A DECISION SUPPORT SYSTEM
for
COMPUTER AIDED TRAIN DESPATCHING

AN INTELLIGENT WAY OF AUTOMATIC RAILWAY SIGNALLING:
A SOFTWARE TO CONTROL SIGNAL AT RAILWAY JUNCTIONS SO THAT WAITING TIME OF EACH TRAIN ON RAILWAY NETWORK WILL BE MINIMUM

COMPUTER AIDED TRAIN DESPATCHING: DECISION SUPPORT THROUGH OPTIMIZATION

DEVELOPED
BY
SUDIP RATAN CHANDRA(M.TECH CS)

INDIAN STATISTICAL INSTITUTE, CALCUTTA (INDIA).

UNDER GUIDENCE
OF
PROF. BIMAL ROY

ACKNOWLEDGEMENT

First of all I like to thank my guide Prof.Bimal Roy because he advised me to solve such a problem which is really challenging to solve .When I first met sir ,he told me that we should solve real time problem like this .In India approximately 10,000 trains run on complicated network and signalling done manually .So trains generally are delayed. That causes the trouble to the common passengers and goods trains .Thus I tried to solve this prob lem such a way so that we can run my algo. Online .Thanks to Mr. L. Ferreira of Queensland University of Technology ,Australia for sending me information about research that are going on in Australia for solving this problem with greater optimality and faster heuristic.

Sudip Ratan Chandra
27/7/98.

Bimal Roy

N. Chakravarti

Decision support system:

Train despatching is a problem of signalling the trains on its way so that each train has to wait minimum amount of time .For signalling the trains on its way of journey, we can use a central computer to take the despatching or signalling decision at each crossings. Here the central computer is connected through wide area network with each crossing signals .At each crossing there is mini computer which is connected through central computer. Each minicomputers gathers information of the traffic at the crossing and sends to the central computer .
Central computer collects the information ,process it and take the decision about which train should be signalled at that particular instant of time. That's why it is called real time decision support system. Our ambition is to formulate this problem as a optimisation model.

Abstract-*This paper describe a mathematical model designed to optimize train schedules on a Railway Network and then find a heuristic solution so that it can be implemented within a reastonable time limit.*

Introduction-*Indian railways operate 13,000 trains daily over avast 63,000 kms of track .They still use manual methods for controlling both fright and passenger trains over the complicated network . Because of heavy traffic on the network, the movement of trains become impossible .This traffic leads to over-the-road-transit delays as well as yard delays and thereby reducing the reliability and efficiency of network as a whole .This situation can be observed clearly in the case of passenger train operations where an increasing number of trains have outstripped the line capacity and thereby ensuing late running . To what extent can this situation can be resolved by certain decision making Tools ?Railway operations and planning have subjected to certain analytical studies specially in U.S ,Australia ,Japan, Europe Etc. In this paper I develop a mathematical programming model which can be used to generate an optimal train operating plan .But I find that to implement this mathematical model we have to sacrifice huge amount of computation(time) even in fast computers. So keeping in mind the computational sensitivity I have developed a heuristics solution which is capable of computing feasible schedule of Trains on a large network with many trains.*

Problem of Train Scheduling: *The Train Scheduling problem is an interesting and practical optimization problem. Suppose I describe the Railway network as a Graph where nodes are Crossings or junctions and edges are tracks of Rails. So each Train has its starting and destination junction. That means each train traverses a particular path of Graph (track) from staring to destination. Now because of heavy traffic the other trains also crossing its path and traverse full or partially of its path. This creates the problem of collision ,overtaking and Blocking of Railway line. My ambition is to find a feasible schedule of each train at each crossing so that there will be no collusion ,overtaking ,line blockage as well as waiting time of each at crossing will be minimum. Here the waiting time of each train at crossing arises because at the crossing we have to clear the other lines through which this particular*

train wants to travel. My mathematical model is based on network optimization.

