International Joint Conference - ICIEOM-ADINGOR-IISE-AIM-ASEM (IJC 2017)
Valencia, Spain, July 6-7, 2017

Using simulated annealing to solve the daily drayage problem with hard time windows

Escudero-Santana, A[1]; Cuberos-Gallardo, M; Muñuzuri, J; Cortés, P.

Abstract Drayage is the stage of the intermodal transport that deals with transport of freight on trucks among the intermodal terminal, and customers and suppliers that are located in its hinterland. This work proposes an algorithm based on simulated annealing heuristics to solve the operations of drayage. This algorithm has been used to solve battery problems, demonstrating the validity and suitability of its results, which were compared with exact method.

Keywords: Drayage; intermodality; simulated annealing; hard time windows.

1 Introduction

Between the different ways of carrying out the transport of goods, the highest growth is on the intermodal transport, which is a particular way of multimodality. It consists in using different modes of transport to carry a unit load. This feature allows to reduce the time the goods must spend in intermodal terminals, due to the replacement of the operations of loading and unloading of goods from a container to another by loading and unloading of containers between different means of transport.

This paper focuses on a problem of land transport, the Daily Drayage Problem (DDP). This issue provides the connection between the intermodal terminal and its hinterland, including the operations of loading and unloading containers in vehicles (Jula et al., 2005). The DDP aims to optimize the tasks of container shipments from the terminal to customers and suppliers, and viceversa. This type of problem is a specific vehicle routing problem with time windows (VRPTW), where the ca-

[1]Alejandro Escudero Santana (e-mail: alejandroescudero@us.es)
Dpto. de Organización Industrial y Gestión de Empresas II. Escuela Técnica Superior de Ingeniería. Universidad de Sevilla. Avd. Descubrimientos S/N, 41092 Seville (Spain)

pacity of the vehicle is defined as a binary variable, so vehicles must be uploaded or downloaded.

The DDP has been solved with different methods. Caris and Janssen (2009) propose a local search heuristic, Caris and Janssen (2010) develop a simulated annealing, Smilowitz (2006) prove a roll-on horizon method, Zang et at. (2009) and Xue et al (2015) use a tabu search, and presents a viral method to solve the problem.

This work aims to solve the daily drayage problem by means of an algorithm based on the heuristic of simulated annealing. Section 2 gives an approach of the daily drayage problem. The state of art of simulated annealing is presented in section 3. The results obtained are shown in section 4. Finally, several conclusions are drawn in section 5.

2 Problem description

Drayage is the beginning and the end of the intermodal transport chain. It is composed of the operations of collection and delivery of empty containers in the deposits, collecting the goods, loading and unloading in intermodal terminals, and delivery charge to the recipient. At this stage, it is considered that the trucks could be in three different states: loaded with a full container, loaded with an empty container, or without charge.

The two fundamental operations of the transport are the pick-up and delivery of containers. These operations always occur in a particular facility (client, depot or terminal). However, the movement of a container may be carrying out more operations. The need to have containers available at different points at certain times makes it essential the transport of empty containers. Also, depending on the property of the containers (containers may belong to the charger, the recipient or the company responsible for the main journey), there will be more or less empty return journeys. These movements must be reduced to the maximum with the objective of maximizing the transport load factor.

These tasks must be performed within some time windows if a good performance of the intermodal chain is required. This peculiarity makes the problem to solve to be defined as a DDPTW, where there are several tasks, *T*, which should be covered with a combination of vehicles, *V*, within certain time windows. The problem is to assign each task to a vehicle so that the generated costs are minimized.

For the formulation of the problem, several simplifications have been conducted to facilitate the modelling:

- Vehicles are similar in terms of capacity and technical characteristics
- Fixed costs are the same for all vehicles.

- Variable costs are proportional to the distance travelled.
- There is only a deposit, which is located within the intermodal terminal.
- Only importation and exportation tasks are carried out.

3 The simulated annealing algorithm

The simulated annealing (SA) algorithm is an iterative method for solving combinatorial and optimization problems (Kirkpatrick, 1984). The main attraction of the algorithm is its convergent behaviour to the optimum of the problem. This is achieved because the algorithm allows to accept worse quality solutions in each interaction, allowing the search not to be complete when a local optimum is found.

