Job-Shop Scheduling Model for Optimization of the Double Track Railway Scheduling

(Case Study: Solo-Yogyakarta Railway Network)

Sarngadi Palgunadi
Department of Informatics, Faculty of Mathematics and Natural Sciences
Sebelas Maret University
Surakarta, Indonesia
palgunadi@staff.uns.ac.id

Dian Supraba
Department of Informatics, Faculty of Mathematics and Natural Sciences
Sebelas Maret University
Surakarta, Indonesia
dian.supraba@student.uns.ac.id

Bambang Harjito
Department of Informatics, faculty of Mathematics and Natural Sciences
Sebelas Maret University
Surakarta, Indonesia
bambang_harjito@staff.uns.ac.id

***Abstract*—Scheduling trains in order to minimize the traveling time is a challenging optimization problem. Delay may occur due to the bottleneck or many trains need to use the same track at the same time. These delays increase the journey time and may cause secondary delays to the other trains in the network. Double track railway scheduling can be modelled as a Job-Shop Scheduling Problem that can be achieved by considering the train trips as jobs which will be scheduled on tracks. This research is focus on scheduling trains in order to minimize the delay time. First Come First Serve (FCFS) dispatching rule is used to calculate the traveling time at each block section by sequencing the trains in order of starting time. The other rule is priority queue to solve the bottleneck problem by assigning the block section to the highest priority train. The simulation result is that the proposed model produce a less optimal schedule than the actual schedule due to the great average delays but the number of the delayed trains are less.**

Keywords—job-shop scheduling; railway scheduling; double track; FCFS; priority queue.

I. INTRODUCTION

Scheduling trains in order to minimize the traveling time is a challenging problem because the passengers demand on railway transportation is expected to increase in the future [13].

Delay may occur due to reasons such as non-availability of free tracks for arriving trains or many trains may have to use the same track at the same time [6]. One track can only be used by exactly one train. When other trains need to use the same track, only one train is allowed to use the track while the other must have to wait at the other free track (in this case, we called the alternative block), thus cause primarily delays. These delays may cause secondary delays to other trains in the network.

Some algorithms has been used to solve the scheduling problem such as tabu search [8], genetic algorithm [15], neural network [3], Ant Colony Optimization (ACO) [16], simulation annealing [14], and greedy job-shop [7].

D'Ariano, Pacciarelli, & Pranzo (2007) showed the graph formulation to solve the Conflict Resolution Problem (CRP) by using branch and bound algorithm and the rule such as First Come First Serve (FCFS), First Leave First Serve (FLFS), and Avoid Most Critical Completion time (AMCC) [2]. Putra (2011) showed a job-shop model to scheduling trains by using the real data schedule but only a few data are used to test the model and he conflict among trains is not considered as a part of the scheduling model [12]. Kumar & Rao (2015) showed the track occupancy by allocate the trains to the appropriate track to reduce the waiting time at the outskirts of a station using FCFS scheduling algorithm [6]. Zhang Yan & Cui Yanping (2013) showed a double track railway scheduling model for passenger and freight train using ACO algorithm [16].

This research is focus on minimizing the delay time by modeling the scheduling problem as Job-Shop Scheduling Problem with FCFS as the main dispatching rule and priority queue to solve the conflict among trains. The main track will be used by the first arriving train at block section by default. But if the collision is occurred, the highest priority train is allowed to use the next block section while the other trains must wait at the alternative block until the highest priority train reach the next 2 block sections in front of the lower priority train.

II. TRAIN DISPATCHING RULES

Railway network is composed of track segments and signals. Signal allows to control of traffic on the network and avoid any potential collision among trains. There are signals along the lines, inside the station, and before every junction. A track segment between two signals is called block section [2].

There are three main aspects of Indonesian signaling system that are red, yellow, and green. The red aspect means that the train must stop because the subsequent block section is not available or occupied by another train. The yellow aspect means that the subsequent block section is empty while the one after is not available. And the green aspect means that the two subsequent block sections are empty and the train allowed to enter the block section. Fig. 1 showed the three aspects of signaling system.

