MULTIOBJECTIVE EVOLUTIONARY ALGORITHM FOR INTEGRATED TIMETABLE OPTIMIZATION WITH VEHICLE SCHEDULING ASPECTS

Michal Weiszer[1], Gabriel Fedorko[2], Zdenek Čujan[3]

Summary:This paper describes the implementation of evolutionary multiobjective genetic algorithm (NSGA-II) to integrate timetabling and vehicle scheduling stages of the transportation planning proces. Model with timetable optimization focused on minimizing the transfer time of the passengers in transfer node along with minimizing the number of vehicles needed to operate such timetable is presented. Results from simple test case illustrate the effectiveness of a such approach. Developed solution is able to optimize conflicting objectives of passengers and transportation company simultaneously.

Key words: multiobjective optimization, genetic algorithm, timetable, vehicle scheduling

1. INTRODUCTION

The planning process of public transport service usually consists of 4 major parts: network design (routing), timetabling, vehicle scheduling, crew scheduling and rostering.

Due to the size and complexity of the whole process, these planning stages are typically carried out in a sequence, where previous step serves as an input to the next step. However, since each planning step has influence on the other, simultaneous approach is desirable, that the public transport planning problem could be treated with respect to global optimality. As the most of real world problems, also transport planning is multiobjective problem with different and often conflicting objectives. On the one side, there is the passengers view (transport service quality mainly determined by network design and timetabling) and on the other side is the view of the transport company (mainly financial criteria involved in vehicle and crew scheduling). Advances in transportation research in the recent years along with information technology make integration of some planning stages possible. Some effort in this direction has been done, examples are the Periodic Event Scheduling problem for railway with partial integration of other planning aspects in [1] and an approach using iterated local search in [2].

The objective of this paper is not only to propose integrated planning approach but also to incorporate a multiobjective evolutionary search into model. Rest of this paper is organized as follows: in next section, basic theory on multiobjective optimization and dedicated genetic

[1] Ing. Michal Weiszer, The Technical University of Košice, Faculty of Mining, Ecology, Process Control and Geotechnology, Logistics Institute of Industry and Transport, Park Komenského 14, Košice, tel.:+4210556023126, e-mail: michal.weiszer@tuke.sk

[2] doc. Ing. Gabriel Fedorko, PhD., The Technical University of Košice, Faculty of Mining, Ecology, Process Control and Geotechnology, Logistics Institute of Industry and Transport, Park Komenského 14, Košice, tel.:+4210556023143, e-mail: gabriel.fedorko@tuke.sk

[3] doc. Ing. Zdeněk Čujan, CSc., College of logistics, Department of Logistics and Technical Disciplines , Palackého 1381/25, 750 02 Přerov, e-mail: zdenek.cujan@vslg.cz

algorithm NSGA-II is presented. Thereafter, a simple test case illustrates the implementation of the multiobjective algorithm. Last section examines the numerical results of the test case.

2. EVOLUTIONARY ALGORITHMS AND MULTIOBJECTIVE OPTIMIZATION

2.1 Introduction to evolutionary multiobjective optimization

Evolutionary algorithms (EA) are broad category of optimization algorithms (Genetic algorithm and differential evolution for example). EA use a population based search in which new population of solutions is evolved in each generation. EA are popular optimization technique used in wide range of applications. Traditional evolutionary algorithms are single-objective in which the fittest individual (with highest objective function value) represents the single suboptimal solution. The fact that EA works with multiple solutions at time makes it very suitable for multiobjective optimization.

Let assume number of K objectives, which have to be minimized. Given a solution $x \in R^n$, it is a vector of n decision variables: $x = \{x_1, x_2, \ldots, x_n\}$ in the solution space **X**. The objective is to find a vector x^* that minimizes a given set of K objective functions $f(x^*) = \{f_1(x^*), \ldots, f_K(x^*)\}$. The solution space **X** is generally restricted by a series of constraints and bounds on the decision variables. Since optimization of x with respect to just one objective can often lead to worse results in other objectives, simultaneous approach is needed and single solution to multiobjective problem often doesn't exist at all. Therefore the aim of multiobjective optimization is to find a set of solutions each of which minimizes the K objective functions at an acceptable level and it is not dominated by any solution.

