as a Multi-Resource Routing Problem with flexible tasks. Section 4 describes the solution methodology for the dynamic and stochastic drayage problem, and section 5 applies this methodology to series of random test cases and summarizes the results. Section 6 summarizes the main conclusions of the work.

2. Drayage Problem Description

Drayage operation can be modelled as a Multi-Resource Routing Problem with Flexible Tasks (MRRP-FT) (Smilowitz, 2006). In a MRRP-FT multiple resources have to be used to complete a series of tasks. The MRRP-FT is defined as follows:

GIVEN: A set of tasks (both well defined and flexible) that require the use of some resources, with certain service times for each resource and time windows; a fleet of each resource type; operating hours at all locations; and a network with stochastic travel times.

FIND: A set of routes for each resource type that satisfies all the tasks while meeting an objective function (minimize operation costs) and observing operating rules for both tasks and resources.

The region where the drayage operations are performed is represented by a graph $G = (N, A)$. The nodes $i \in N$ represent the different facilities of interest for the problem: terminals, depots, loading/unloading points. Each of these nodes is associated a time to attach/detach the container to/from the vehicle, τ_i. Between each pair of nodes $i, j \in N$ there is an arc (i, j) characterized by the transit time τ_{ij}, not known in advance. The transit time has an associated discrete distribution, TT if known.

Every day a series of drayage tasks T must be completed, and failure to do so implies a given subcontracting cost. The drayage tasks can be classified into two groups: well-defined tasks, T_w, and flexible tasks, T_f. Each $t \in T$ has a time window $[a_t^{ini}, b_t^{ini}]$ associated with it. This window limits the time period in which the task has to be completed.

Well-defined tasks represent movements between terminals and customers or vice versa, being both, origin $o_t \in N$ and destination $d_t \in N$ of the movement known. Time windows for well-defined tasks can be flexible, as shown in Figure 1: if the task represents the pickup of a container in the terminal, this task can never start before the train or vessel arrives, on the other hand, if the drayage driver is late then the task can still be completed although it will be penalised. In this last example, a given amount will have to be paid for the time the container remains waiting at the terminal. In a similar manner, if the task represents the delivery of a container to the terminal and it is completed before the allocated time, the container will also be subject to a waiting cost. This cost has been considered proportional to the waiting time.

Flexible tasks represent the movement of empty containers between customers and the depot. Delivery or collection movements of empty containers can take place between a customer and the depot, but also between customers under certain circumstances. For example, from a customer who has requested the collection of an empty container directly to another who has requested the delivery of an empty container, given that their time windows overlap. Therefore, for flexible tasks only the origin or destination is known a-priori, and therefore multiple scenarios, denoted as R_t, are possible. The set of all movements, well-defined tasks and different scenarios generated by possible flexible tasks, is represented by M.

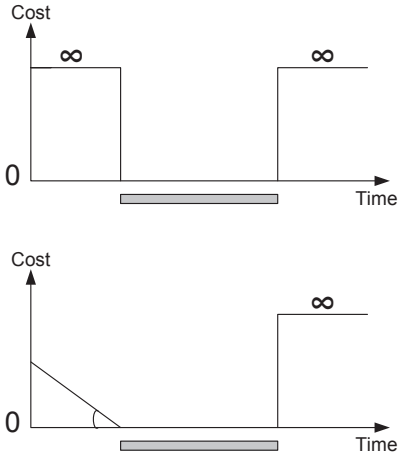

Figure 1 Types of time windows considered for well-defined tasks: hard (above) and flexible (below)

In order to perform all the tasks requested a set of resources is available: containers, vehicles and drivers. Containers are linked to the movement of the tasks with no additional restrictions. Driver-vehicle pairs are considered and represented by V. Each pair is characterised by a location where the working day starts and ends. The different drivers have a time window for the start of their working day $[a_v^{ini}, b_v^{ini}]$ and cannot work longer than MAX_v hours a day. In addition, driver-vehicle pairs have different costs per unit of time depending on whether the vehicle is stopped or moving.

In order to enrich the model with dynamic conditions, a geographic positioning system by satellite (GPS, Galileo, Glonass) is considered in order to provide real time information about the position of each vehicle. This data is used to improve the solution dynamically. Figure 2 shows a scheme of the functioning of the dynamic part of the system considered.

