Multiagent Based Algorithmic Approach for Fast Response in Railway Disaster Handling

Poulami Dalapati
Department of IT
NIT Durgapur
India
dalapati89@gmail.com

Arambam James Singh
School of Information Systems
Singapore Management University
Singapore
ajsingh@smu.edu.sg

Animesh Dutta
Department of IT
NIT Durgapur
India
animeshnit@gmail.com

***Abstract*—Disaster management in railway network is an important issue. It requires to minimize negative impact and also fast, efficient recovery from the disturbances. The main challenge here is that, the effect of inconvenience spreads out very fast in time and space. It takes noticeable amount of time to get back everything in the previous situation. This paper proposes a multi agent based algorithmic approach for disaster handling in Railway Network. This takes care of fast response to get total number of affected trains in a fast and efficient manner. We propose few algorithms to handle this situation and simulate it using JADE (Java Agent Development Framework) platform. Finally we take a case study and compare our proposed method with an existing manual technique.**

***Keywords*—Multi agent system; Disaster management; Optimization; Distributed systems.**

I. Introduction

Real world problems very often have higher complexity levels. Usually it is possible to decompose them into different complex units with certain well defined functionality. With this notion, various traffic environments, their control and scheduling have been focused by number of researchers [2], [3], [4], [16]. In particular, the need for some degree of autonomy, to enable components to respond dynamically changing circumstances while trying to achieve over-arching objectives becomes fundamental. During the last years, agent based approaches to handle railway network [6], [1], [7] have shown that they are able to capture necessary details at entity level as well as to reproduce relevant realistic phenomena. In practice any railway network is broadly distributed all over the country. All the time large number of trains are in circulation. Every train has a particular arrival and departure time and a specific route of journey. This is generally published in timetable which is known to the passenger. These details are also monitored by the station authority. The problem here with such centralized system is that the control is on only one system (central server), which is very vulnerable in practice. Anytime any disaster can happen to the network, due to natural calamities, technical fault, signaling error or due to sabotage. These causes deviation from scheduled operation and hampers number of trains. The severeness of the disruption is measured by the number of these affected trains. But due to complex infrastructural network the effect of one disaster easily gets distributed into some other parts also. This is known as knock-on effect [10].

Avgoustinos Filippoupolitis [8] in his paper has proposed a fully distributed system, which takes into account the spatial characteristics of hazard propagation. Their system is composed of number of decision nodes (DN). When a change occurs in environment, the DN close to the respective location detects the event. In order to inform the rest of the DNs regarding this change, the system floods the information. S. Cicerone et. al. [9] have given a new concept for planning under disturbance, dividing it into two phases: Strategic Planning Phase and Operational Phase. But the drawback is, typically there is not only one place of disruption. The consequence may appear one after another. Their model do not handle this. Victor Sanchez-Anguix et. al. [13] in their paper have presented an agent-based add-on for the Social-Net Tourism Recommender System that uses information extraction and natural language processing techniques in order to automatically extract and classify information. Aitor Mata and Beln Prez and Juan M. Corchado [14] have given an idea about Organization Based System for Forest Fires Forecasting (OBSFFF), which is able to generate a prediction about the evolution in certain areas. This is based on the Case-Based Reasoning methodology, which uses historical data to create new solutions to current problems. The system employs a distributed multi-agent architecture so that the main components of the system can be remotely accessed. Some researchers in their paper [15] have presented a self-adaptive cooperation model to achieve collaborative goals in crisis management scenarios. Though there are large number of works on disaster management in various traffic related problem [5], [11], [12], but a very few paper properly handle the same in an optimized and autonomous way. So, to handle such challenging scenario in efficient manner, where network is broad and complex, we use multiagent based approach which is autonomous and inherently distributed in nature.

II. System model

As the Railway System is concerned, the problem of scheduling a new train in an existing timetable can suitably modeled through discrete mathematics where we can represent the Railway System as a graph $G =< V, E >$. We put Railway Network (RN) as a pair of a graph (G) and an agency (A), $RN =< G, A >$. Again, $G =< V, E >$, where V is set of vertices and E is set of edges. In our system, V represents a station S and E represents a track between two stations. In general, a station can have more than one platforms and trains can stop here.