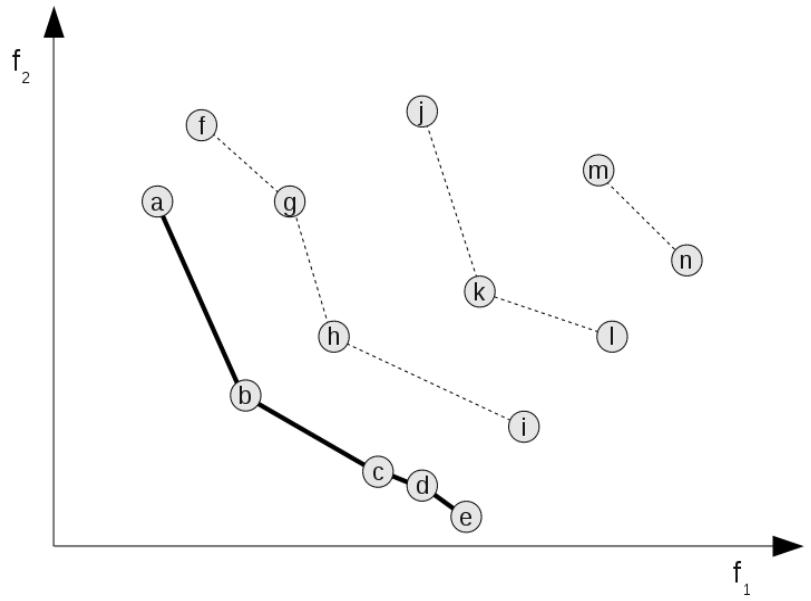
A solution x_1 is said to dominate the other solution x_2, if:

1. The solution x_1 is no worse than x_2 in all objectives. Thus, the solutions are compared based on their objective function values, and

2.The solution x_1 is strictly better than x_2 in at least one objective.

Solution, that is not dominated by any other solution in solution space is called Pareto-optimal. A Pareto-optimal solution can not be improved in any objective without worsening in at least one other objective. The all non-dominated solutions in solution space **X** constitute the Pareto optimal set. The corresponding objective functions values of the Pareto-optimal set in the objective space form a Pareto front. Fig. 1 illustrates the Pareto front in objective space for two objective functions f_1, f_2. The Pareto front is highlighted. For example, in Fig. 1 solution i is dominated by solutions c, d and e. Solutions f, g and h are dominated by only a single solution (a, b, b).

Several evolutionary algorithms for multiobjective optimization have been proposed. Review of existing algorithms can be found in [3] and [4]. One of them is well tested and computationally efficient Fast Non-dominated Sorting Genetic Algorithm NSGA-II by Deb [5].



Source: Authors

Fig. 1 – Pareto front in objective space

2.2 Fast Non-dominated Sorting Genetic Algorithm (NSGA-II)

NSGA-II uses dominance ranking and crowding distance evaluation for an individual instead of objective function value. Crowding distance promotes search of solutions uniformly spread along the Pareto front. The crowding distance is used in following manner (Fig. 2):

Step 1: The population is ranked by dominance rule and non-dominated fronts F_1, F_2,...,F_R are identified. For each front F_j, $j = 1, \ldots, R$ repeat Steps 2 and 3.