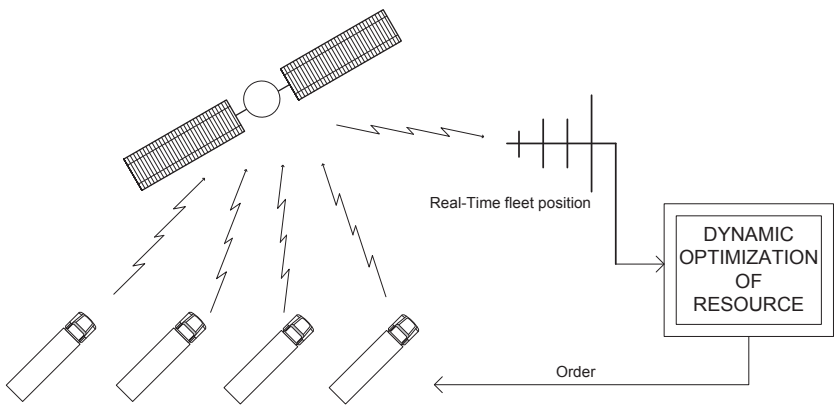

Figure 2 Use of real-time information

3. Genetic Algorithm for Stochastic Transit Time Drayage Problem

The stochastic drayage problem is a NP-hard problem that is extremely difficult to solve analytically. Exact solutions have been found for small problems, but the computation time is high. The stochastic problem appears undoubtedly unsolvable analytically, even more when flexible tasks are incorporated. Furthermore, the use of the real time information about the geographic position of the vehicles requires a high-speed procedure to find the desired schedules. So, an evolutionary algorithm has been used to solve the problem described.

The genetic algorithm used for solving the problem is as follows:

Genetic Algorithm

```
population = InsertionHeuristic (n_task, n_vehicle);
for i=1:max_iter
        population = PopulationGeneration (n_task, n_vehicle, size_population);
        fitness = Evaluation (population);
        parents = Selection(population, fitness, size_selection,'TOP');
        offsprings = GeneticCross (parents, cross_probability);
        offsprings = Mutate s(offsprings, mutate_probability);
        population = population + offsprings ;
        fitness = Evaluation (population);
        dead = Selection (population, fitness, size_selection,'BOTTOM');
        population = population – dead;
end
```

The chromosome which represents each solution is of the same pattern shown as presented in Wang et al. (2007). An example chromosome is shown in Table 1. In this representation, each chromosome is composed of a number of genes, with each gene being a task to complete. So, each task is associated to a fixed gene. A gene is characterized by four features, first being the vehicle to which the task is associated, and is used to identify the order in which each vehicle completes the tasks. The routes represented by the chromosome below would be: Vehicle 1: task1→ task2→ task4; Vehicle 2: task3→ task6→ task5.

Table 1 Chromosome example

1	2	3	4	5	6
1.123	1.673	2.234	1.942	2.440	2.294

The parameters of the genetic algorithm were tested with a sample of problems, and no clear tendency was observed in its performance. The population size was finally set to 60 individuals, 59 of which were initially generated at random and the last one by an insertion heuristic, which also provided the base for comparison of the effectiveness of the algorithm. In every iteration, 2 individuals are selected, taking into account the probability of selection of each individual, as shown in equation (1).

$$PSelection_i^{TOP} = \frac{\left(\min(fitness)/fitness_i\right)^{ElitismFator}}{\sum_j \left(\min(fitness)/fitness_j\right)^{ElitismFator}} \quad (1)$$

These individuals are then allowed to cross and mutate with probabilities of 0.9 and 0.1 respectively. After this, two of the worst individuals are eliminated from the resulting population. Equation (2) shows the mechanism to choose these individuals.

$$PSelection_i^{BOTTOM} = \frac{\left(fitness_i / \min(fitness) \right)^{ElitismFator}}{\sum_j \left(fitness_j / \min(fitness) \right)^{ElitismFator}} \quad (2)$$

The *ElitismFactor* takes a value of 1 at the beginning of the algorithm execution, and it is incremented by 1 every 10 iterations until reaching its highest value, 20. This way of varying the *EliticismFactor* value favours diversified search process at the beginning and intensified at the end (very desirable characteristics for a genetic algorithm applied to a problem as the one presented here).