The algorithm is based on the process of annealing of metals and ceramics for crystallized materials, with minimum internal energy. In this process the temperature of the material rises to crystallize, thus increasing its internal power, and subsequently allowing to cool slowly. In this process of the loss of energy, the particles that make up the material move in search of positions of lower energy, thus achieving the crystallized form.

There have been some authors who have used the Simulated Annealing algorithm to find near-optimal solutions in various problems of combinatorial optimization (Chiang and Russell, 1996; Eglese, 1990).

3.1 Solution encoding

The form of the solution has been considered a vector that includes numbers from 1 to $n + m$, where n is the number of tasks that should be covered and m the number of available vehicles. In the solution, each number from 1 to n corresponds to a task, while the numbers from $n+1$ to $n+m$ correspond to vehicles. The interpretation of the solution is the following: each vehicle must cover all the tasks until another number corresponding to another vehicle is found. This is illustrated with an example in Table 1, where there are 3 vehicles that must cover 6 tasks.

Table 1 Examples of interpreting a matrix solution

Matrix solution	Interpretation
[7 1 2 8 3 4 5 9 6]	The vehicle 7 covers tasks 1 and 2. The vehicle 8 covers tasks 3, 4 and 5. The vehicle 9 covers task 6
[7 1 2 3 4 8 9 5 6]	The vehicle 7 covers tasks 1, 2, 3 and 4. The vehicle 8 is free. The vehicle 9 covers tasks 5 and 6

3.2 Algorithm

The algorithm developed to solve de DDPTW follows the next steps:

1. Firstly, an initial vector solution is randomly created, S_0. Best solution, $S_{best}= S_0$
2. That solution is evaluated using the objective function, $f(S_0)$.
3. The neighbourhood of solution is generated, $N(S_0)$. Searching of neighbour solutions is implemented by swapping all the numbers of the current solution. Each new permutation between two numbers (either vehicles or tasks) generates a new solution.
4. A solution from the neighbourhood is randomly chosen, S' *in* $N(S_0)$, and its cost is calculated by evaluating it in the objective function $f(S')$.
5. This new solution is adopted as a solution with a probability P, depending on the improvement of the new solution with respect to the current solution, $\delta=f(S')-f(S_0)$.
   - $P = 1$ if $\delta<0$
   - $P = e^{-\delta/t}$ if $\delta > 0$
6. If $\delta<0$, then , $S_{best}= S'$
7. The process (3-5) is repeated L times for each value of t.
8. t is updated with the cooling function, $\alpha(t)$.
9. The process is repeated until the stop condition, t_{min}.

Parameters of the SA are:

- Temperature, t: temperature is a control parameter that determines the probability of accepting a new solution as one whose cost is greater than the current solution, allowing that way to escape from local minima.
- Cooling function, $\alpha(t)$: determines the speed with which the temperature decreases.
- Initial temperature, t_0: determines the value of temperature when the process starts.
- Final temperature, t_{min}: marks the algorithm stop condition.
- Length of the string, L: sets the number of iterations which the algorithm makes for each value of the temperature.

The value of parameters should be set based on the experience. In the present work, the value of L depends on the size of the problem, to allow for more thorough searches on larger problems. The values of the defined parameters are shown in Table 2. These values were selected by mean different experiments. These experiments were done on the Solomon Benchmark.

Table 2 Parameters of the algorithm

Parameter	Value
t_0	50
t_{min}	1
$\alpha(t)$	$t_{k+1}=t_k/(1+0.005 \cdot t_k)$
L	10

4 Results

The solved problems are part of a battery of problems generated in Escudero et al. (2015), which is an adaptation of the Benchmarking created in Solomon (1987) for the VRPTW. It is a battery of 12 Random problems (R), 9 Cluster problems (C) and 8 Random + Cluster problems (RC).

The problems have been solved by considering hard time windows where the delay in the solution are not allowed, so the total cost generated by a solution can be broken down into two distinct costs:

- Cost per distance: is the cost that is directly dependent on the total distance travelled. It has a value of 1.
- Cost per vehicle: fixed cost associated with the use of each vehicle. This cost has a value of 10 per vehicle, and will be 0 when the car is idle.

Further details of the problem are the average speed of vehicles, 60 km/h, and the time of loading and unloading, estimated both in 15 minutes.