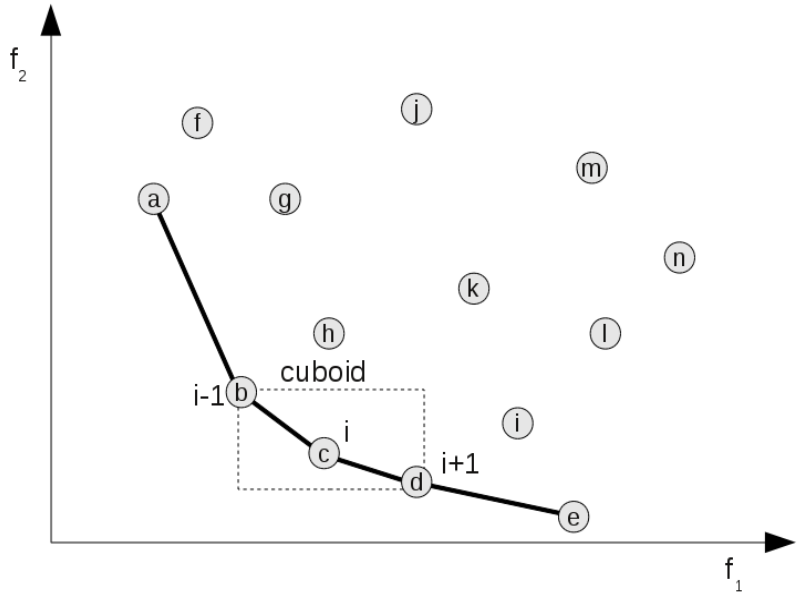
Step 2: The solutions in front F_j are sorted in ascending order. The sorting is repeated for each objective function f. Let $l = |F_j|$ and $x_{i,k}$ represent the i-th solution in the sorted list with respect to the objective function f_k.. Assign $cd_k(x_{1,k}) = \infty$ and $cd_k(x_{l,k}) = \infty$, and then assign for $i = 2, \ldots, l$-1 :

$$cd_k(x_{i,k}) = \frac{f_k(x_{i+1,k}) - f_k(x_{i-1,k})}{f_k^{\max} - f_k^{\min}}$$

Where $f_k^{\max}, f_k^{\min}$ are the maximum resp. minimum value of objective function f_k so far.

Step 3: To compute the total crowding distance $cd(x)$ of a solution x, the solution's crowding distances with respect to each objective are summed, $cd(x) = \sum_k cd_k(x)$.

This crowding distance measure is used in crowded tournament selection operator. That is, between two solutions with differing nondomination ranks, we prefer the solution with the lower (better) rank. Otherwise, if both solutions belong to the same front, then we prefer the solution that has higher crowding distance. A solution with a higher value of this distance measure is less crowded by other solutions [5]. The complete algorithm is described in detail in [5].



Source: [4]

Fig. 2 – Crowding distance calculation

3. TEST CASE

There are 4 lines given (Fig. 3), with transfer node *T*. Some routes run in parallel between specific nodes. Each line consists of pair of routes, one route in forward direction and one in opposite direction. Headways for lines are given as follows: 12 minutes for line 1 and line 2, 10 minutes for line 3 and line 4. The objective is to determine timetable τ for node *T* for each route within the time period of 60 minutes that transfer time of passengers and number of vehicles needed to operate the timetable are minimized. Timetable τ consists of timetables for 8 routes (4 lines * 2), where each timetable for route *i* consists of *n* arrivals in node *T* within the time period. The term h_i is the headway for route *i*:

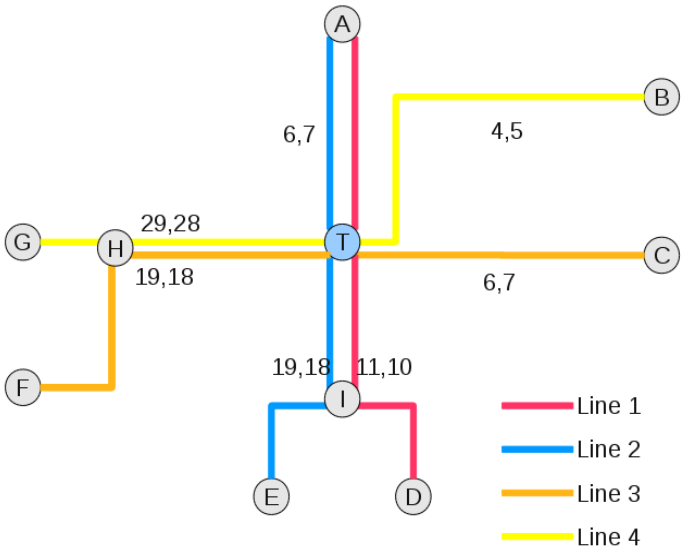
$$\tau = \{T_1, T_2, ..., T_i\},\ i = 8$$
$$T_i = \{t_{i1}, t_{i2}, ..., t_{in}\}\ n = 60/h_i,\ i = 1,..,8$$