The repetition of individuals is not allowed in the population. When a new individual that is already in the population appears (repetition case) it is deleted and a new individual is random generated. In addition, when the average fitness of the population is only 10% worse than the best individual of the population, the whole population is randomly regenerated except for that best individual.

An example of the workings of the typical genetic operators of mutation and crossover are shown in Table 2 and Table 3. The crossover operator switches the genes of two parents between two tasks which are selected randomly, tasks 2 and 4 in the following example.

Table 2 Crossover operator example

Task	1	**2**	**3**	**4**	5	6
Parent 1	1.123	**1.673**	**2.234**	**1.942**	2.440	2.294
Parent 2	2.432	**1.721**	**2.325**	**1.987**	1.006	1.396
Child 1	1.123	**1.721**	**2.325**	**1.987**	2.440	2.294
Child 2	2.432	**1.673**	**2.234**	**1.942**	1.006	1.396

The mutation operator randomly selects a gene of the parent individual and changes its first digit to another possible/feasible value.

Table 3 Crossover operator example

Task	1	2	**3**	4	5	6
Parent	1.123	1.673	**2.234**	1.942	2.440	2.294
Child	1.123	1.673	**1.234**	1.942	2.440	2.294

The fitness of each individual represents the total costs associated with the resulting routes. The costs contemplated for each route includes four factors:

- Fixed cost per vehicle,
- Distance cost,
- Waiting cost of containers at the terminals due to early arrival or late collection, and
- Cost of task loss, assimilated to the subcontracting cost of that task to an external company.

However, trip times are stochastic, so the fitness needs to be calculated as an estimate of the expected costs. An iterative algorithm was developed to complete that estimation, calculating the probability of reaching the next link of the route at a given time and the resulting costs involved. The different phenomenon that may occur during a task execution has to be considered by the iterative algorithm. If the arrival time of a vehicle at the beginning of a given task is prior to the opening of its time window, this vehicle will have to wait until the start of the time window, or else incur cost in a proportional the amount of waiting time. On the other hand, if the arrival is after the time window has closed, there is a higher penalty due to the waiting cost at the terminal or to the possible task loss

(because of the departure of the train or vessel). It can also happen that two tasks on the same route are both flexible and complementary, they will be combined and completed at the same time, thus avoiding a return to the depot.

4. Dynamic Methodology

As previously explained, the dynamic aspects of the tasks completion are incorporated in the model and the problem is solved through real time vehicle localization. So, the proposed algorithm is applied to the problem as follows: the genetic algorithm is run at the beginning of the day, and the best solution is obtained given the cost estimations that are available at that time. After this, every time a vehicle completes a task, the algorithm is run again considering the updated data only for the remaining, pending tasks. The real-time location of the vehicles is considered, and cost and time estimations are run for each iteration for the vehicles that are yet to complete a task. Figure 3 shows a schematic view of the dynamic methodology used.

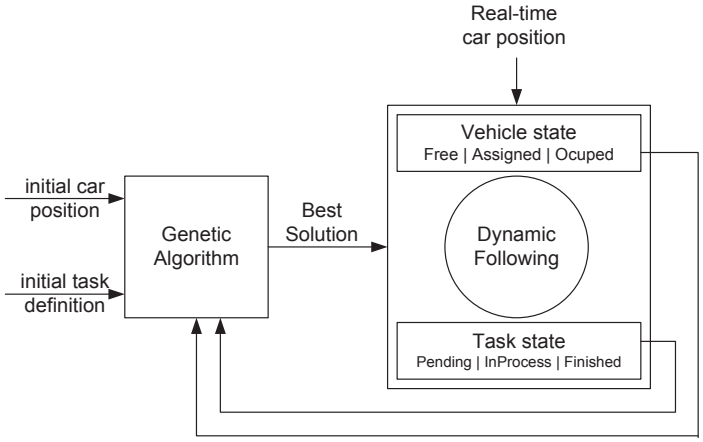

Figure 3 Dynamic methodology

5. Test Problem and Result

The genetic algorithm was tested using a problem generator that randomly builds problems of a specified size. The inputs to the generator were the number of available vehicles in the fleet and the number of flexible and well-defined tasks.