RAILWAY NETWORK

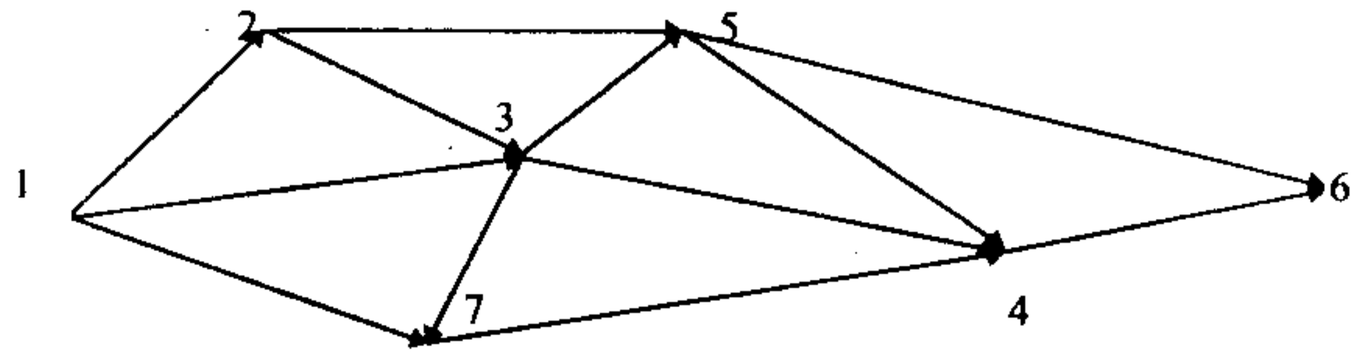
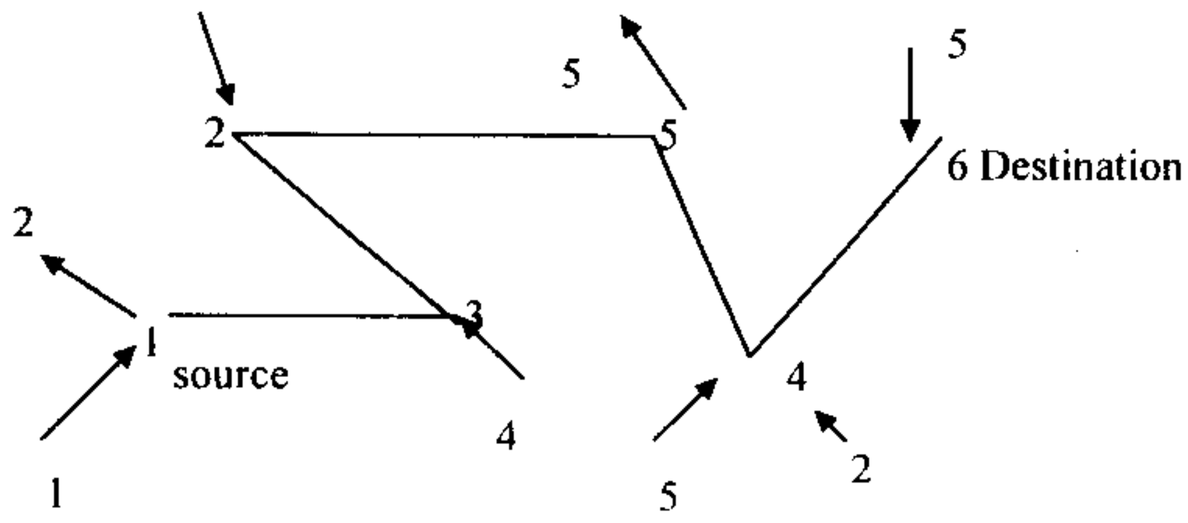


Fig.1 :In this network nodes are CROSSINGS and edges are TRACKS of the Railway

Consider the above network where each train has its own path of traverse from source to destination .Let us assume that there are five trains running on this complicated network and each train has its starting time. The paths of each train as follows:

Train_no	*path_of_train*
1:	*1-3-2-5-4-6*
2:	*4-5-2-3-1*
3:	*6-5-3-7-1*
4:	*3-2-1-7-4*
5:	*1-7-3-6-4-5*

Now we can see from the above paths that one track is accessed by several trains .So if we allow all trains to move parallely there may be collision, overtaking conflict, or blockage of line .Here blockage of line means that suppose a train is running on its particular path and another train has also entered its path at a particular instant of time .As several trains are accessing same track this situation may arise. In this situation the solution is that one train has to be backed. Consider a example of train no. 1 and observe the traffic on its path as follows;