Hence timetables are cyclic with constant headways, the timetable for route *i* can be determined from first arrival, using integer multiples of headway. This way the number of decision variables can be downsized significantly.

$$T_i = \{t_{i1}, t_{i1}+hi, t_{i1}+2h_i ..., t_{i1}+(n\text{-}1)h_i\}\ ,\ t_{i1} < h_i$$

The representation of a solution for the EA (the individual) has 4 genes representing the arrivals t_{11}, t_{21}, t_{51}, t_{61}. Due to the overlaping routes, these are synchronized and the arrivals for the rest of the lines are determined as follows:

$$t_j = (t_i + \frac{h_i}{2}) \bmod h_i \qquad i=1,2,5,6\ ,\ j=3,4,7,8$$



Source: Authors

Fig. 3 – Test case

3.1 Transfer time of passengers

First objective function f_1 that gives the tranfer time of passengers can be constructed as follows:

$$\text{Minimize } f_1 = \sum_i \sum_j \sum_k \sum_l \alpha_{ij} (t_{ik} - t_{jl}) c_{kl}^{ij}$$

Subject to:

$$(t_{ik} - t_{jl}) \geq 1 \qquad \forall i, j, k, l, \alpha_{ij} = 1 \qquad (1)$$

$$\alpha_{ij} \in \langle 0,1 \rangle \qquad \forall i, j \qquad (2)$$

$$t_{ik}, t_{jl} \in Z^+ \qquad \forall i, j, k, l \qquad (3)$$

$$t_{ik}, t_{jl} \in \langle 0, 60 \rangle \qquad \forall i, j, k, l \qquad (4)$$

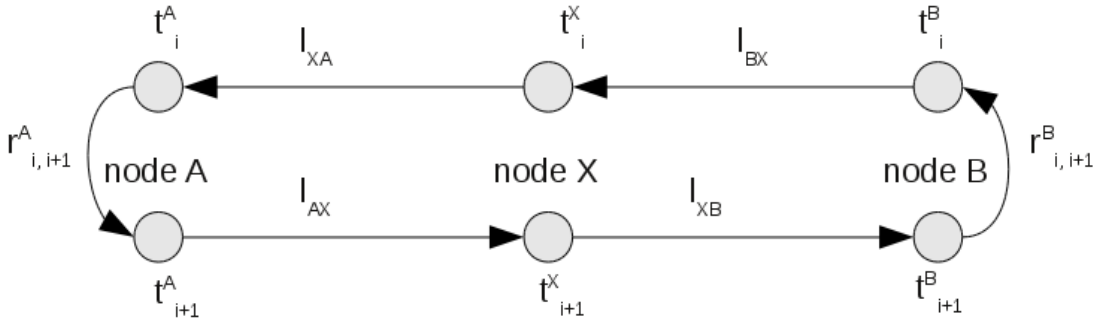
Where the transfer time is a difference between the arrival of k-th bus/tram of i-th route and the arrival of l-th bus/tram of j-th route, multiplied by number of transfering passengers c_{kl}^{ij} and summed over all buses/trams of all routes . The term α_{ij} can be either 0 or 1 according to the possibility of transfer between routes i and j. Constraint (1) ensures that transfer time is greater or equal than 1 minute, constraint (2) states that α_{ij} can take binary values and constraint (3) ensures that the arrivals have integer values. The constraint (4) ensures that the arrivals are within the time period of 60 minutes.

3.2 Number of vehicles

On a single traffic line (Fig. 4) between terminal A and terminal B (with route i and i+1 for opposite direction), if we assume that no interlining for vehicles is allowed, minimum number of vehicles N to serve this line is exactly:

$$N = \frac{l_{XA} + l_{BX} + l_{AX} + l_{XB} + r^{A}_{i,i+1} + r^{B}_{i,i+1}}{h_i}$$

Where l_{XA}, l_{AX}, l_{BX}, l_{XB} is travel time between corresponding nodes, $r^{A}_{i,i+1}$, $r^{B}_{i,i+1}$ is the difference between the arrival t^{A}_{i}, resp. t^{B}_{i+1} and departure t^{A}_{i+1}, resp. t^{B}_{i} from terminal. t^{X}_{i}, t^{X}_{i+1} is the arrival in node *X*.



