Proceedings of the 2012 Winter Simulation Conference
C. Laroque, J. Himmelspach, R. Pasupathy, O. Rose, and A.M. Uhrmacher, eds

AN INTEGRATED SIMULATION MODEL AND EVOLUTIONARY ALGORITHM FOR TRAIN TIMETABLING PROBLEM WITH CONSIDERING TRAIN STOPS FOR PRAYING

Erfan Hasannayebi

Sharif University of Technology
PO Box 11365-11155
Tehran, IRAN

Soheil Mardani

Simaron Pardaz Co.
PO Box 15959-54511
Tehran, IRAN

Arman Sajedinejad

Tarbiat Modares University
PO Box 14115-143
Tehran, IRAN

S. Ahmad Reza Mir Mohammadi K.

Delft University of Technology
PO Box 5015, NL-2600GA
Delft, The Netherlands

ABSTRACT

This paper presents a simulation-based optimization approach for railway timetabling, which is made interesting by the need for trains to stop periodically to allow passengers to pray. The developed framework is based on integration of a simulation model and an evolutionary path re-linking algorithm with the capability of scheduling trains, subject to the capacity constraints in order to minimize the total waiting times. A customized deadlock avoidance method has been developed which is based on a conditional capacity allocation. The proposed look-ahead deadlock avoidance approach is effective and easy to implement in the simulation model. A case study of the Iranian Railway (RAI) is selected for examining the efficiency of the meta-heuristic algorithm. The result shows that proposed algorithm has the capability of generating good quality solution in real-world problems.

1 INTRODUCTION

Railway network timetabling refers to scheduling of trains and determining the departure and arrival of them in a railway network. Offline train scheduling problem in a large scale railway network is still a difficult optimization problem of the railway transportation systems. This paper represents a specific application of simulation methodology and optimization in Iranian railway context, linking the fields of computer simulation and artificial intelligence. A primary attraction of the paper is its emphasis on both methodology and applications. A variety of interesting aspects relating to timetabling in the context of the Iranian railway network are presented. These aspects include how to plan stops for regular prayer services, conflict detection procedures, deadlock avoidance strategies, and application of evolutionary path re-linking for the train timetabling problem.

For comprehensive surveying of railway optimization models, see Cacchiani and Toth (2012). Simulation approaches for the train scheduling problem provide a model to generate a feasible timetable or to validate a given timetable. The most proposed simulation modeling applications in context of railway systems are mainly limited to the timetable evaluation. For example, an advance simulation software SIMONE (Middelkoop and Bouwman, 2001) is developed in the Netherlands for railway capacity measurement and timetable robustness evaluation. Marinov and Viegas (2010) presented a simulation modeling methodology using an event-based simulation computer package called SIMUL8 for analyzing freight

train operations. Computer simulation is also an important and useful tool for automatic generation of timetables and simulation of train's operations. Petersen and Taylor (1982) developed a simulation model for train scheduling problem with a deadlock blockage avoidance algorithm in a railway line. Dessouky and Leachman (1995) developed a simulation modeling approach for single-track or double-track rail networks. A hybrid methodology of the event-driven and network-based simulation methods is proposed in Cheng (1998) for resolving resource conflicts in train rescheduling problem. Geske (2006) presented a constraint based deterministic simulation model to minimize the tardiness of trains. Li et al. (2008) presented an advanced simulation method based on the global information for solving the train scheduling problem to reduce total traveling time on a single-track network. Yalçınkaya and Bayhan (2012) developed a feasible timetable generator framework and also a stochastic simulation model in a railway line, but no attempt was made to generate a near optimal timetable. Some limited efforts have been made on using a hybrid simulation modeling and optimization procedure to generate near optimal timetable. Cui (2010) proposed the framework of a hybrid model for a partially-automatic dispatching of railway operations. While simulation modeling approach has been used in the literature, there is a lack of powerful simulation based optimization approach for train timetabling problem. This has been the major motivation of this research. There are different meta-heuristic algorithms which have not been applied on railway scheduling problem till now. Nonetheless, there is no application of path re-linking algorithm for train scheduling problem in the literature. This study proposed an evolutionary path re-linking algorithm for train timetabling problem.

2 SIMULATION BASED OPTIMIZATION APPROACH

Simulation models can be integrated with optimization engines to solve optimization problems. Simulation and optimization seem to be outstanding approaches that are extensively used to help decision making process (Hillier and Lieberman, 2009). In the simulation-based optimization method, optimization usually searches the solution space to find best quality solutions which are evaluated by a simulation model. Some attempt has been made to produce general purpose optimization engines. Commercial optimization packages include OptQuest®, Extend Optimizer® and Witness Optimizer® utilize different search methods. In most cases, developing customized optimization algorithm is more suitable for complex optimization problems. Until now, there is not any recognized detail of the way simulation and optimization techniques should be integrated with each other. In order to ensure the timetable feasibility, the solution must be tested within a detailed simulation model. In the proposed integrated approach, the meta-heuristic optimizer sets the values of decision variables and uses the responses to make new decisions for the next iteration. The simulation model also modifies the departure times, subject to minimum headway time which guarantees the achieving of best quality solutions. The output timetable can also be validated by human experts. The cooperative framework has some benefits such as the fact that simulation module can identify deadlock and conflict situations and resolving them to obtain feasible timetable.