A. Notation

Indices and Parameters:

i	Station index	j	Train index
l	Track index	k	Platform index
n	Number of stations	m	Number of trains
p	Number of platforms at each station	o_{ji}^{AT}	Arrival time of train j at i^{th} station in scheduled timetable
o_{ji}^{DT}	Departure time of train j from station i in scheduled timetable	δ_{ji}	Delay of train j at station i
		t	Time instant
δ_{Th}	Threshold value for delay of all trains	o_{ji}^{J}	Journey time of train j in original timetable
a	Agent index	q	Number of agents

Decision Variables:

x_{ji}^{AT}	Arrival time of train j at station i due to disaster
x_{ji}^{DT}	Departure time of train j from station i due to disaster
t_D	Time of disaster
t_R	Time to recover with the density function $\phi(x)$, where, $x \in [\tau_1, \tau_2]$

So, from the above notations,$V = \{v_i | i \in [1, n]\}$ and $v_i = s_i$ means vertex is a station. There exist number of trains (T) which are already in circulation, $T = \{T_j | j \in [1, m]\}$. The agency is composed of number of agents as, $A = \{A_a | a \in [1, q]\}$. Each station and train is associated with an agent. SA and TA denote the station agent and train agent respectively, where $s_i \in S$ with $sa_a \in SA$ and $T_j \in T$ with $Ta_a \in TA$.

B. Properties of the System

Properties of the Railway System can be expressed by fluents (functions whose values change over time) and by persistent functions (whose values do not change over time).
Persistent Function of the RN Physical Network:
$max\_capacity : S \rightarrow N$
$max\_capacity(s_i) = n$ iff station s_i can host at most n trains. This information is only available to station agent.
Fluents Representing RN's Features:
$current\_capacity : N \times S \rightarrow N$
$current\_capacity_t(s_i) = n$ iff station s_i has room for n more trains at time t. This information is only available to station agent sa_i.
$running\_on : N \times E \rightarrow T$
$running\_on_T(e(i, j, k)) = T_n, \dots, T_1$ iff $T_n, \dots, T_1$ are the trains currently running on edge $e(i, j)$, where,T_1 is the first train that left the station and T_n is the last who is following previous trains maintaining a critical distance. This information is only available to the station agent in charge of the station from which the edge exits.
Persistent Functions of Train Schedule:
$route : T \times N \rightarrow S$
$route(t_i, i_r) = S_i$ iff the ${i_r}^th$ station in $t's$ scheduled path is $S_i(i_r$ ranges between 1 and the maximum number of stations that t_i is expected to traverse).
$scheduled\_arrival : T \times S \rightarrow N$
scheduled arrival $(T_j, S_i) = n$ iff n is the time instant when T_j should arrive in S_i according to the planned schedule.
$max\_speed : T \rightarrow R$
$max\_speed(T_j) = r$ iff the maximum allowed speed for T_j is r, expressed in some suitable speed unit measure.
$stop\_value : T \times S \rightarrow Bool$
$stop\_value(T_j, S_i)$ is true if T_j will stop in station S_i and false otherwise.
Fluents of Train's Features:
$current\_position : N \rightarrow (V \cup E) \times R$
current position $(T_j) = (v/e, d)$ iff T_j is currently either on vertex v, in which case r is 0, or on edge e, in which case d is the distance from the edge origin expressed in a suitable distance measure unit.

C. Assumption

- There is only one track connecting two neighboring stations and no crossover in-between.
- There is at least one platform at each station.
- Station Agent can communicate with incoming and outgoing trains and with neighboring stations.
- Train agent can communicate with station agents only.
- All the trains begin and end their journey at stations.
- All the trains move at a constant speed (generally with its average speed).

III. Metric Definition

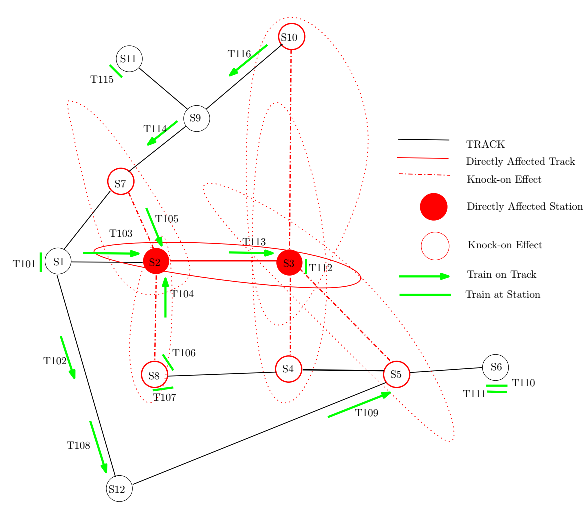

Fig. 1: Representation of Severeness and Knock-on Effect

- *Severeness* ($|T_{AFF_o}|$)*:* It is described as how many trains will be affected. Severeness of any disruption is not easily assessed.

$$T_{AFF_o} \subseteq T$$

- *Knock-on effect* (KoE)*:* A very common problem in railway is that, due to strong interdependencies in RN, and due to cost efficient resource schedules, disruptions are very likely to spread over the network. The key to good performance of railways is to limit the knock on effect and thereby limit the impact of single disruptions. We represent it in percentage. It can be defined as,

$$KoE = \frac{|T_{AFF_o}|}{|T_I|} \times 100 \tag{1}$$

where T_I is the set of affected trains in the RN.