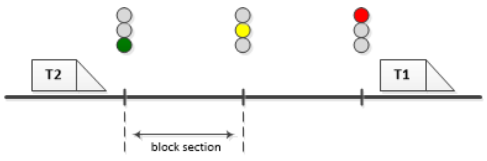

Fig. 1. Signaling System

The passing of a train through a block section is called operation which takes a time to be traversed, called travel time. Delay may occur along the lines when the bottleneck is occurred. The travel time of one operation starts when the head of the train enters the new block section. The train is not allowed to stop before the head enters a new block section. The previous block section remains out of service for a certain amount of time called setup time. The setup time is necessary because it takes a time to travel to the next block section plus a service time at the present block section. After the setup time, the signal of previous block section switched into yellow and the one before switched into green.

Based on the safety rule, only one train is allowed to enter one block section and conflict occurs whenever more than one trains require the same block section at the same time. A conflict happens usually in complicated network with bidirectional travels which cause the primarily delays to the waiting trains and may cause the secondary delays to the next scheduled trains [5].

The most common dispatching rule is First Come First Serve (FCFS). With FCFS, the block section is assigned to the top queue train or the train with the smallest start time. This rule does not require any special dispatching order and lets the traffic proceed by its actual order [1]. While the priority queue rule assigns the block section to the highest priority train. With this rule, the priority has got to be defined first and the precedence is then given to the highest priority train while the lower priority train must wait during setup time of the highest priority train.

III. DATA MODELLING

We use 86 data of trains where 43 trains are scheduled to the left block section and 43 trains are scheduled to the right block section.

A. Trains

The passenger and freight train divided into 6 type that is executive, mixed (executive/business/economy), business, economy, local, and cargo [9]. Each train has name, type, departure and arrival time, direction of departure, average speed (km/hour), and service time at each block section. Service time are classified as follows:

Passing: trains that only passes the block section. This service time is different for each train since the trains traveled in different speed. Continuous: trains which passes through the present block section and stop for a while for around 2 up to 15 minutes and leaves the block section. Start/End: trains that start from the present block section or end at the present block section. Trains those start from the present block section will be kept before their next scheduled departure time. And trains those final destination block section is the present block section will stay for amount of times and sent to their loco sheds [6]. These time are already considered at the actual schedule and not give in the simulation.

B. Blocks

The railway network along Solo – Yogyakarta is divided into 11 block sections which are the active stations. Each block section consist of blocks as in Fig, 1. The block may be the main block or the alternative block which is the branch of the main block. The bold lines are the main blocks which will be traverse by trains while the other are the alternative blocks.

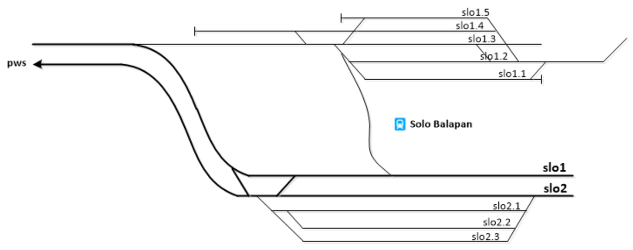

Fig. 2. Example of One Block Section

Every block section has length which is the distance from the current block section to the subsequent block section. The track is for bidirectional travels with two main blocks. The first block is for the left direction trains (Solo – Yogya) and the second is for the right direction trains (Yogya – Solo).

IV. IMPLEMETATION

Implementation runs in Greedy method. There are three main implementation steps. The first step is to calculate the traveling time from start block to finish block including total travel time, actual finish time, and penalty for each train. The second step is to calculate the arrival time at each block using the first step result as the input. The first and second step result are used in the last step. The last step is to check the potential collision between two trains and calculate the delay time and then update the total travel time, actual finish time, and penalty.

A. Job-Shop Scheduling Model

In this scheduling case, trains are considered as jobs which will be scheduled on block sections. A set of train $i = \{i_1, i_2, . . . , i_n\}$ must be scheduled on a set of block $j = \{j_1, j_2, . . . , j_n\}$. Every job i consist of a set of operations where every operation require exactly one block section for certain amount of time. Table I showed the notations that will be used in the model.