3 PROBLEM DEFINITION

A railway network is composed of block sections, stations and links separated by traffic signals. A block section is a track segment between two adjacent stations and only one train can occupy it at a time. The traveling of a train through a particular block section is termed a train operation. It is assumed that all trains have a pre-specified traveling route. Furthermore, the free running times of trains at segments are assumed to be constant. The religious obligations enforce trains to stop in the praying time windows for performing praying services. The stopping schedule consists of location of stop for each praying interval. Trains stop for a time which is enough for passengers to get off the train and pray in a mosque. The stop plan for performing praying services depends on the departure and also arrival time to the destination. The two conditions of prayer service activation can be expressed as the following equations:

$$\begin{cases} d_{r(1,j),j} \leq L_{p,r(1,j)} & (1) \\ c_{r(N_j,j),j} \geq U_{p,r(N_j,j)} & (2) \end{cases}$$

Equations (1) and (2) describe the conditions that enforce trains to be held up at particular stations during certain fixed hours of the day. Here, p specifies the index of the praying intervals (e.g. p=1,2,3 indicate the morning, afternoon and night praying intervals respectively). The symbol $r(k,j)$ also specifies the index of kth station in the traveling route of train j. We denote $d_{r(k,j),j}$ and $c_{r(k,j),j}$ as departure time of train j from kth station and arrival time of train j at kth station of its route. The total number of stations visited in the route of train j is also N_j. Therefore $d_{r(1,j),j}$ and $c_{r(N_j,j),j}$ are departure time of train j from origin and arrival time of train j to destination respectively. The pth praying interval in station i denoted by $[L_{p,i}, U_{p,i}]$. In the simulation model, the actual arrival time of trains to stations are unknown in advance, therefore trains check the conditions of prayer service activation during the passing stations. If the conditions (1) and (2) are satisfied, the concerned train can stop to perform praying service.

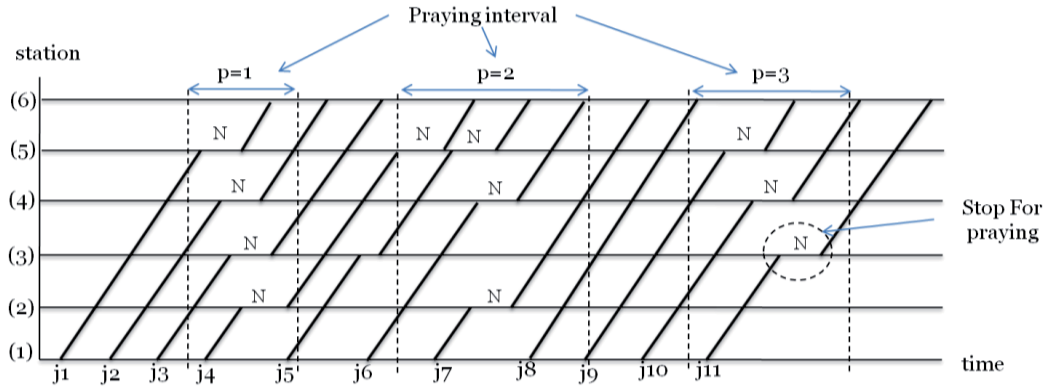

Fig 1- Stopping schedule during the praying intervals in a time-space graph

Railway planners use time-space or time-station diagram as the main tool for constructing and establishing a timetable. An example of a time-space graph that represents the stopping schedule in praying intervals is illustrated in Fig 1. The dash lines represent the praying time intervals for every prayer service in each station. The symbol *N* shows the station selected for performing praying services.

4 DISCRETE EVENT SIMULATION OF RAILWAY OPERATIONS

Discrete event simulation is extensively used for modeling the behavior of a complex system within a discrete time framework usually through a sequence of events. All the state variables of the system are discrete variables and the number of states is finite. For example, track occupancy's status belongs to {empty, occupied}. Every event which occurs during the time window $[t_i, t_i+1]$ is executed at the end of the time window, t_i+1. When all these events have been computed, the clock jumps to the time point of the next scheduled instant, t_i+2. An event signifies the happening of a change in the status of system at a specific time. In event driven simulation model, an event list is used to store system events in the order of the time of occurrence. The event list updates at the end of each iteration. This process continues to execute the next event until either a terminating condition of the simulation model is satisfied or the event list becomes empty.

5 SIMULATION MODELING IN ENTERPRISE DYNAMICS (ED)

Enterprise Dynamics simulation software has been chosen as a simulation platform because of its powerful simulation engine, flexibility to design customized objects and ability to implement optimization algorithms. Enterprise Dynamics from Incontrol Simulations Solutions is a leading object oriented simulation platform to design and implement simulation solutions. The users can pick up objects (called atoms) from standard libraries in order to build their own models. Advanced users can build and use their own atoms, in many goals such as modeling a block section with very specific characteristics. ED also has a built-in

programming language called 4DScript. The list of events in the atom editor window defines the behaviors of an atom. During a simulation run, an atom responds to certain events that happen. The classification of events has been introduced as a useful feature in ED. The main event list for every atom consists of on-event, on-entered, on-exited and on-reset. An event can be created for atom e_1 that will occur e_2 seconds from the moment of creation, with event code e_3. When the event occurs, the atom's "on-event" event handler is executed. The variable "event-code" specifies which event is executed. It can be used to execute different commands within one "on-event" event handler, depending on the event code value. The On-Entering event handler defines the behavior of an atom when it enters into another atom. But, the On-Entered event handler is executed when another atom enters to an atom. Atoms and particularly objects in simulation model can communicate with each other by means of input, output and central channels. Input and output channels are also used to control movement of atoms and generally flow control in the simulation model. In the proposed simulation model, an input channel describe an inflow. The routing of trains through the network is performed by the output channels.