We can see from above figure that the Train no 5, 2 , 4 are partially accessing the path of train _no .1 .So that creates the problem of smooth movement of train -no .1.To gain the conflict free movement of train-no.1 we should not allow the other trains to enter its path.. But now if we stop other trains at the crossings of train _no.1's path allow train no.1 until it crosses the conflict free sections. In that situation other train may have to wait for a long time. In this situation my aim is to decide which train I should allow first and then others one by one so that the waiting time of each train will be minimum. Then comes the question of easible

For example after final scheduling we get that train no 1, 3 ,7,5 will run on track -no2, 5 ,7 in the time interval 12.30pm to 1.00pm ,4.00pm to 4.50pm , 6.00pm to 8.00pm.Then no other will not in this track for that particular time period. That means we have to find a time -table of each train at each crossing.

Now another question arises that if trains are running late(Indian context) then how do I find a feasible schedule. My solution is we should find a schedule after 2-min time interval

So every two min we calculate new schedule by taking the recent state or distance from junctions. So our algorithm should fast enough to compute within 2 -min or less .

Now I develop a mathematical programming approach to find feasible schedule of trains.

Consider the above path of train _no.1 , here we see that some of the trains are coming from source direction of train -no.1 (train-no.1, 4) and going to the destination of train_no1's direction . We call this set of trains to be northbound trains assuming source as north . Another set of trains (2 ,5) are coming from destination direction of train_no.1 and going to the source direction of train_no.1.We call this set of trains to be southbound trains.

Here we consider the path of each train on a network and apply mathematical model for scheduling .Consider the train -i :

We define:

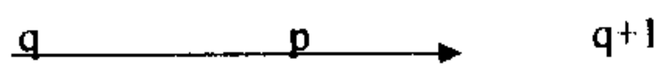
I= Total set of trains running on path of the train -k.

= {1 ,2 ,3, ...,m,m+1,.....N}

Where 1-m set of trains are north bound and m-N set of trains are southbound trains.

Q = Total no. of sidings between the source - destination pairs of train -i.

We consider that the track -p is separated by two crossings -k and k+1.



We consider the integer decision variables for determining which train traverses after which .That means to find the sequence of each train so that no conflict arises and move with a minimum time.

Let

$A_{i,j,p}$=1 if northbound train $i \le m$ traverses track $p \in P$ before northbound train $j \le m$

=0 otherwise

where P= set of track segments of the path of train-i.

$B_{i,j,p}$=1 if northbound train $i \le m$ traverses track segment $p \in P$ before southbound train $j > m$

= 0 otherwise

$C_{i,j,p}$= 1 if southbound train $i > m$ traverses track segment $p \in P$ before southbound train $j > m$

= 0 otherwise

Consider the above path of train _no.1 , here we see that some of the trains are coming from source direction of train -no.1 (train-no.1, 4) and going to the destination of train_no1's direction . We call this set of trains to be northbound trains assuming source as north . Another set of trains (2 ,5) are coming from destination direction of train_no.1 and going to the source direction of train_no.1.We call this set of trains to be southbound trains.

Here we consider the path of each train on a network and apply mathematical model for scheduling .Consider the train -i :

We define:

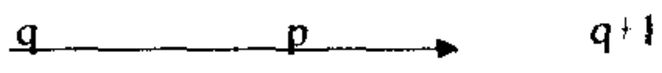
I= Total set of trains running on path of the train -k.

= {1 ,2 ,3, ……,m,m+1,……N}

Where 1-m are north bound and m-N are southbound trains.

Q = Total no. of sidings between the source - destination pairs of train -i.

We consider that the track -p is separated by two crossings -k and k+1.



We consider the integer decision variables for determining which train traverses after which .That means to find the sequence of each train so that no conflict arises and move with a minimum time.

Let

$A_{i,j,p}$=1 if northbound train i≤m traverses track p∈P before northbound train j <=m.

=0 otherwise

where P= set of track segments of the path of train-i.