The generator randomly distributes the customers, the intermodal terminal and the depot in a 100x100 area. The well-defined tasks consist, with equal probability, either of a pickup or delivery of containers at the terminal. Flexible tasks imply either the collection or delivery of empty containers at the customers, also with equal probability.

Time windows for well-defined tasks range from 30 minutes to 4 hours with a uniform stochastic distribution, and their start time is fixed randomly during the day. Time windows for flexible tasks will be open from the beginning of the day until a specified time for empty container deliveries, and from a specified time until the end of the day for empty container collections. Those specified times are also generated randomly with a uniform distribution.

To simplify calculations, the time horizon is divided into 5 minute intervals. Finally, to simulate in real time the position of each vehicle, a uniformly distributed speed between 45 and 55 km/h is calculated for each 5-minute period.

In order to test the performance of the algorithm for problems of different size and characteristics, we have built a set of random drayage problems using the problem generator (see Table 4). For each random problem, we have determined the improvement of the genetic algorithm with regards to the insertion heuristic in the first iteration (see

Table 5, column 2), the average improvement of the estimated cost for the best solution in iteration *i+1* with respect to the simulated cost for iteration *i* (column 4), and the estimated cost reduction between the first and last iteration of the genetic algorithm (column 5).

Table 4 Description of the problem set

Problem code	Task number	No. of well-defined tasks	No. of flexible tasks	Fleet size
L1	20	0	20	5
L2		5	15	5
L3		10	10	5
L4		15	5	5
L5		20	0	5
M1	30	0	30	7
M2		10	20	7
M3		15	15	7
M4		20	10	7
M5		30	0	7
H1	40	0	40	9
H2		10	30	9
H3		20	20	9
H4		30	10	9
H5		40	0	9

Table 5 Results obtained for the random problem set

Problem code	Genetic vs. Insertion Improvement (%)	No of iterations	Average improvement of the GA (%)	Dynamic Improvement (%)
L1	5,9109	6	0,23981	27,229
L2	20,015	10	0,98411	39,084
L3	0	8	2,6657	11,052
L4	0,3848	10	0,73763	29,83
L5	5,0968	12	2,864	34,36
M1	27,01	8	0,5923	49,326
M2	7,7103	10	2,3859	30,608
M3	15,857	13	5,0287	36,844
M4	5,507	12	2,6161	23,785
M5	6,3091	15	6,6728	25,95
H1	6,3557	12	0,76608	35,271
H2	1,1787	10	3,5312	10,38
H3	16,126	12	2,9754	42,62
H4	2,6864	13	2,3971	6,8958
H5	10,128	18	2,4007	43,871

To clarify the results, the M3 test problem is explained with more detail (See Table 6). For each iteration, the cost estimations correspond to the real costs involved in the vehicles operations up to that moment plus the estimated costs remaining until the end of the day, as previously explained. We have also simulated the best solution provided by the genetic algorithm for each iteration until the end of the day, leaving aside the dynamic data, in order to find out to what extent the dynamic data helps to continuously improve the solution as the day proceeds. If the genetic algorithm finds a task allocation that is better than the one simulated in the previous iteration, the estimated cost for the genetic algorithm is lower than the simulated cost obtained before. Otherwise, both costs are identical or at least very close.

Table 6 Cost results for each iteration of the test problem M3

	Estimated cost for the insertion heuristic	Estimated cost for the genetic algorithm	Genetic Improvement (%)	Simulated cost
Iteration 1	1993,3	1677,3	15,8569	1941
Iteration 2		1539,5	20,6852	1501,5
Iteration 3		1427,5	4,9284	1507,3
Iteration 4		1448,3	3,9142	1759,8
Iteration 5		1461,3	16.9626	1439,3
Iteration 6		1439,3	0	1576,8
Iteration 7		1403,3	11,0036	1403,5
Iteration 8		1394,5	0,6413	1301,5
Iteration 9		1298,5	0,2305	1521
Iteration 10		1499	1,4464	1496,9
Iteration 11		1496,9	0	1316,4
Iteration 12		1309,4	0,5317	1258,9
Iteration 13		1258,9	0	1263,9

The overall improvement achieved by the genetic algorithm can be estimated by comparing the estimated cost in the last iteration, 1258.9, with the estimated cost for the initial insertion heuristic, 1993.3, resulting in a 36.84% cost reduction.