6 TRAINS OPERATIONS MODELING

The simulation model simulates train's operations on the railway network to generate timetable. Trains reserve a free line of successor station when exiting from current station. A train can dispatch from current station if successor block section is considered available and there is at least one free line for passing train or one free platform for stopping train in the next station. A block section is available if it contains no train. The dispatching of a train to this successor block section also should not create a potential deadlock situation. The deadlock situation will be discussed in section (7). When a train enters a waiting queue, a procedure is applied to check the dispatching conditions. If all conditions are satisfied, the involved train moves to the successor block section. Otherwise, a circular event will be created to execute dispatch checking algorithm at t=t+1 and the involved train should wait in current station until all conditions become satisfied.

7 PROPOSED SYSTEM ARCHITECTURE

The proposed system consists of four main parts, a database, a simulation engine that generates a timetable, an optimization module and a graphical user interface. The outline of proposed system architecture is presented in Fig 2.

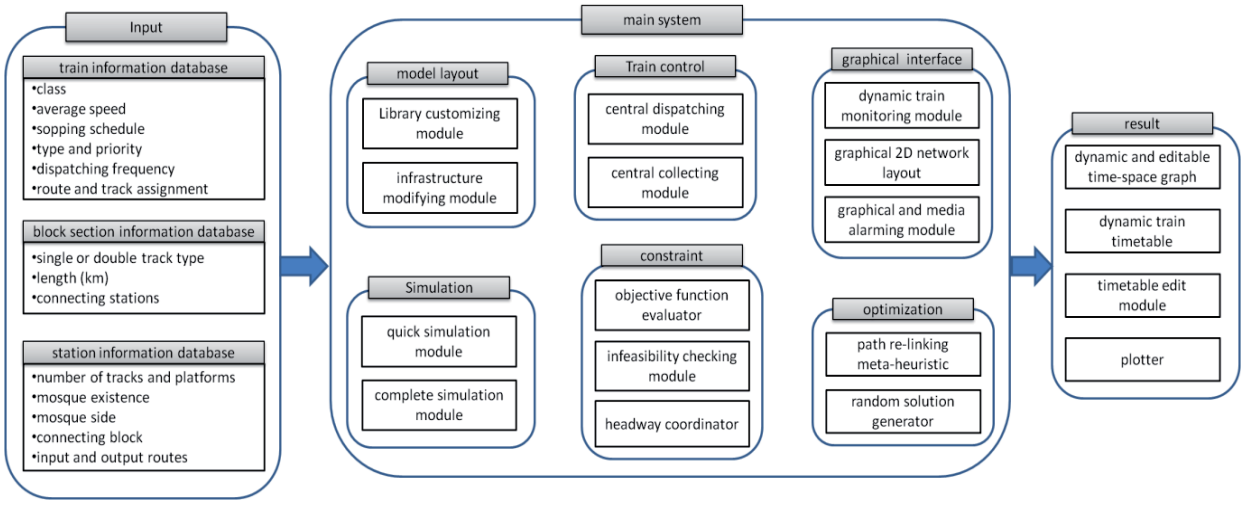

Fig 2- System architecture outline

Users can define the routes, track configuration and various characteristics of the stations. Users also can construct a simulation model of a railway network by drag and drop the required objects into the 2D layout and then connect the channels. The proposed decision support system has train movement simulation capability, which shows trains traversing routes in detailed graphical time-space graph. The system has a very fast simulation engine that produces a complete timetable. In brief, it includes quick simulation module, complete simulation module, optimization module, timetable modifying module, graphical timetable output and report customizing module. Users need to insert the details of the railway infrastructure network and train types into the database. This railway infrastructure network information includes the set of stations, connection points, railway lines and station characteristics.

By simulating the train movements and logging the information during the whole simulation process, the information of the given departure plan can be obtained and evaluated. In order to find a relative near-optimal timetable in a certain time period, several runs of simulation may be executed with different input parameters. The integration of the simulation model and optimization is practical for large scale optimization problems. The proposed approach is well known and powerful methodology and it can be used to model a general type of railway infrastructure.

8 OBJECTS AND EVENT LIST

The proposed framework is constructed by defining a set of railway objects, with their interrelationships, in an object-oriented simulation environment. The atoms (objects) represent the major components of a railway system. The railway operation rules are implemented by the conditions and coding in the event list of the objects. In the developed simulation model, a customized library of atoms is designed which can be used to construct various railway infrastructure configuration. The entities of the simulation model are trains, stations, block sections, waiting queues, headway justifiers, central dispatchers and central collector. Some of the main simulation model entities and their connections are shown in Fig 3.