TABLE I. NOTATIONS

	Notation	Description
Sets / Parameter Input	R	A set of trains which travel from right to left
	L	A set of trains which travel from left to right
	X	A set of blocks that must be through by R
	Y	A set of blocks that must be through by L
	A	The travel time matrix of train R
	B	The travel time matrix of train L
	C	The arrive time matrix of train R
	D	The arrive time matrix of train L
	$S_{i,j}$	Service time of train i at block j
	s_i	The departure time of train i at the start station
	f_i	The arrival time of train i at the destination station
	l_j	The length of block j or the distance between block j and $j+1$
	v_i	The average speed of train i
Output	d_i	Delay of train i
	$tt_{i,j}$	Travel time of train i at block j
	$at_{i,j}$	Arrive time of train i at block j
	ttl_i	Total travel time of train i through blocks
	af_i	Actual finish time: the simulation result of arrival time at the destination station
	P_i	Penalty of train i, which is the objective function

Train i has a *start time* (s_i) and *finish time* (f_i). Train i must travel through blocks to reach the destination station. The time to traverse one block is called *travel time.* A travel time is the time to traverse one block plus the service time at its block, as in (1) as follows:

$$tt_{i,j} = \frac{l_j}{v_i} + S_{i,j} \tag{1}$$

According to train dispatching rule, the distance between two trains is two block sections away. A collision occur when train *i-1* set to depart from its current block section while train i is still in the two block section in front of train *i-1*. If the train type of *i-1* is higher than i, train i is moved to the alternative block and wait for a setup time which is the travel time of train *i-1* from its current block section until two block sections away in front of train i. Then, train i is allowed to move forward. Fig. 3 showed the delay calculation for the lower priority train. Train i has the higher priority than train k. If the arrive time of train k at its current block section is greater than the arrive time of train i at the same block section of train k current block section, then these trains are collide. Train k has to wait at its current station while train i is allowed to move forward. The delay of train k is the traveling time of train i from its current block section until two block sections in front of train k, as in (2) as follows:

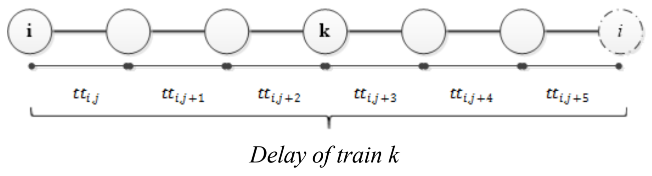

Fig. 3. Delay calculation of the collision among two trains

$$d_k = tt_{i,j} + tt_{i,j+1} + tt_{i,j+2} + tt_{i,j+3} + tt_{i,j+4} + tt_{i,j+5} \tag{2}$$

Arrive time is used to check the potential collusion among two trains. Train i and k are non-conflicting if $f_k \leq s_i$ [4] as we show in the model, $at_{i,j} \leq at_{i-1,j+3}$. As in (3), arrive time of train i at block j is its start time. Then we need to add the traveling time to calculate the arrive time at the next block section,

$$at_{i,j} = \begin{cases} s_i & for\ j=1 \\ s_i + tt_{i,1} & for\ j=2 \\ at_{i,j-1} + tt_{i,j-1} & for\ j=3, 4, ..., n \end{cases} \tag{3}$$

As in (4), the total time of train i to traverse blocks is called *total travel time*, which is the total traveling time of train i to traverse blocks from start block to finish block plus its delay.

$$ttl_i = \sum_{j=1}^{n} tt_{i,j} + d_i \tag{4}$$

As in (5), *actual finish time* is the arrival time of train i at the destination station which is the start time of train i plus its total travel time.

$$af_i = s_i + ttl_i \tag{5}$$

The aim of this problem is to minimize the delay time. So the objective function can be described as in (6) to find out how much losses due to the waiting time or delay. This function can be represented as *penalty*. The negative value indicate that the train has been delayed and the positive value indicate that the train reach the destination earlier than the actual schedule. Penalty is the difference between finish time and actual finish time of train i multiplied by its weight. Table II showed the weight for each train type. These weights also indicate the priority. The greater the weight, the higher the priority.

$$P_i = (f_i - af_i) \times weight \quad (6)$$

TABLE II. TRAIN TYPE AND ITS WEIGHT

Weight	Train Type
6	Executive
5	Mixed
4	Business
3	Economy
2	Local
1	Cargo

B. *Pseudo Code*

There are three main pseudo code which implement the proposed steps as follows:

1) *Travel time calculation*

This pseudo code below is used to calculate the travel time. The input are the data of trains and block sections while the output is a matrix of travel time of trains at each block section. This output will be used in the second and third step.

```
while $R$ and $L \neq \emptyset$ do
  remove $i$ from $R$ or $L$ with the smallest start time (if $i \in R$, schedule $i$ on $X$, if $i \in L$ schedule $i$ on $Y$)
  $ttl_i \leftarrow 0$
  for ($j$ from Start Block to Finish Block) do
    if ($j$ != Finish Block) then
      schedule task $i$ on block $j$
      calculate $tt_{i,j}$
      $ttl_i \leftarrow ttl_i + tt_{i,j}$
    else
      $tt_{i,j} \leftarrow 0$
      $ttl_i \leftarrow ttl_i + tt_{i,j}$
      calculate $af_i$ dan $P_i$
    end if
  end for
end while
```

2) *Arrival time calculation*

This pseudo code below is used to calculate the arrival time. The input are the data of trains and the matrix of travel time while the output is a matrix of arrive time of trains at each block section. This output will also be used in the last step.

```
while $A$ and $B \neq \emptyset$ do
  remove $i$ from $A$ or $B$
  for ($j$ from Start Block to Finish Block) do
    if (j = 1) then $at_{i,1} \leftarrow s_i$
    elseif (j = 2) then $at_{i,2} \leftarrow s_i + tt_{i,0}$
    else $at_{i,j} \leftarrow at_{i,j-1} + tt_{i,j-1}$
    end if
  end for
end while
```

3) *Checking potential collusion and delay calculation*

This pseudo code below is used to check the potential collision among train *i* and *i-1* and calculate the delay of train *i-1*. The input are the matrix of travel time and the matrix of arrive time while the output is the delay of each train.

```
while $C$ and $D \neq \emptyset$ do
  remove $i$ from $C$ or $D$
  if ($f_{i-1} \leq s_i$) then train $i$ and $i-1$ are non-conflicting
  else train $i$ and $i-1$ are conflict
    if ($trainType_i > trainType_{i-1}$) then
      train $i-1$ delayed
      for (j from 1 to n) do
        if (j $\neq$ n-2 or j $\neq$ n-1) do
          if ($at_{i-1,j+3} \geq at_{i,j}$) then
            train $i-1$ moved to the alternative block and wait for amount of time as follows:
            if (Alternative block$_{j+3}$=available) then
              train $i-1$ waiting at Alternative block$_{j+3}$
              calculate $d_{i-1}$
            else
              looping to search the previous alternative blocks
              for ($k$ from $j+3$ to Start Block) do
                if (Alternative block$_k$=available) then
                  train $i-1$ waiting at Alternative block$_k$
                  calculate $d_{i-1}$ start from $tt_{i,j}$ until two block sections in front of $k$
                else
                  there is no alterative block, train $i-1$ is not delayed, $d_{i-1} \leftarrow 0$
                end if
              end for
            end if
        else
          the end of block section, $d_{i-1} \leftarrow 0$
        end if
      end for
    else
      train $i-1$ is not delayed, $d_{i-1} \leftarrow 0$
    end if
  end if
  $ttl_{i-1} \leftarrow ttl_{i-1} + d_{i-1}$
  $af_{i-1} \leftarrow af_{i-1} + d_{i-1}$
  $P_{i-1} \leftarrow (f_{i-1} + af_{i-1}) \times trainType_{i-1}$
end while
```

V. RESULT

Table III showed the result of the first step of implementation. Table IV showed the result of the second step. There are 10 delayed trains as in Table V and Table VI, while others arrive on time as the actual schedule. Fig. 4 showed the difference of actual finish time between simulation and the actual schedule.