Source: Authors

Fig. 4 – Single traffic line

Due to the periodic property of the timetable, the arrival t^{A}_{i} and departure t^{A}_{i+1} can be determined:

$$t^{A}_{i} = (t^{X}_{i} + l_{XA}) \bmod h_i$$
$$t^{A}_{i+1} = (t^{X}_{i+1} - l_{AX}) \bmod h_i$$

Note that modulo operator gives always positive remainder. Next, difference $r^{A}_{i,i+1}$ equals:

$$r^{A}_{i,i+1} = (t^{A}_{i+1} - t^{A}_{i}) \bmod h_i = (t^{X}_{i+1} - t^{X}_{i} - l_{XA} - l_{AX}) \bmod h_i$$

The difference is a sum of minimum turnover time of the vehicle and the idle waiting time. When altering the timetable (the arrivals) only this idle waiting times are affected. The corresponding travel times between tranfer node and terminals with minimum turn over times included for presented test case are depicted in (Fig. 3).

Finally, the second objective function f_2 is the summary of N_L vehicles for all L = 1,..,4 lines:

$$\text{minimize } f_2 = N_1+N_2+N_3+N_4.$$

4. RESULTS

Assume that majority of transferring passengers travel between node *A* and *H*, therefore only transfer between lines 1,2 to lines 3,4 is desirable and only transfer time between these lines is measured. Library with NSGA-II implementation in Python language [6] was used. The parameters of algorithm are following, population size is 100 and maximum generations are 80. The experiment was carried out on Athlon Dual Core processor X2 4600+ with 2GB of memory and it took 22 seconds. Within the specified number of generations, solutions in Tab 1. were found.

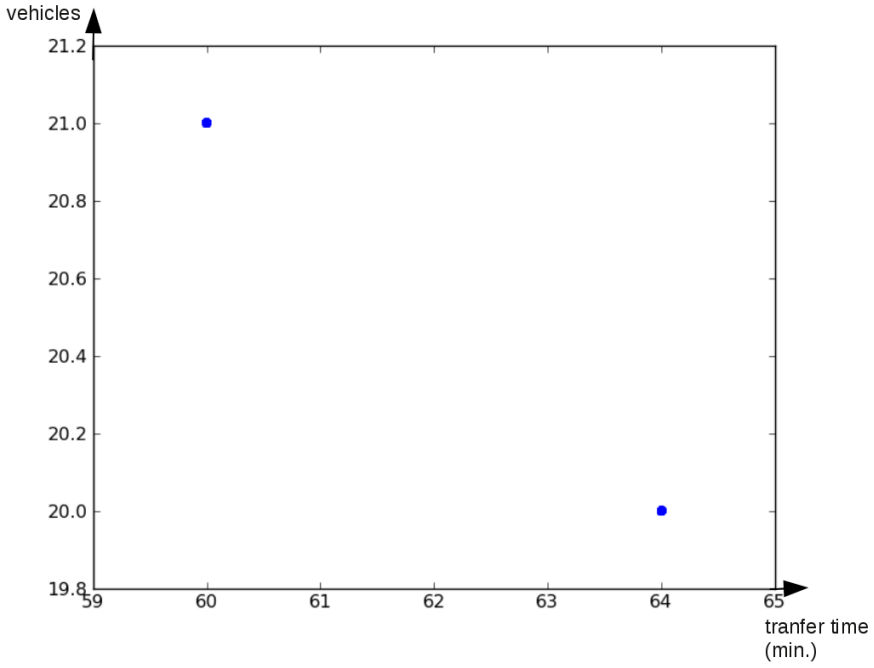
Tab. 1 – Solutions found with NSGA-II

Solution $\{t_{11}, t_{21}, t_{51}, t_{61}\}$	f_1 (min.)	f_2 (vehicles)
7, 6, 5, 4	60	21
8, 6, 6, 4	60	21
8, 7, 5, 2	64	20
7, 6, 5, 2	64	20
5, 4, 6, 3	64	20
4, 3, 5, 2	64	20
8, 7, 6, 3	64	20
3, 2, 5, 2	64	20
4, 3, 6, 3	64	20
9, 8, 6, 3	64	20