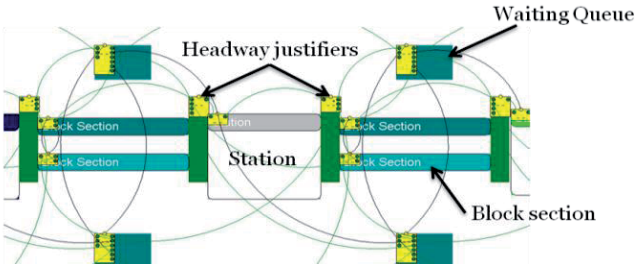

Fig 3- Main entities in 2D model layout

Railway signaling technology in IRAN's railway network has been based mainly on the so called "Conventional Fixed Block System" principle. In this traditional signaling system which is based on fixed blocks, the railway is divided into track sections, which are separated by signals. A train is not allowed to enter a track section or block section before the preceding train has cleared it. Signaling system ensure that trains do not enter tracks those are occupied or may become occupied by other trains. This can be ensured by proper settings of dispatching rules in the infrastructure elements (entities) in the simulation model. A waiting queue is used to hold a waiting train in the station. A headway justifier coordinates the required headway between trains. Central dispatching object is used for circular event generating and designing the sequence of dispatches from origin station. A central collector object is also used for collecting trains and also evaluating the objective function at the end of each iteration. A general station object can be built with desirable output and input channel configuration. That means a general network configuration such as N-track railway network can also be modeled in the proposed simulation framework. The description of attributes of the entities are summarized in Table 1.

Table 1- main entities and their attributes

Entity	Status variables and attributes	Type of value
Station (server)	1- Number of free tracks without platform	Integer
	2- Number of free tracks with platform	Integer
	3- Number of output lines (channels)	Integer
	4- Number of input lines (channels)	Integer
	5- Mosque existence	Binary
	6- Praying activation type	Binary
	7- Praying intervals for every praying services	Integer interval
Block (server)	1- Occupancy	Binary
	2- Single or double track	Binary
	3- Availability according to maintenance period	Binary
Train (product)	1- Origin station	Integer
	2- Destination station	Integer
	3- Total delay	Integer
	4- Station passed counter	Integer
	5- Praying service status	{0,1,2,3}

9 CONFLICT DETECTION AND RESOLUTION IN SIMULATION MODEL

In the simulation model, trains run with different frequency of departure. Some trains departure daily but some departure in specific days of a week (such as Wednesdays and Fridays). To construct a daily timetable, all the departure information should be considered in the simulation run. To avoid a time consuming seven-day simulation run, the train's conflicts should be resolved according to departure days. In short, trains departure daily in the three-day simulation model, but the track and station capacity allocation is handled by existence of trains in specific day. For precise detecting of conflicts in block sections, a representation theme of departure days is introduced. In this approach, an abstract representation of all weekly information of a timetable is used instead of simulating the system for a week. This abstract representation is called dispatch code. Dispatch code of each train (DC_j) is a seven digit binary array. The element *i* of the dispatch code array (a_{ij}) represents the existence of train *j* in day *i* of the week. For example if train j_1 departure daily and train j_2 departure only on Mondays and Fridays then dispatch code of these trains are DC_{j1} =[1111111] and DC_{j2}=[0010001] respectively. For accurate conflict detection, it is necessary to update the dispatch code of trains periodically. The cycle time of the update procedure is T=24h. Update procedure changes the dispatch code by shifting digits one step. For example, the new dispatch code of for train j_2 is DC_{j2}=[1001000] after execution of update procedure (Fig 4).

Fig 4-Updating the dispatch code

When more than one train want to enter the same block section, they have to check the dispatch code of existing trains in the block section. The involved train is permitted to enter to a block section if there is not any conflict with respect to the departure days. In Fig 5, two examples are represented for the conflict detection procedure.

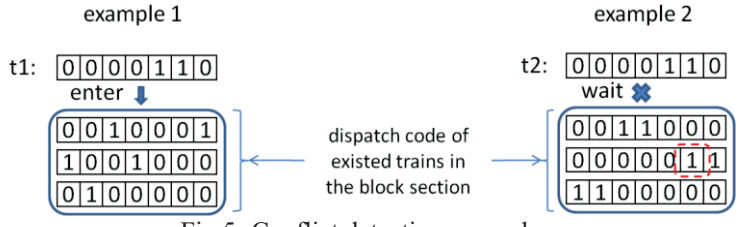

Fig 5- Conflict detection examples

Consider two train services, t_1 and t_2 (Fig 5). In example 1, train t_1 can enter the block section because the existing trains in the block section do not have any departure day in common with train t_1. But in example 2, train t_2 should wait because dispatch codes of existing trains in the block section have conflict with dispatch code of train t_2. Note that the discussion above makes no assumption on the direction of the trains. A simple equation can be used for conflict detection. Let the train j the involved train and the set of trains existed in the current block section denoted by I. If $\sum_{k\epsilon I}\sum_{i=1}^{7} a_{ik}.a_{ij} > 0$ then the involved train is not permitted to enter the block section. For precise execution of capacity reservation procedures, a similar representation can be utilized. Instead of specifying the capacity by only a value, a two dimensional integer array is utilized. The capacity of stations is divided into number of track lines with platform and number of track lines without platform. This modified representation is compatible with dispatch code of trains and defines the capacity allocated in each day of week. Hence, the capacity of station can be controlled and updated according to dispatch code of entering and exiting trains. For example, if a station has three tracks with platform and one track without platform, the capacity code can be represented in Fig 6. An example of capacity allocation is also showed in Fig 6. In this example, the first entering train (t_1) which needs platform reduces the capacity of station according to its dispatch code. The next entering train (t_1) which does not need platform also reduce the capacity of station according to its dispatch code. These trains update the capacity of station in the same manner on exiting event.