Results are shown in Table 3, Table 4 and Table 5. Every instance was runned 10 times. It is clear that the execution times increase as the size of the problem increases. However, for problems of the same size, these times are very similar, and then the computational load only depends on the size of the problem to solve, and no other variants as the random initial solution.

In Table 6 a comparison between SA results and the results of exact method (Gurobi) is shown. Gurobi use different methods as heuristics, branch and bound and cutting planes. It is important to consider that Gurobi present a tolerance of $10e^{-4}$ on the constraints. Despite this, gap between SA and Gurobi is low. On execution time, Gurobi was tested with test R3. Result of Gurobi for 25 tasks is 872,63 with an execution time of 105,29 seconds; for 50 tasks, the best result reaches by Gurobi after one hour is 1784,31 (SA present a gap of 0,01%), and for

100 tasks the best result provides by Gurobi in one hour is 3928,68 (result of SA is 3258,15).

Table 3 Solutions of problems with 25 tasks

Test	Class R			Class C			Class RC		
	Ave. Cost	Min. Cost	Exe. Time	Ave. Cost	Min. Cost	Exe. Time	Ave. Cost	Min. Cost	Exe. Time
1	1082,40	1061,60	17,17	712,48	700,31	15,24	1483,64	1477,84	23,52
2	1028,84	1024,80	22,91	674,24	669,11	23,16	1196,23	1184,52	28,01
3	878,63	872,63	29,39	687,90	681,83	28,19	1152,57	1149,75	32,38
4	870,85	858,57	32,60	685,83	672,05	30,85	1123,01	1117,93	33,52
5	1047,00	1034,16	21,70	700,51	678,97	18,54	1400,87	1398,12	25,25
6	1000,27	994,00	25,04	740,17	735,03	16,68	1391,96	1379,84	30,96
7	971,65	968,02	29,44	683,71	673,07	19,51	1288,03	1283,02	33,56
8	936,16	929,04	31,32	656,04	649,04	32,51	1221,58	1178,61	34,58
9	955,85	949,89	26,72	680,36	669,08	26,17			
10	909,60	901,92	32,51						
11	943,31	933,33	30,60						
12	933,02	928,28	29,97						

Table 4 Solutions of problems with 50 tasks.

Test	Class R			Class C			Class RC		
	Ave. Cost	Min. Cost	Exe. Time	Ave. Cost	Min. Cost	Exe. Time	Ave. Cost	Min. Cost	Exe. Time
1	2180,61	2067,84	55,60	1512,98	1488,72	45,92	3035,65	2961,35	71,22
2	1967,81	1953,58	76,98	1481,07	1459,34	69,06	2635,28	2607,01	79,72
3	1814,64	1803,51	89,71	1464,92	1450,70	82,46	2428,31	2408,62	92,74
4	1751,74	1743,88	97,05	1490,76	1481,47	90,10	2335,36	2328,00	99,38
5	1980,13	1962,87	72,10	1499,15	1476,95	51,86	3112,31	3084,84	77,13
6	1875,62	1861,08	82,57	1553,13	1523,48	56,13	3036,86	2987,29	85,23
7	1811,35	1802,20	90,14	1496,73	1482,13	58,32	2613,20	2568,27	99,70
8	1736,29	1724,52	97,41	1424,04	1408,66	93,19	2540,98	2439,32	100,27
9	1860,88	949,89	86,23	680,36	669,08	78,58			
10	1822,91	901,92	99,29						
11	1789,42	933,33	99,00						
12	1772,77	928,28	99,85						

Table 5 Solutions of problems with 100 tasks.

Test	Class R			Class C			Class RC		
	Ave. Cost	Min. Cost	Exe. Time	Ave. Cost	Min. Cost	Exe. Time	Ave. Cost	Min. Cost	Exe. Time
1	3923,88	3858,50	206,63	3648,29	3587,18	183,88	4825,37	4729,63	251,85
2	3642,99	3584,55	270,91	3559,44	3491,33	260,82	4356,98	4293,25	312,33
3	3285,65	3258,15	338,09	3467,24	3424,08	323,41	4088,93	4043,49	343,73
4	3165,90	3144,81	384,41	3427,53	3401,51	361,84	4037,85	3998,86	368,07
5	3599,15	3544,08	256,46	3662,17	3582,56	198,37	4849,72	4740,79	302,30
6	3462,41	3424,64	315,75	3527,42	3471,54	283,27	4718,19	4543,67	331,38
7	3259,01	3236,06	339,87	3572,76	3505,77	235,40	4283,26	4234,76	371,39
8	3145,32	3121,36	370,67	3339,09	3327,02	374,37	4145,33	4105,18	389,17
9	3412,79	3356,38	319,47	3464,77	3431,89	296,95			
10	3307,31	3245,70	359,75						
11	3401,02	3373,58	354,43						
12	3203,80	3167,17	379,95						