TABLE III. EXAMPLE OF TRAVEL TIME (SOLO – YOGYA)

Train Id		1	2	3	…	43
Travel Time at Each Block Section (min)	slo	2.02	2.02	2.56		1.94
	pws	5.18	7.18	9.58		4.98
	gw	4.19	6.19	5.32		4.03
	dlg	3.92	5.92	4.98		3.77
	ce	11.81	13.81	14.99		11.36
	kt	7.03	7.03	8.38		4.84
	swt	4.57	6.57	5.80		4.40
	bbn	9.72	11.72	12.34		9.35
	mgw	4.57	6.57	7.80		4.40
	lpn	0.98	2.98	4.24		0.94
	yk	0	0	0		0
Total Travel		54	70	76		50
Actual Finish		237	273	391		1455
Penalty		0	0	0		0

TABLE IV. EXAMPLE OF ARRIVE TIME (SOLO – YOGYA)

Train Id		1	2	3	. . .	43
Arrive Time at Each Block Section (min)	slo	183	203	315		1405
	pws	185.02	205.02	317.56		1406.94
	gw	190.20	212.20	327.14		1411.92
	dlg	194.39	218.39	332.46		1415.95
	ce	198.32	224.32	337.44		1419.73
	kt	210.13	238.13	352.43		1431.08
	swt	217.16	245.16	360.81		1435.92
	bbn	221.73	251.73	366.62		1440.32
	mgw	231.45	263.45	378.96		1449.67
	lpn	236.02	270.02	386.76		1454.06
	yk	237	273	391		1455

TABLE V. DELAY TIME (SOLO – YOGYA)

Train Id	Train Type	Delay	Total Travel	Actual Finish	Penalty
14	3	27.83	103.83	795.83	-83.48
21	1	39.11	109.11	964.11	-39.11
30	2	28.83	103.82	1123.82	-57.65
35	5	19.77	81.77	1221.77	-98.83
37	2	32.92	123.92	1203.92	-65.84
Average Delays	29.69 minutes				

TABLE VI. DELAY TIME (YOGYA - SOLO)

Train Id	Train Type	Delay	Total Travel	Actual Finish	Penalty
1	4	37.66	85.66	133.66	-150.64
16	3	43.21	103.22	538.22	-129.64
26	1	40.73	110.73	905.73	-40.73
28	2	58.66	139.66	969.66	-117.31
40	2	44.67	117.66	1342.66	-89.33
Average Delays	44.99 minutes				

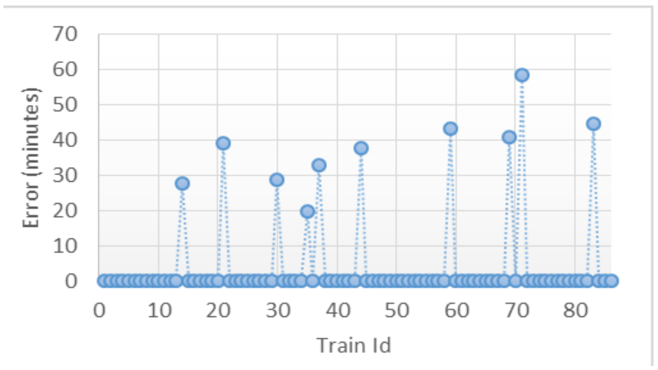

Fig. 4. The Difference of Actual Finish Time Between The Simulation and The Actual Schedule

The average delays are quite great because the low priority train must wait at the alternative block for the high priority to traverse the next block section at least 3 to 6 block sections. Delays are occurring to the middle and the low type trains because the priority is determined by train type which causes the possibility delays for the high trains type are less than the lower priority.

The other possible scenario is to determine the priority based on the total continuous stations or the lay-off station. According to [10], the total continuous stations for the high type trains are less than the lower type trains which caused the delays for the high type trains over the lower type. We have not considered this possible result as an optimal solution since the high type trains cost are more expensive. The objective function (penalty) will also become higher than the proposed model.

Based on the actual schedule [11], the average delays for the passenger and freight trains is 28.69 minutes. It is less than the total average delays of the simulation result, which is 37.34 minutes. While the number of delayed trains of the simulation is less than the actual schedule. The realization of the actual schedule shows that almost of the trains were delayed every day although the average delays are less than the proposed model.

VI. CONCLUSION AND FUTURE WORKS

This paper presents the scheduling model for the double-track railway. A mathematical model for a train scheduling problem was developed and then the algorithm based on Job-

Shop with FCFS and priority queue as the dispatching rules was presented to solve the problem. According to this case study, the proposed scheduling model is not promising to solve the problem because the average delays are greater than the actual schedule although the number of delayed train are less.