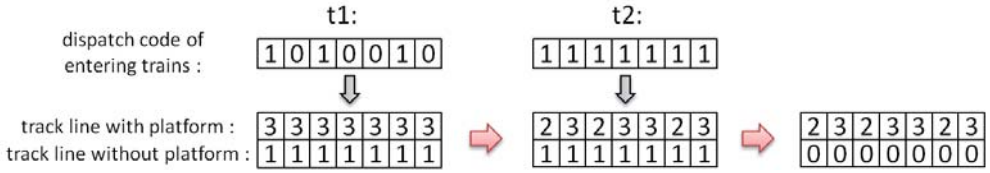

Fig 6- station capacity representation before and after capacity allocation

10 DEADLOCK SITUATION IN RAILWAY OPERATIONS

In railway operation, a train should not be allowed to enter a block section if this movement leads into a real deadlock situation. A deadlock situation occurs when all waited trains are blocked by others and none of them can continue their journey. For accurate simulation, deadlocks should either be avoided completely or at least be decreased to a satisfactory level. A train's movement can only be approved if the simulation model will still keep processing without deadlock situation. The work of Petersen & Taylor (1989) is one of the first papers concerning with the deadlock avoidance problem. Mills and Pudney (2003) developed a deadlock avoiding algorithm applicable on single track railway lines. They developed a vector-based algorithm based on the algorithm of Petersen & Taylor, called the 'labeling algorithm'. Cui (2010) addressed the deadlock problem during a synchronous simulation. Cui (2010) also proposed the Banker's algorithm for railway synchronous simulation, which is developed originally for deadlock prevention problem in computer science. Deadlocks usually occur in the single track lines or in the station yard with bidirectional operations. Resolving the deadlock conflict can be done by determining the sequence of entering trains in block sections. A simple example of deadlock situation is shown in Fig 7. There is a potential for deadlocks between trains going in opposite directions. Here, trains T_1 and T_2 are inbound trains and also T_3 and T_4 are outbound trains. Train T_3 is going to run along the specific route marked in Fig 7. But other trains can enter to every track of stations 1, 2 and 3.

Fig 7. A simple potential deadlock situation

If train T_1 enters to block B_1 then a deadlock situation occurs, because all track of station 2 is reserved and the capacity of station 3 is also full. Hence, Trains T_1 and T_2 are blocked by trains T_3 and T_4 and a deadlock situation occurs. Train T_2 cannot enter to block B_2 because there is no free track in the next station (S_3). For avoiding deadlock situation, the correct sequence of the train movements on block B_2 should be T_4-> T_2-> T_3-> T_1.

11 THE PROPOSED DEADLOCK AVOIDANCE APPROACH

The proposed deadlock avoidance techniques in computer science cannot effectively be adapted to railroad applications. Also, deadlock avoidance algorithms developed in railway literature are not compatible with the dispatching rules of Iranian railway network. Therefore, a customized deadlock avoidance method has been developed based on a conditional capacity allocation. The proposed method follows the idea that, before a train be approved to enter a block section, the unreserved tracks in the next station may be untouched to avoid creation of a real deadlock. A simple rule for deadlock avoidance is a way that all the capacity of a station cannot be occupied by trains with the same direction. That means, one free line and one platform always exist for passing trains with opposite direction. The implementation of this rule in the simulation model can reduce the utilization level of railway resources. Because a potential deadlock situation does not always sufficiently lead to real deadlock. The deadlock avoidance is performed by imposing additional waiting time in stations. All the potential deadlock situations should be detected in the simulation model. When a train arrives at the stopping check point in a waiting queue, a dispatch checking algorithm is called up to examine all the conditions of occurring potential deadlock situation. A potential deadlock situation in the next station occurs if reserving a track in the next station sets its capacity to zero. In case of potential deadlock, if there is not any train with opposite direction in the two next stations or there is at least one train with the same direction in the next station, the involved train have permission to dispatch. But if both above conditions are not satisfied, then the involved train has no permission to dispatch because this potential deadlock situation may become a real deadlock. The proposed look-ahead deadlock avoidance approach is easy to implement in the simulation and very effective for real time railway operation control.

12 OPTIMIZATION ALGORITHM

In this section, the proposed meta-heuristic algorithm is described. Before introducing the meta-heuristic algorithm, the solution representation should be developed. Each solution (S_i) is represented with an array consist of departure time of trains from origin stations. In solution S_i=[d_1, d_2, d_3,…,d_n] the d_j denotes departure time of train *j* from origin station. Departure time of train *j* produces in interval [d_j^{min},d_j^{max}]. If new generated solution is an infeasible solution then a penalty value will be added to objective function value. Therefore, the objective function value (z) is calculated as below:

$$minimize\ \ z = \sum_{j=1}^{n} \sum_{i=1}^{m_j} T_{ij} + \sigma . \sum_{p=1}^{K} \sum_{j=1}^{n} pen_{pj} \quad (3)$$

In equation (3), the T_{ij} is the unplanned stopping time of train *j* in station *i*, the parameter m_j is the number of stations visited by train *j*, the σ is penalty value of infeasibility and $pen_{pj} \in \{0,1\}$ determines whether train *j* has an infeasible stopping plan for *p*th praying service. A train stop is infeasible if there exists no station that satisfies the praying constraint.