$B_{i,j,p}$=1 if northbound train i≤m traverses track segment p∈P before before the southbound train j>m .

= 0 otherwise

$C_{i,j,p}$= 1 if southbound train i>m traverses track segment p∈P before southbound train j >m .

= 0 otherwise

X_{dq}^{i} =departure time of train i∈ I at q∈Q

X_{aq}^{i} = arrival time of train i∈ I at q∈Q

T_p^i = running time for train i∈ I at the destination

W_i = priority of train i∈ I

S_q^i = schedule stop for the train i∈ I at station q∈Q.

Delay at each crossing /waiting time at crossing q is

$$D_q^i = (X_{dq}^i - X_{aq}^i)$$

So our objective function :

$$\min \sum_{i=1}^{n} (X_{dq}^i - X_{aq}^i) W_i$$

Subjected to :

$$X_{dq}^i - X_{aq}^i \leq \partial t$$

Where ∂t =maximum allowable delay of each train at each crossing.

Overtaking Constraint : Identify southbound trains i,j >m for every p on the path .

Case 1: if train-j goes first :

$$X_{a\,q+1}^i \geq X_{a\,q+1}^j + (T_p^i - M C_{i,j,p})$$

$$X_{dq}^i \geq X_{dq}^j + (T_p^j - MC_{i,j,p})$$

Case 2: if train-i goes first :

$$X_{a\,q+1}^j \geq X_{dq}^i + [T_p^j - M(1-C_{i,j,p})]$$

$$X_{a\,q+1}^j \geq X_{dq}^i + [T_p^i - M(1-C_{i,j,p})]$$

Overtaking Constraint: Identify the northbound trains $i,j \leq m$ for every p on path of train.

Case 1: if train-j goes first

$$X_{aq}^i \geq X_{aq}^j + (T_p^i - MA_{i,j,p})$$

$$X^i_{d\,q+1} \geq X^j_{d\,q+1} + (T_p^i - MA_{i,j,p})$$

Case 2: if train-i goes first

$$X^i_{aq} \geq X^j_{aq} + [T^j_p - M(1-A_{i,j,p})]$$

$$X^i_{d\,q+1} \geq X^j_{d\,q+1} + [T^i_p - M(1-A_{i,j,p})]$$

Collision Constraint : When trains approach each other : $i \leq m$, $j \geq m$.

Case 1: if train j goes first

$$X^{j}_{a\,q+1} \leq X^{i}_{d\,q+1} + MB_{i,j,p}$$

Case 2: if train i goes first .

$$X^{i}_{aq} \leq X^{j}_{dq} + (1 - MB_{i,j,p})$$

Where the i,j are the one pair of trains.

$$X^{i}_{a\,q+1} \leq X^{i}_{dq} + T^{i}_{p} \quad \text{for} \quad i>m$$

$$X^{i}_{aq} \leq X^{i}_{d\,q+1} + T^{i}_{p} \quad \text{for} \quad i \leq m .$$

Where M is a large number for the safety allowance.

Application of above set of Eqns. :

Consider the path of a train i where a set of trains accessed the path of train i partially or fully .Say this set of train be $\{i_1, i_2, i_3, \ldots\ldots, i_n\}$.we call it conflicting set of train.

Apply the above equations by pairing (i,j) among the conflicting set of train(including i) .For that generate the all possible sequences of train (by permutation of train numbers) .From there pair up two consecutive trains and apply the above equations.

For example : if train 1,2 accessed the path of train 3 (say) then ,we generate factorial(3) sequences as follows:

Permutation	sequence	pairs(i,j)
1 2 3	1<2<3	(1,2) (2,3)
1 3 2	1<3<2	(1,3) (3,2)
2 3 1	2<3<1	(2,3) (3,1)
2 1 3	2<1<3	(2,1) (1,3)
3 1 2	3<1<2	(3,1) (1,2)
3 2 1	3<2<1	(3,2) (2,1)

For each pairs find out relationships between (i,j) and apply appropriate constraints.

By solving these set of equations for each train with particular path we can get a feasible arrival and departure time of each train at crossings .