IV. Problem Formulation

There are n number of stations and m number of trains, i.e. $S = \{S_i | i \in [1, n]\}$ and $T = \{T_j \mid j \in [1, m]\}$. We are assuming that every station S_i has a particular number of platforms P, $(0 < P \leq p)$ and there is only one track connecting S_i to $S_{i'}$, where $i, i^{'} \in [i, n]$ and $i \neq i^{'}$. These are called the required resources $R(T_j)$ for train T_j at any time instant t. Every train T_j has a predefined route of its journey $Route(T_j)$ from source and destination. This route is defined by sequence of stations S_i where $S_i \in V$ from the system model of RN. Every station has its own database with all the details of its neighboring stations and incoming-outgoing trains, their arrival and departure and stop time. They also have their own updation mechanism on time or trigger basis, i.e. whenever any changes occur either for schedule or disastrous phenomena it updates its database accordingly.

Initially $TRACK(S_i, S_{i'}) = 1$ if two stations S_i and $S_{i'}$ are adjacent to each other and zero otherwise. Now if any link gets destroyed due to disaster, the end stations of this particular edge updates its database as $TRACK(S_i, S_{i'}) = -1$. This is denoted as $TRACK_D$.

Let us now assume that $\exists\, T_j$ where $j \in [1, m]$ who have this $TRACK_D$ in their scheduled route. These T_j are directly affected trains. So, initially $T_I = T_j | j \in [1, m]$, where T_I is the set of affected trains in the RN.

Both the stations S_i and $S_{i'}$ have an idea about the recovery time t_R of the disaster as per prior experience. S_i and $S_{i'}$ send this t_R as a message to all its neighbors $S_{(i+1)}$, $S_{(i'+1)}$, where $i, i^{'} \in [1, n]$, from where some trains are scheduled to come to these stations(either S_i or $S_{i'}$ or both). Now $S_{(i+1)}$ and $S_{(i'+1)}$ will check for the fastest, say, T_{j_F} which is to arrive in S_i or $S_{i'}$ within that recovery time t_R. i.e.

$$o_{ji}^{AT}(T_{j_F}, S_i) = o_{ji}^{AT}(T_{j_F}, S_{i'}) = t_D + t_R \tag{2}$$

or

$$o_{ji}^{DT}(T_{j_F}, S_i) = o_{ji}^{DT}(T_{j_F}, S_{i'}) = t_D + t_R \tag{3}$$

Then set of affected trains will be updates as

$$T_{AFF} = T_I \cap T_{j_F} \tag{4}$$

We then check for other trains at that particular route which may arrive to S_i or $S_{i'}$ within t_R. From this we will get the set of trains which are *Directly Affected Trains* due to disaster. Another consequence of this is the other trains which follow this trains those may or may not get affected. For the directly affected trains if we reschedule its path to avoid disturbed route, there may arise a case where they conflict with other trains in the network in terms of resources. As the priority of trains also matters so for some trains T_j, where $j \in [1, m] \backslash T_{AFF}$, rescheduling may hamper scheduled route of the other. Then these train will also be added with T_{AFF}. So, this will be our total optimal set of affected trains T_{AFF_O}.

A. Proposed Algorithm

To formulate our methodology we propose three algorithms here. Algorithm 1. gives the idea about *database updation of neighboring stations before and after disaster*. Algorithm 2. determines the *directly affected trains* whereas, *final optimized number of trains* are determined by Algorithm 3. as discussed previously in section IV.