13 EVOLUTIONARY PATH RE-LINKING

Path re-linking is a search strategy that explores the path connecting two solutions. Given two solutions (initial and target solutions), their common elements are kept fix and the space of solutions spanned by these elements is searched with the objective of finding a better solution. This search technique was originally proposed by Glover (1996) as an intensification strategy. For each intermediate solution, the objective function value is obtained and the best intermediate solution is selected for next iteration of path re-linking procedure. Numerous strategies for path re-linking have been considered in recent applications.

These include forward, backward, back and forward, truncated, mixed, greedy randomized adaptive and evolutionary path re-linking (Ribeiro and Resende, 2010). In this paper, an evolutionary path re-linking algorithm is proposed to solve the train scheduling problem. Evolutionary optimization approaches explore the solution space by generating and then evolving a population of solutions. The evolution is done by methods that generate new solutions by means of combination of two or more solutions those are in the current population. The evolutionary path re-linking algorithm searches the path between two elite solutions selected from elite set to find new high quality solutions and to improve the quality and diversity of the elite set. Resende et al. (2010) used evolutionary path re-linking for the max-min diversity problem.

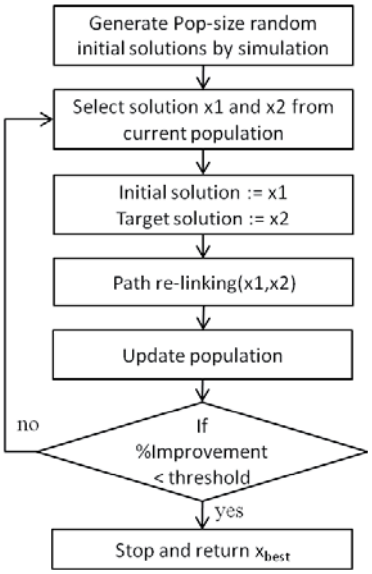

Fig 8- The flowchart of proposed evolutionary path re-linking algorithm

In evolutionary path re-linking, the solutions are evolved as a series of populations $P_1,P_2,\ldots$ of equal size. In the kth iteration of the optimization algorithm, path re-linking is applied between a set of pairs of solutions in population P_k . Each output solution from the path re-linking method is a candidate for insertion in population k+1. If population k+1 is not completed yet, then the candidate solution is accepted if it differs from all solutions in the population. If population k+1 is complete, the solution is accepted if either it is better than the best solution in the population or it is not the worst and is sufficiently different from all solutions in the population. Once a solution is accepted for insertion into population k+1, it replaces the solution in population k+1 that does not have a better cost and that is most similar to it. The distance (similarity) between two solutions is defined as the number of trains with different departure times. This evolutionary process is repeated until no improvement is seen from one population to the next. The proposed optimization algorithm starts by producing pop-size number of initial random solutions by using simulation model. In the next step, two solutions x_1 and x_2 will be selected randomly from the current population and they used as initial and target solutions in path re-linking algorithm. The best generated solution is candidate to be added to elite set. This procedure stops when the percentage of improvement is less than a threshold after a number of iterations. The example in Fig 9. illustrates the path re-linking process. Each solution consist of trains departure times from origin station, during a 24 hour period. Departure times should be set in 5 minute steps (e.g. 8:00, 8:05, 8:10...). The path which explored between initial and target solutions in two iteration of forward path re-linking algorithm are depicted in Fig 9.

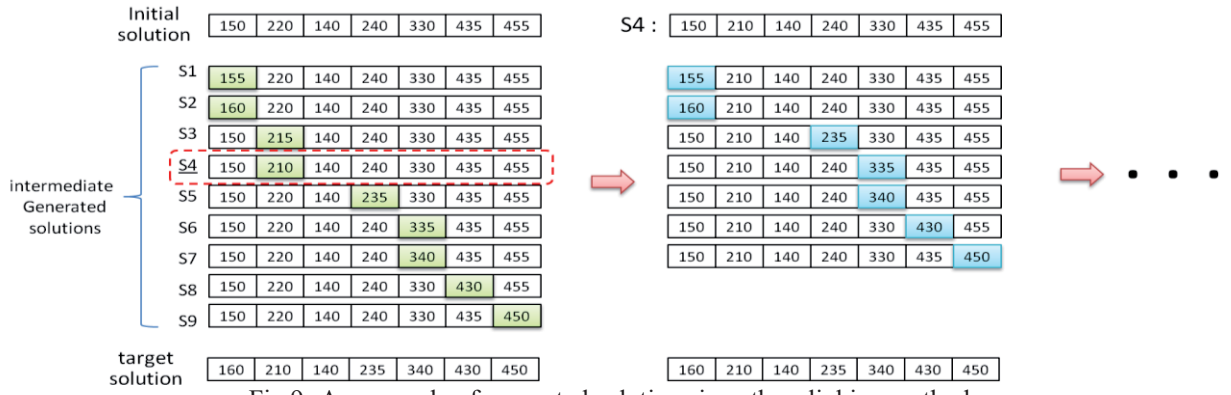

Fig 9- An example of generated solutions in path re-linking method

In forward path re-linking, the path is constructed during movement from initial solution to target solution. In the first iteration of path re-linking algorithm, a set of intermediate solutions has been generated. The distance between solutions can be calculated by number of trains with different dispatching times. In the next iterations, this procedure continues by linking the best intermediate generated solution (S_4 in this example) to the target solution. The best generated solution is candidate as initial solution in the next iteration of path re-linking algorithm. The trains with changing in departure time are highlighted in Fig 9.