As for future research, a number of issues remain that need further development such as the consideration for the delayed train departure, a method to solve a conflict among trains, etc. Rescheduling method with some policies such as cancelling, reordering, etc. seems good to consider. The precision of the travel time calculation can be improved by considering train speed at each block rather than constant average speed.

ACKNOWLEDGMENT

The authors would like to acknowledge the support of Indonesian Railways especially PT KAI DAOP VI Yogyakarta for providing the relevant data.

REFERENCES

[1] Afonso, P. A. d. C., 2008. "Railway Traffic Management", Master thesis, Electrical and Computer Engineering: Universidade Técnica de Lisboa.

[2] D'Ariano, A., Pacciarelli, D. & Pranzo, M., 2007. "A Branch and Bound Algorithm for Scheduling Trains In A Railway Network". European Journal of Operational Research, 183(2), pp. 643-657.

[3] Dündar, S. & Şahin, I., 2013. "Train Re-scheduling with Genetic Algorithms and Artificial Neural Networks for Single-track Railways". Transportation Research Part C 27, pp. 1-15.

[4] Goodrich, M. T. & Tamassia, R., 2008. "Chapter 5, Fundamental Techniques: The Greedy Method. In: ALGORITHM DESIGN: Foundations, Analysis, and Internet Examples Second Edition". Daryaganj: Wiley India Pvt. Ltd., pp. 259-262.

[5] Khosravi, B., Bennell, J. A. & Potts, C. N., 2012. "Train Scheduling and Rescheduling in the UK with a Modified Shifting Bottleneck Procedure". 12th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS'12), pp. 120-131.

[6] Kumar, A. V. & Rao, S. G. R., 2015. "Optimization Of Railway Track Occupancy Using Scheduling Algorithm". International Journal of Advanced Research in Electronics and Communication Engineering (IJARECE), 4(9), pp. 2463-2466.

[7] Oliveira, E. & Smith, B. M., 2000. "A Job-Shop Scheduling Model for the Single-Track Railway Scheduling Problem". School of Computing Research Report 2000.21, University of Leeds, England.

[8] Pacciarelli, D. & Pranzo, M., 2001. "A Tabu Search Algorithm for the Railway Scheduling Problem. Porto", Proceedings of the 4th Metaheuristic.

[9] PT Kereta Api Indonesia (Persero), 2014. "Company Profile". [Online] Available at: https://www.kereta-api.co.id/media/document/company_profile_2014.pdf [Accessed March 1, 2016].

[10] PT Kereta Api Indonesia (Persero), 2015. "Jadwal Perjalanan Kereta Api 2015". [Online] Available at: https://tiket.kereta-api.co.id [Accessed March 1, 2016].

[11] PT Kereta Api Indonesia (Persero), 2016. "Evaluasi Keterlambatan Kereta Api Bulan Februari 2016", Yogyakarta: PT Kereta Api Indonesia (Persero).

[12] Putra, R. S., 2011. "Penjadwalan Kereta Api di Daerah Operasi 8 Surabaya", Surabaya: Institut Teknologi Sepuluh November. Unpublished.

[13] Sajedinejad, A., Mardani, S., Hasannayebi, E. & Mir Mohammadi K, S. A. R., 2011. "SIMARAIL: Simulation Based Optimization Software for Scheduling Railway Network". Tehran, Proceedings of the 2011 Winter Simulation Conference.

[14] Tamannaei, M., Saffarzadeh, M., Jamili, A. & Seyedabrishami, S., 2015. "A Novel Train Rescheduling Approach In Double-Track Railways: Optimization Model and Solution Method Based on Simulated Annealing Algorithm". International Journal of Computer Engineering (IJCE), 13(3).

[15] Tormos, P., Lova, A., Barber, F., Ingolotti, L., Abril, M. & Salido, M. A., 2008. "A Genetic Algorithm for Railway Scheduling Problems". Studies in Computational Intelligence (SCI) 128, pp. 255-276.

[16] Yan, Z., Yanping, C. & Wentao, Y., 2013. "A Method for Generating the Timetable of Double-track Railway Line". Paris, Proceedings of the 2nd International Conference on Computer Science and Electronics Engineering (ICCSEE 2013).