Algorithm 1 : Updation of Neighbourhood Stations

for $\forall S_i \in S$ **do**
    if $S_{i'} = Adj(S_i)$ **then**
        $TRACK(S_i, S_{i'}) = 1$
        if $TRACK(S_i, S_{i'}) = TRACK_D$ **then**
            $TRACK(S_i, S_{i'}) = -1$
        end if
    end if
    if $S_{i'} = NAdj(S_i)$ **then**
        $TRACK(S_i, S_{i'}) = 0$
    end if
end for

Algorithm 2 : Getting the Directly Affected Trains

while $TRACK(S_i, S_{i'}) = -1$ **do**
    for $\forall T_j \in T$ **do**
        if $(Route(T_j) = S_i) \vee (Route(T_j) = S_{i'})$ **then**
            $T_I = T_j$
        end if
    end for
    for $\forall T_I \in T$ **do**
        if $((o_{ji}^{AT}(T_{j_F}, S_i) = t_D + t_R) \vee (o_{ji}^{AT}(T_{j_F}, S_{i'}) = t_D + t_R))$ **then**
            $T_{AFF} = T_I \cap \mathrm{T}_{j_F}$
        end if
        if $(o_{ji}^{DT}(T_{j_F}, S_i) = t_D + t_R) \vee (o_{ji}^{DT}(T_{j_F}, S_{i'}) = t_D + t_R))$ **then**
            $T_{AFF} = T_I \cap \mathrm{T}_{j_F}$
        end if
    end for
end while

Algorithm 3 : Getting Total Optimized Number of Affected Trains

for $\forall T_j \in T_{AFF}$ **do**
    $CALL(Reschedule(T_j(S_i, S_{i'})))$
    $T_j = T - T_{AFF}$
    if $R(T_j)_t = R(T_{j'})_t$ **then**
        $T_{AFF_O} = T_j + T_{j'}$
    end if
end for

B. Indian Railway System : A Case Study

In Indian Railway System, divided into 17 main zones in total, which are again divided into number of divisions and sub−divisions, a large number of trains are always in circulation. However, unexpected events during the operation process may cause disturbances. So, railway authority needs to manage the whole network the trains with the help of the real-time traffic management system to minimize the negative effects arising from those disturbances. What makes it even more challenging is that depending on the recovery time for disturbance they have to take the decision that which trains are being affected. Moreover traditionally, in Indian Railway System, the operations are done manually. So, it takes huge time to resolve everything (takes few hours or more). Whereas in our case as autonomous agents communicate with each other and collaborate in distributed fashion, it takes much less time to overcome the scenario (few seconds only) as shown in Fig. 4.

V. Experiments and Results

In our system we use two metrics Knock-on Effect [10] and Severeness [10]. The effect of disaster in Railway Network

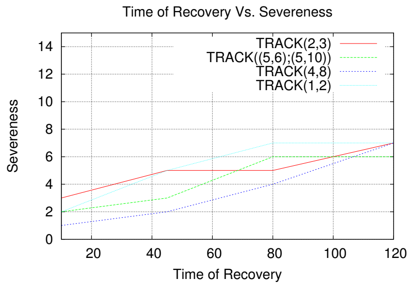

Fig. 2: Severeness for Disaster in different Track with respect to Time of Recovery

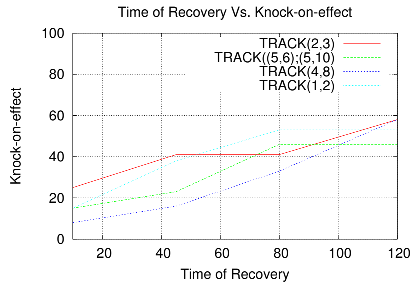

Fig. 3: Knock-on-effect for Disaster in different Track with respect to Time of Recovery

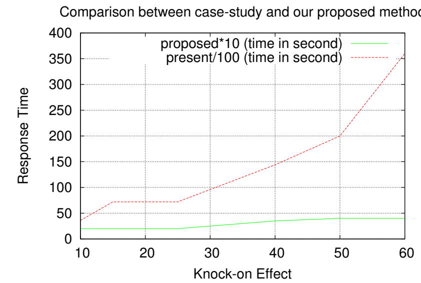

Fig. 4: Comparison between case-study and our proposed methodology

is measured using the expression of metrics given in section III. We consider ten stations with defined paths along with fifteen trains in the network as represented in Fig. 1. In order to simulate our proposed methodology i.e. to schedule a new train in an existing timetable with delay optimization, we use JADE [17]. We run our algorithm for different TRACK which faces inconvenience due to disruption. We also vary recovery time to elaborate the effect. Fig. 2. represents the graph of severeness. Here we plot severeness with respect to time of recovery for each track. In the second graph, i.e. Fig. 3., we plot knock-on effect with respect to time of recovery for each track. It is plotted in percentage basis for convenience. Fig. 4. shows the comparison between existing solution from case study and our proposed solution, where the deviation of the proposed method is noticeable in positive sense. It is also clearly seen that our scheduling algorithm generates faster response.