14 COMPUTATIONAL ANALYSIS

In this section, the effectiveness of proposed optimization algorithm and simulation methodology will be demonstrated. The test instance is based on Tehran-Razi corridor which is a part of the Iran railway network. This real case includes 46 train services, 56 stations, 4 double-tracked and 51 single-tracked block sections. The information of train services in Tehran-Razi corridor is summarized in Table 2.

Table 2- Information of train services in Tehran-Razi corridor

Train ID (Inbound)	Initial departure	Departure tolerance (min)	Dispatch code	Train ID (Outbound)	Initial departure	Departure tolerance (min)	Dispatch code
430	18:20	±60	1111111	431	17:10	±60	1111111
432	17:15	±60	1111111	433	17:55	±60	1111111
434	20:50	±60	0101010	435	20:15	±60	1010101
422	20:35	±60	1111111	423	18:15	±60	1111111
420	23:00	±60	1111111	421	16:55	±60	1111111
462	14:35	±30	1111111	463	06:15	±30	1111111
454	05:55	±30	1111111	455	14:30	±30	1111111
460	15:20	±60	1111111	461	06:45	±60	1111111
450	07:20	±60	1111111	451	15:50	±60	1111111
452	16:10	±60	1111111	453	06:00	±60	1111111
490	22:20	±60	0100100	491	03:30	0	0001010
494	22:30	±30	0010000	495	02:10	0	0000100
944	06:30	±60	0101011	945	16:00	±60	0101011
946	07:15	±25	1111110	947	08:25	±25	1111110
948	09:35	±25	1111110	949	12:15	±25	1111110
954	13:10	±30	1111110	955	15:35	±30	1111110
956	16:40	±30	1111110	957	17:35	±30	1111110
480	10:45	±60	1111111	481	17:40	±60	1111111
489	09:00	0	0100010	488	19:10	±60	0001001
498	21:50	±60	1000100	499	09:00	0	0010001
482	20:25	±60	0101010	483	21:50	±60	1010101
484	22:15	±60	1010101	485	23:55	±60	0101010
486	23:40	±60	0101010	487	23:55	±60	1010101

The stopping time for every praying service is considered 20 minutes and the infeasibility value is $\sigma = 50$ minute according to previous experiments. The optimization algorithm was run on an Intel Core i5 3.3 GHz processor PC with 4 Gb of RAM, implemented in 4DScript programming language in ED.

Fig 10, shows the best objective value in different iterations for evolutionary path re-linking algorithm in solving the problem with 46 trains and 56 stations. The evolutionary path re-linking algorithm finds its objective value (225.2) in the 165th iteration and stays in this value up to the 210th iteration. The convergence graph briefly illustrates the step by step attempt of algorithm towards achieving the near optimal solution.

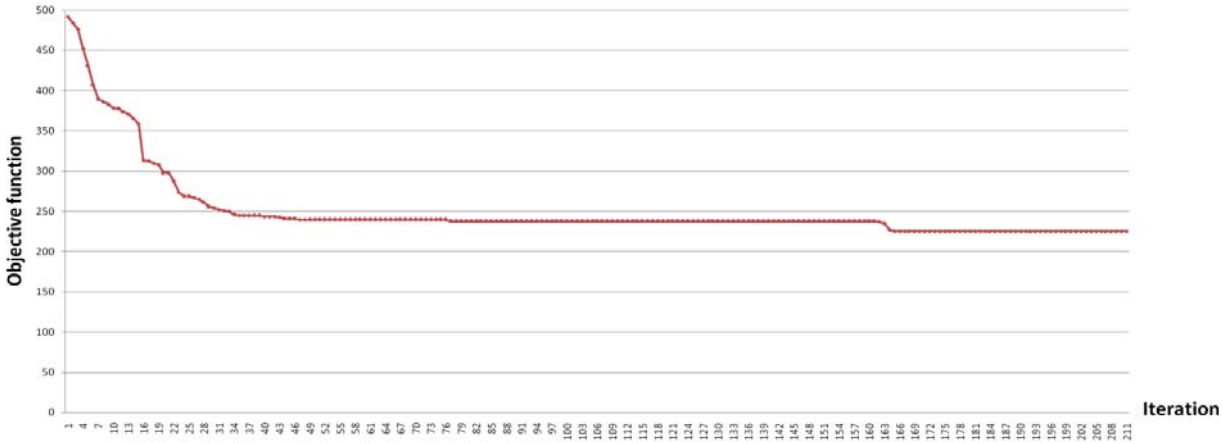

Fig 10- Convergence graph of evolutionary path re-linking algorithm

15 CONCLUSION

In this study, a variety of motivating aspects relating to timetabling in the context of the Iranian railway network are presented. These aspects include how to plan stops for regular prayer breaks, conflict detection and resolution procedure in the simulation model, deadlock avoidance method and the application of evolutionary path re-linking to the timetabling problem. The integrating of simulation modeling and metaheuristic algorithm is introduced as an effective approach in solving train scheduling problems. The simulation model is designed as a decision support system which can be employed to evaluate different train dispatching scenarios and generating a conflict and deadlock free timetable. In optimization module, an evolutionary path re-linking algorithm is proposed for obtaining good quality train timetable. A case study of railway network of Iran is selected for examining the efficiency of the evolutionary path re-linking algorithm. The result shows that the proposed algorithm has the capability of generating good quality timetable in real-world train scheduling problems. Future research should concentrate on the investigation of more detailed simulation models and powerful optimization algorithms.