These results were obtained running the genetic algorithm for a fixed amount of time (2 min) for each iteration.

6. Conclusion

In this paper we have shown the importance of obtaining real-time data of vehicle locations in a drayage fleet through the use of a satellite positioning system. This knowledge, together with an optimisation algorithm based on metaheuristics, enables real-time management of the fleet in a changing environment, which reduces operation costs by as much as 30%. These results are especially valuable for intermodal operations in congested metropolitan areas, where travel times are stochastic due to congestion. Besides this, we have modelled the problem as a MRRP with flexible tasks, allowing both, intermodal drayage operations and the repositioning of empty containers, to be optimized at the same time.

To solve the drayage problem, we have developed a real-time optimisation model based on a genetic algorithm that operates with stochastic cost estimations, and that has been tested with a series of drayage problems generated randomly. The genetic algorithm improves the initial solution, provided by an insertion heuristic, with an average improvement of around 5.57% for each dynamic iteration for the type of problems considered.

Acknowledgement

This research was supported from Era-Star Regions Program and regional government of Andalusia, grant SR-0197/2008 (Project Galileo-Drayage).

References

Bent, R. W., & Van Hentenryck, P. (2004). Scenario-based planning for partially dynamic vehicle routing with stochastic customers. *Operations Research*, 52(6), 977-987.

Bertsimas, D. J. (1992). A vehicle routing problem with stochastic demand. *Operations Research*, 40(3), 574-585.

Bodin, L., Mingozzi, A., Baldacci, R., & Ball, M. (2000). The rollon–rolloff vehicle routing problem. *Transportation Science*, 34(3), 271-288.

Cheung, R. K., & Hang, D. D. (2003). A time-window sliding procedure for driver-task assignment with random service times. *IIE Transactions*, 35(5), 433-444.

Cheung, R. K., Hang, D. D., & Shi, N. (2005). A labeling method for dynamic driver-task assignment with uncertain task durations. *Operations Research Letters*, 33(4), 411-420.

De Meulemeester, L., Laporte, G., Louveaux, F. V., & Semet, F. (1997). Optimal sequencing of skip collections and deliveries. *Journal of the Operational Research Society*, 48(1), 57-64.

Francis, P., Zhang, G., & Smilowitz, K. (2007). Improved modeling and solution methods for the multi-resource routing problem. *European Journal of Operational Research*, 180(3), 1045-1059.

Gendreau, M., Laporte, G., & Seguin, R. (1995). An exact algorithm for the vehicle routing problem with stochastic demands and customers. *Transportation Science*, 29(2), 143-155.

Gendreau, M., Hertz, A., & Laporte, G. (1994). A tabu search heuristic for the vehicle routing problem. *Management Science*, 40, 1276-1290.

Ileri, Y., Bazaraa, M., Gifford, T., Nemhauser, G., Sokol, J., & Wikum, E. (2006). An optimisation approach for planning daily drayage operations. *Central European Journal of Operations Research*, 14(2), 141-156.

Laporte, G., Louveaux, F., & Mercure, H. (1992). The vehicle routing problem with stochastic travel times. *Transportation Science*, 26(3), 161-170.

Morlok, E., & Spasovic, L. (1994). Redesigning rail-truck intermodal drayage operations for enhanced service and cost performance. *Journal of the Transportation Research Forum*, 34(1), 16-31.

Psaraftis, H. N. (1995). Dynamic vehicle routing: Status and prospects. *Annals of Operations Research*, 61(1), 143-164.

Smilowitz, K. (2006). Multi-resource routing with flexible tasks: an application in drayage operations. *IIE Transactions*, 38(7), 577-590.

Wang, J. Q., Tong, X. N., & Li, Z. M. (2007). An improved evolutionary algorithm for dynamic vehicle routing problem with time windows. In *Computational science* – ICCS 2007 (pp. 1147-1154). Springer Berlin / Heidelberg.

Wang, X., & Regan, A. C. (2002). Local truckload pickup and delivery with hard time window constraints. *Transportation Research Part B*, 36(2), 97-112.