Drawbacks: Now to solve these set of equations the time required as follows:

Time to calculate factorial for permutation of all train = $O(n.n!)$ $O(n^{n+1})$

Time to solve one linear program with n variables $=O(2^n)$ [Worst case]

So the total time is Exponential. **That means that this model is not time sensitive and online implementation is not feasible.**

OTHER SOLUTION: For calculating the feasible time table of trains I have developed another method which will take less time than the above model .

STEPS ARE AS FOLLOWS:

Step 1: Consider the path of each train and modify the expected arrival time table stated below.

Step 2: Consider the path of a train-i and find out the trains which are partially or fully accessing the path of train-i .We call this group of trains as conflicting group $I=\{i_1, i_2, \ldots\ldots, i_n\}$.

Step 3: Find out the all possible sequence of the trains $I \cup \{i\}$ on this path of train i

For example : say $I \cup \{i\}=\{1,2,4\}$,then all possible sequences are {124,142,214,241,412,421}.Each sequence, say 124 implies that we will allow the train 1 first and after passing the crossing of 2 and 4 ,the train 2 will be allowed in this path to go then train 4 Whenever one train is allowed to enter in path others are waiting for a time interval to pass the allowed train.

Step 4: Find out the waiting timing of each train for factorial(n) sequences. Find the minimum waiting time among factorial(n) sequences or schedules and Take that schedule which is corresponding to min Time.

Step 5: According the min waiting time schedule modify the arrival times of other conflicting group of trains by adding waiting on a crossing on the path of train i .

Step 6: Repeat the above steps by considering the rest of the trains.

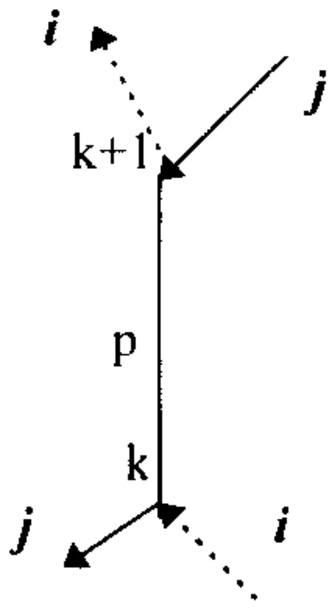
Time Complexity of algo: For generating the all possible sequences of conflicting train . We need factorial (n) computations .Finding the minimum among the factorial (n) waiting time we need O(n!) computation.

So for each trains we need O(n.n!) computations which is finally $O(n^{n+1})$ computations. **Which require much less amount of time than the previous mathematical model. But still this also quite expensive as far as computation time is concern. So I have found a HEURISTIC solution for the above train scheduling problem.**

A FAST HEURUSTIC FOR TRAIN SCHEDULING ON NETWORK:

Greedy Heuristic as a local optimality Criteria:

_We shall assume that the objective of scheduling trains is to minimize the waiting time of trains at each crossings .Now the GREEDY HEURISTIC means that local minimization will give the Global minimized solution. Here Local minimization is based on the local decision .Consider the following case.



Here we can see that two trains i,j are waiting at the junction k and k+1 respectively. Now we can't allow both the trains to enter in the track section p. So the question comes in mind that which train I should allow first to enter the section p. This decision is called the Local Decision .

Suppose the Northbound train i and Southbound train j arrived at k and k+1 at the time $\mathbf{a}_{i,k}$ and $\mathbf{a}_{j,k+1}$.