REFERENCES

Cacchiani, V. and Toth, P .2012. " Nominal and robust train timetabling problems", European Journal of Operational Research 219 , 727–737.

Cui, Y. 2010, "Simulation-Based Hybrid Model for a Partially-Automatic Dispatching of Railway Operation", Institut für Eisenbahn- und Verkehrswesen der Universität Stuttgart.

Cheng, Y. 1998. "Hybrid Simulation for Resolving Resource Conflicts in Train Traffic Rescheduling" Computers in Industry 35(3):233–246.

Dessouky, M. M., and Leachman, R .1995. “A Simulation Modeling Methodology For Analyzing Large Complex Rail Networks.” Simulation 65(2):131–142.

Dorfman, M.J., Medanic, J .2004. "Scheduling Trains On A Railway Network Using A Discrete Event Model Of Railway Traffic" Transportation Research Part B 38 (1), 81–98.

Glover, F .1999. "Scatter Search and Path Re-linking", New ideas in optimization, McGraw-Hill Ltd., UK Maidenhead, UK, England.

Glover, F., Laguna, M .1997. "Tabu Search" Boston: Kluwer Academic Publishers.

Geske, U .2006. "Railway Scheduling With Declarative Constraint Programming", Lecture Notes in Artificial Intelligence 4369 117–134.

Gorman, M. F .1998. "An Application Of Genetic And Tabu Searches To The Freight Railroad Operating Plan Problem", Annals of Operations Research 78, 51–69.

Hillier, F. S., and G. J. Lieberman. 2009. "Introduction to Operations Research" 9th edition. New York:McGraw-Hill.

Li, F., Gao, Z. Li, K. and Yang, L .2008. "Efficient Scheduling of Railway Traffic Based on Global Information of Train" Transportation Research Part B 42:1008–1030.

Marinov, M., Viegas, J .2010. "A Mesoscopic Simulation Modeling Methodology For Analyzing And Evaluating Freight Train Operations In A Rail Network", Simulation Modeling and Practice Theory.

Mills, G.; Pudney, P .2003. "The Effects of Deadlock Avoidance on Rail Network Capacity and Performance" Proceedings of the Mathematics-in-Industry Study Group.

Middelkoop, D. and Bouwman, M .2001. "Simone: Large Scale Train Network Simulations" In Proceeding of the 2001 Winter Simulation Conference, volume 2, pages 1042–1047.

Petersen, E. R. and Taylor, A. J .1982. "A Structured Model for Rail Line Simulation and Optimization," Transportation Science, Vol. 16, pp. 192-206.

Pachl, Jörn: Avoiding Deadlocks in Synchronous Railway Simulations., 2007.

Petersen, E. R., Taylor, A. J .1983. "Line Block Prevention in Rail Line Dispatch", In: INFOR Journal 21, No. 1,pp. 46-51.

Ribeiro, C.C. and Resende, M.G.C. 2010. "Path Re-linking Intensification Methods For Stochastic Local Search Algorithms", Journal of Heuristics Volume 18, Number 2, 193-214.

Schriber, T. J. 1995. "How Discrete-Event Simulation Software Works". EUROSIM ’95, pp 17-28.

Valkenaers, P. and Van Brussel, H .2003. "Deadlock Avoidance In Flexible Flow Shops With Loops" Journal of Intelligent Manufacturing 14, S. 137-144.

AUTHOR BIOGRAPHIES

ERFAN HASSAN NAYEBI received his M.Sc. for industrial engineering at Sharif university of Technology in 2012. During his career at SIMARON he has contributed in a wide variety of simulation-based optimization studies for railway companies. His research interests are in the area of mathematical programming, simulation and meta-heuristic algorithm. He can be reached at e.nayebi@simaron.com.

SOHEIL MARDANI is a founder of Simaron Pardaz Co., a company dedicated to developing simulation solutions for different industries. He is a specialist in applying discrete-event simulation for complex manufacturing, logistics & transportation systems, and for modeling business systems as well. Besides running his company, he also teaches correspondence courses about simulation at universities for BSc & MSc students of Industrial & Transportation Engineering. His email address is s.mardani@simaron.com.

ARMAN SAJEDINEJAD is an Assistant Professor of Industrial Engineering at the Industrial Engineering Department, Faculty of Engineering, Azad University, MIS, Iran. His research background includes: Supply Chain Management, JIT, Simulation based optimization and also Heuristic Algorithms. He obtained his PhD from Tarbiat Modares University, Iran. His email address is sajedinejad@simaron.com.

S. AHMAD REZA MIR MOHAMMADI K. received the B.Sc. degree in Industrial Engineering from Amirkabir University of Technology, Tehran, Iran, in 2010. Currently, he is studying M.Sc. Systems En-

gineering, Policy Analysis and Management (SEPAM) at Delft University of Technology, Delft, The Netherlands. His research interests are simulation modeling, mathematical programming, scheduling and expert systems. He had been working in Simaron Pardaz Co. from 2010 to 2011. His e-mail address is s.a.r.mirmohammadikooshknow@student.tudelft.nl.