Table 6 SA algorithm vs. Exact method (Class R – 25 tasks)

	SA	Gurobi	Gap
R25-1	1061,60	1061,60	0,00%
R25-2	1024,80	1009,66	1,48%
R25-3	872,63	862,63	1,15%
R25-4	858,57	848,10	1,22%
R25-5	1034,16	1023,36	1,04%
R25-6	994,00	985,41	0,86%
R25-7	968,02	965,33	0,28%
R25-8	929,04	919,03	1,08%
R25-9	949,89	939,88	1,05%
R25-10	901,92	891,92	1,11%
R25-11	933,33	931,72	0,17%
R25-12	928,28	919,95	0,90%

5 Conclusion

Simulated Annealing is a widely used meta-heuristic in the resolution of optimisation problems. Several research papers have been able to verify its usefulness. A great advantage is its proved convergence to optimum. The convergence is achieved through a process of slow cooling. It involves high execution times, and thus, computational cost of the algorithm that can not be assumed. So, an adequate setting of the parameters is extremely important, mainly both the cooling function and the length of the spring, so that an equilibrium between computational cost and goodness of the solution could be accomplished.

Related to the DDPTW, it is necessary to analyse the suitability of the algorithm with regard to this kind of problems. Analysing the reached results, it is concluded the algorithm achieves satisfactory results in comparison with the classical heuristic, in spite of its greater computational cost. Therefore, the application of SA to solve the DDPTW is appropriate for solving problems without an excessive level of response time, such as the planning of daily routes. Nevertheless, it is not suitable to solve real-time problems.

An important concern about the usefulness of SA in the field of drayage is its convergence. So, this method could be used as a comparison tool for future development.

6 References

Caris A, Janssens GK (2009) *A local search heuristic for the pre- and end-haulage of intermodal container terminals.* Computers & Operations Research 36(10): 2763-2772.

Caris A, Janssens GK (2010) *A deterministic annealing algorithm for the pre- and end-haulage of intermodal container terminals*. International Journal of Computer Aided Engineering and Technology 2(4): 340-355.

Chiang WC, Russell RA (1996) *Simulated annealing metaheuristics for the vehicle routing problem with time windows*. Annals of Operations Research 63(1): 3-27.

Eglese RW (1990) *Simulated Annealing: a Tool for Operational Research.* European Journal of Operational Research 46: 271-281.

Escudero-Santana A, Munuzuri J, Cortes P, Aparicio P (2015*) A viral system to optimise the daily drayage problem*. International Journal of Bio-Inspired Computation 3(7): 176-182.

Escudero-Santana A, Munuzuri J, Guadix J, Arango C (2013) *Dynamic approach to solve the daily drayage problem with transit time uncertainty*. Computers in Industry 62(2): 165-175.

Jula H, Dessouky MM, Ioannou P, Chassiakos A (2005) *Container movement by trucks in metropolitan networks: modeling and optimization*. Transportation Research Part E 41(3): 235-259.

Kirkpatrick S, Gelatt CD, Vecchi MP (1983) *Optimization by simulated annealing*. Science 220(4598): 671-680.

Smilowitz K (2006) *Multi-resource routing with flexible tasks: An application in drayage operations*. IIE Transactions Institute of Industrial Engineers 38 (7): 570-590

Solomon M (1987) *Algorithms for the vehicle routing and scheduling problems with time window constraints*. Operations Research 35(2): 254-265.

Xue Z, Zhang C, Lin WH, Miao L, Yang P (2014) *A tabu search heuristic for the local container drayage problem under a new operation mode*. Transportation Research Part E 62: 136-150.

Zhang R, Yun WY, Moon I (2009) *A reactive tabu search algorithm for the multi-depot container truck transportation problem*. Transportation Research Part E 45(6): 904-914.