If we stop the train j and pass the i then time to wait train j

$= \mathbf{d}_j = \mathbf{a}_{i,k} + \mathbf{t}_p^{\,i} - \mathbf{a}_{j,k+1}$

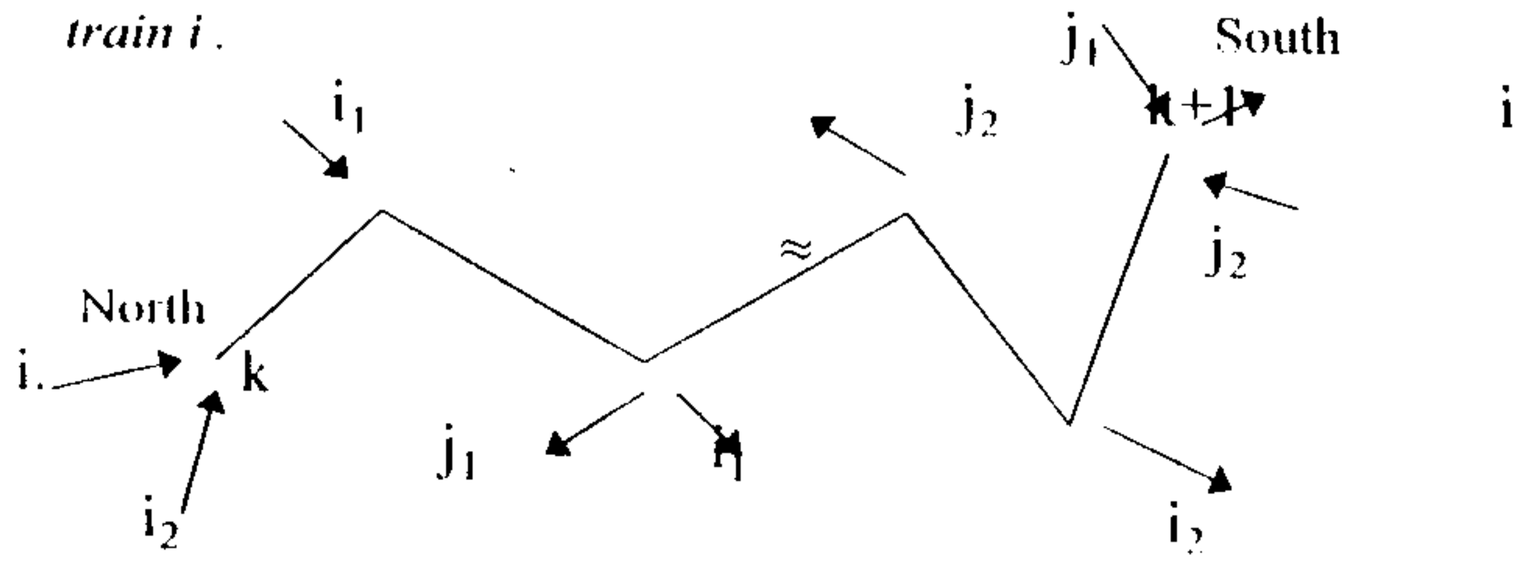
If we stop the train i and pass the train j then time to wait train i

$\mathbf{d}_i \; \mathbf{a}_{j,k+1} + \mathbf{t}_p^{\,j} - \mathbf{a}_{i,k}$

DECISION: If $d_j < d_i$ then stop the train j and pass the train i .
else stop the train i and pass the train j.
This is the local optimality criteria.

So we allow train with minimum waiting time and we can take such decision over entire network if this particular situation arises. This will give **optimal decision Globally** over entire network.

Greedy algorithm :*For Greedy algo , consider the path of a train i on the Railway network and find out the conflicting group of trains Which are partially or fully accessing the path of the train i . Consider the path of train i .*



Consider the path of a train i .Where we can see that a set of train is Northbound $S_n = \{ i_1, i_2, \ldots\ldots, \}$ *and other set is southbound* $S_s = \{j_1 j_2, \ldots.\}$.

Determination of conflicting group of trains S_n , S_s :

We shall call the train i to be incumbent train and we define the zone of the path of train i to be conflicting if other trains enter in that zone .We identify the zone as separated by label K and K + 1.

Initialisation: *Incumbent train i (Northbound) has reached K at* $a_{i,k}$:

$S_s = \{\varnothing\}$; $S_n = \{i.\}$; $T_k^m = a_{i,k}$; $T_k^f = a_{i,k}$;

$T^m_{k+1} = a_{i,k}$; $T^f_{k+1} = a_{i,k} + t^i_p$;

Step 1: *Consider the southbound trains which may arrive in the conflicting zone in the time interval* $[T_{k+1}^m , T_{k+1}^f]$.*Say this set of trains be* $C_s = \{j_1, j_2, \ldots\ldots, j_m\}$.