VI. Conclusion

Few algorithms are proposed here which handle disastrous situation in railway network. The simulation is also done to make sure the effectiveness of those algorithms. The use of agent technology makes the system more efficient. Future work focuses on probabilistic nature of disaster and its recovery through rescheduling.

Acknowledgment

The authors are gateful for partial funding provided by DST and MHRD.

References

[1] David Handford and Alex Rogers, "Modelling Driver Interdependent Behaviour in Agent-Based Traffic Simulations for Disaster Management", *Advances on Practical Applications of Agents and Multiagent Systems, AISC 88, Springer- Verlag Berlin Heidelberg*, pp. 163-172, 2011.

[2] Maksims Fiosins, Jelena Fiosina, Jorg P. Muller, and Jana Gormer, "Agent-Based Integrated Decision Making for Autonomous Vehicles in Urban Traffic", *Advances on Practical Applications of Agents and Multiagent Systems, AISC 88, Springer-Verlag Berlin Heidelberg*, pp. 173-178, 2011.

[3] Neila Bhouri, Flavien Balbo, and Suzanne Pinson, "Towards Urban Traffic Regulation Using a Multi-Agent System", *Advances on Practical Applications of Agents and Multiagent Systems, AISC 88, Springer-Verlag Berlin Heidelberg*, pp. 179-188, 2011.

[4] Jarosaw Kozlak, Sebastian Pisarski, and Magorzata Zabinska, "Multi-agent Models for Transportation Problems with Different Strategies of Environment Information Propagation", *PAAMS 2013, LNAI 7879, Springer-Verlag Berlin Heidelberg*, pp. 145-156, 2013.

[5] Mohamed Haitam Laarabi, Claudio Roncoli, Roberto Sacile, Azedine Boulmakoul, and Emmanuel Garbolino, "An Overview of a Multiagent-Based Simulation System for Dynamic Management of Risk Related to Dangerous Goods Transport", *IEEE*, 2013.

[6] Ana L. C. Bazzan and Franziska Klugl, "A review on agent-based technology for traffic and transportation", *The Knowledge Engineering Review, Cambridge University Press*, pp. 1-29, 2013.

[7] Sbastien Fournier, Alain Ferrarini, and Aline Cauvin , "A Co-operative AgentBased Scheduling Repair Method for Managing Disruptions in Complex Organisations" , *IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises*, pp. 47-52, 2011.

[8] Avgoustinos Filippoupolitis , "An adaptive system for movement decision support in building evacuation", *ALADDIN (Autonomous Learning Agents for Decentralised Data and Information Networks) project*, 2011.

[9] S. Cicerone, G. Di Stefano, M. Schachtebeck, and A. Schobel , "Multi-Stage Recovery Robustness for Optimization Problems: a new Concept for Planning under Disturbances", *project ARRIVAL*, 2009.

[10] Julie Jespersen-Groth, Daniel Potthoff, Jens Clausen, Dennis Huisman, Leo Kroon, Gabor Maroti and Morten Nyhave Nielsen, "Disruption Management in Passenger Railway Transportation", *project ARRIVAL*, 2007.

[11] Gilbert Laporte, Juan A. Mesa and Federico Perea , "A Game Theory Framework for the Robust Transportation Network Design Problem", *project ARRIVAL*, 2008.

[12] Marin A., Mesa J.A., and Perea F., "Integrating Robust Network Design and Line Planning under Failures", *project ARRIVAL*, 2009.

[13] Victor Sanchez-Anguix, Sergio Esparcia, Estefani Argente, Ana Garcia-Fornes and Vicente Julian, "Collaborative Information Extraction for Adaptive Recommendations in a Multiagent Tourism Recommender System" , *Advances in PAAMS, AISC 70*, pp. 3540, 2010.

[14] Aitor Mata, Beln Prez, and Juan M. Corchado , "Forest Fires Prediction by an Organization Based System", *Advances in PAAMS, AISC 70*, pp. 135-144, 2010.

[15] Jean-Pierre Georg, Marie-Pierre Gleizes, Francisco J. Garijo, Victor Nol and Jean-Paul Arcangeli , "Self-adaptive Coordination for Robot Teams Accomplishing Critical Activities" , *Advances in PAAMS, AISC 70*, pp. 145-150, 2010.

[16] Dalapati P., Singh A.J., Dutta A., and Bhattacharya S., "Multi agent based railway scheduling and optimization", *TENCON - 2014 IEEE Region 10 Conference*, pp. 1-6, 2014.

[17] http://jade.tilab.com/