Source: Authors

Found solutions can be classified into two groups with equivalent solutions according to objective functions values. In one group solutions achieved minimum transfer time $f_1 = 60$ minutes. In the other group solutions have worse transfer time but achieved minimum number of vehicles $f_2 = 20$. In this case, the Pareto-optimal set consists of only two solutions which form the Pareto front (Fig. 5).

For illustration the best solutions found using traditional genetic algorithm (GA) with single objective function are in Tab. 2. In the first row, solution of GA with f_1 objective function after specified number of generations is presented. Solution of GA with f_2 objective function is in the second row. It is clear that solutions are optimal only in respect to single objective.



Source: Authors

Fig. 5 – Pareto front for test case

Tab. 2 - Solutions found with GA

Solution $\{t_{11}, t_{21}, t_{51}, t_{61}\}$	f_1 (min.)	f_2 (vehicles)
10,6,8,9	**60**	22
11,8,1,8	66	**20**

Source: Authors

5. CONCLUSION

The public transport planning process is complex and difficult and it is processed usually in a sequence. However, research in the recent years have made integration of some stages possible. In this paper, integration of timetable optimization with vehicle scheduling aspects is presented. In addition, the optimization is carried out in multiobjective and evolutionary manner. This way the conflicting objectives of passengers and transportation company can be optimized simultaneously. Therefore the decision maker can investigate various scenarios and make a trade-off.

6. ACKNOWLEDGEMENTS

This work was supported by the Slovak Grant Agency for Science VEGA 1/0864/10 and VEGA 1/0864/10 Design of the Model of Integrated Transport System with Implementation Green Logistic and VEGA 1/0095/10 The research of conditions affecting degradation and lifetime decrease of conveyer belts for pipe conveyors with usage of progressive mathematical and simulation methods for reliability growth and APVV Project SK-SRB-0034-09.

REFERENCES

[1] LIEBCHEN, C., MÖHRING, R., H. The Modeling Power of the Periodic Event Scheduling Problem : Railway Timetables — and Beyond. In *Algorithmic Methods for Railway Optimization* : Lecture Notes in Computer Science, 2007, vol. 4359, p. 3-40. ISSN 0302-9743.

[2] GUIHAIRE, V., HAO J., K. Transit network re-timetabling and vehicle scheduling. In *Communications in Computer and Information Science (CCIS)*. Vol. 14, Berlin: Springer, 2008, p. 135-144, ISBN 978-3-540-87477-5.

[3] KONAK, A., COIT, D., W., SMITH, A., E., Multiobjective optimization using genetic algorithms: A tutorial, *Reliability Engineering & System Safety*, Special Issue - Genetic Algorithms and Reliability, September 2006, Vol. 91, Issue 9, Pages 992-1007, ISSN 0951-8320.

[4] DEB, K. Introduction to Evolutionary Multiobjective Optimization. In *Multiobjective Optimization* : Lecture Notes in Computer Science, 2008, vol. 5252, p. 59-96. ISSN 0302-9743.

[5] DEB, K., PRATAP, A., AGARWAL, S., MEYARIVAN, T., A fast and elitist multiobjective genetic algorithm: NSGA-II, *IEEE Transactions on Evolutionary Computation*, Apr 2002, vol.6, no.2, pp.182-197, ISSN 1089-778X.

[6] GARRETT, A. ECsPy: Evolutionary Computations in Python, [program]. Ver. 0.7. Jacksonville (USA): Jacksonville State University, 2010, [cited 2010-11-6] Available online: <http://code.google.com/p/ecspy/>, Library for Python.

[7] FEDORKO, G., WEISZER, M., Optimalizácia cestovných poriadkov s využitím genetického algoritmu, *Perner's Contacts*, 2009, Vol. 4, no. 1 (2009), p. 92-100.