Let j_m = *Last train in* C_s *to arrive in the conflicting zone .*

$a_{jm,l}$ =max $\{a_{j1,l1}, a_{j2,l2}, \ldots\ldots, a_{jm,lm}\}$

Where l ,l1,l2,.....,lm are the crossings of the path of i.

At this crossings the conflicting group of trains are entering into the path of i.

$T_k^m = T_k^f$; $T_k^f = a_{jm,l} + t_p^{jm}$; and

$S_s = S_s \cup C_s = \{j_1, j_2, \ldots j_m\}$;

Where p is travelling section of j_m along the path of i.

Step 2: *Consider all northbound trains which may arrive in the conflicting zone in the time interval* $[T_m^k, T_m^f]$.

Say this set of trains be $C_n = \{i_1, i_2, \ldots\ldots, i_m\}$

Let the i_m *be the last train in this set of trains arrive in the conflicting zone .*

$a_{im,r} = \max\{a_{i1,r1}, a_{i2,r2} \ldots\ldots, a_{im,rm}\}$

$T_{k+1}^m = T_{k+1}^f$; $T_{k+1}^f = a_{im,r} + t_p^{im}$;

$C_n = C_n \cup S_n = \{i, i_1, i_2, \ldots . i_m\}$;

Algorithm :

step 1: *Consider this path of each train i and find out the set of conflicting group of trains .*

step 2 : *Take decision of entering the conflicting set of trains in a particular sequences according to the* ***options stated bellow.***

Option-1: *Let all trains in* C_n *pass without stopping and be allowed to proceed one by one until passes the conflicting zone. So*

stop j_1 *at* l_1 *and wait for time* $a_{im,l1} - a_{j1,l1}$

stop j2 at l2 and wait for time $a_{im,l2} - a_{j2,l2}$

.....

.....

and so on.

Option-2: *Let all the trains in* C_s *passes without stopping and be allowed to proceed one by one until passes the conflicting zone. So*

stop train i at k and wait for $a_{jm,k} - a_{i,k}$

stop train i_1 *at* r_1 *and wait for* $a_{jm,r1} - a_{i1,r1}$

.....

So on .

DECISION: *Compare the cost of stopping of two options. Select the cheaper one and modify the arrival time of each trains which have to wait because of this decision.*

step 3: *For every train i repeat the above procedure. Go to step1.*

Time complexity of the algo.

For each train we need the calculation (minus operation) upto 2.N and N arithmetic assignments for modification of time table . So overall complexity of the algo is $O(N.N)=O(N^2)$ *and*

Which is feasible for on-line implementation .

But here we have lost the exact Optimal solution.

Original implementation of this DSS for Railway :

The first Railway Despatching DSS implemented at Southern Railway USA between Cincinnati and Washington D.C in real life in 1982. There they have developed meet-pass plans to schedule the trains on a single line .Its basic function is to generate a meet-pass plan in order minimize the delay of all trains.. ***The plan is computer-generated by a branch and bound algorithm to minimize the delay.***

PAST WORKS AND REFERRENCES :

Optimizing the train schedule on single line :

Early attempts at modelling the single track train scheduling problem started with Szpigel(1973) who developed an integer programming model to optimise overtaking and positions , given a set of departure times and train speeds.

A more comprehensive formulation , which included the minimum headway between trains was proposed by peterson(1986).

Jovanovic (1989) formulated the problem to allow both single and double tracks.

Kraay and Mills(1991) proposed the optimization model for minimisation of fuel and waiting time or delays.

A.Higgins ,E . Kozan and Ferreira(1997) solve this problem as single line.

Optimization of train scheduling on a Network:

A.I .Mees (1991) modelled the single line rail as a network structure.

Each track segments made up of arcs ,separated by nodes.The resulting formulation is a multi-commodity network flow which is converted to the lagrange multiplier problem for solution.

My work: ***Among the above references A.I .Mees (1991) tried to schedule trains on network where he formulated the problem as a multi-commodity network flow. So to find global solution we have to go through exhaustive checking which is not possible to imple ment on line for such a real time system. Actually scheduling trains on a network is a NP-Hard problem.***

So I have formulated the problem in two different ways and modelled a heuristic which is not optimal but give a near optimal solution with online implementation capability .