Exploiting MAS-based Simulation to Improve the Indian Railways' Efficiency

Supriyo Ghosh[1], Animesh Dutta[1], Viviana Mascardi[2], Daniela Briola[2]

[1]Department of Information Technology
National Institute of Technology
Durgapur 713209, INDIA
`ghosh.supriyo.cse@gmail.com, animesh.dutta@it.nitdgp.ac.in`
[2]Department of Informatics, Bioengineering, Robotics and System Engineering
University of Genova
Via Dodecaneso 35, 16146, Genova, ITALY
{`viviana.mascardi,daniela.briola`}`@unige.it`

Abstract. Despite being one of the world's largest railways networks, with a daily transportation of over 25 million passengers and 2.8 million tons of freight, the Indian Railways perform their signaling, traffic management and trains scheduling activities in a completely manual way.
The lack of automation causes not only significant delays, but also frequent collisions where passengers die or remain seriously injured.
This paper describes how we modeled the Indian Railways system as a MAS and the results of the simulations run using NetLogo on real data retrieved from the Indian Railways dataset. The simulated system is – under some assumptions – collision-free and, thanks to the integration of the MAX-SUM algorithm, minimizes the individual train delay.

1 Introduction

Indian Railways (IR) is one of the world's largest railways networks comprising 115,000 km of track over a route of 65,000 km and 7,500 stations. In 2012, it transported over 25 million passengers daily (over 9 billion on an annual basis). In 2011, the IR carried over 8,900 million passengers annually and more than 24 million passengers daily (roughly half of which were suburban passengers) and 2.8 million tons of freight daily.

In spite of this huge traffic of people and goods, signaling, traffic management and trains scheduling are carried out by the IR staff who communicates and coordinates in a completely manual way. This explains the need of 1.4 million of employees[1] and the impressive amount of collisions that take place[2], almost always causing deaths and serious injuries.

The problems of the IR manually controlled railway system include:

[1] Data referred to 2011: `http://www.indianrailways.gov.in/railwayboard/uploads/directorate/stat_econ/Stat_0910/Year%20Book%202009-10-Sml_size_English.pdf`, accessed on May, 2013.

[2] `http://en.wikipedia.org/wiki/List_of_Indian_rail_incidents`, accessed on May, 2013.

- The control room of a station can control a train only if it is within the range of 365 meters, which is an almost limited range.
- The train driver has to look out the train for the signal board even while driving during night and in foggy or bad weather.
- Computing the exact distance of the train from a signal post is difficult, and controlling the train's speed in order to ensure that it will stop in time if the next signal will require so, is also hard.
- The signaling system is fixed and signals cannot convey information of speed restrictions; such restrictions are communicated from the crew operating on a train who realizes the need of a speed restriction, to the control room in the next station, by writing a report when they reach the station.
- Tracks management is performed by means of high frequency walkie-talkie communication between the control room staff and the train driver or crew members; being an entirely human-based process, this activity is subject to many errors.

Besides these problems, trains are scheduled manually and the resulting timetable does not assume that all trains operate on full speed, which decreases the throughput of the system in terms of resource utilization; also, the timetable includes redundant waiting time for some trains, creating unnecessary delays.

A specification stating the functional requirements of a Centralized Traffic Control (CTC)/Traffic Management System (TMS) has been recently released by the Ministry of Railways[3]. The CTC/TMS should carry out the following two categories of functions:

1. centralized operations of signaling for a large area encompassing multiple interlocked stations;
2. centralized real time monitoring of trains traffic for enabling efficient decisions making for traffic control of large areas.

Although the CTC/TMS will definitely improve the current IR situation, a centralized approach does not seem to be the most suitable one for a network where about 10,000 trains run daily and must coordinate among themselves and with the stations' control rooms.

As observed in [2], Agent-Based Models (ABMs) are particularly suited to tackle situations characterized by the presence of a high number of autonomous entities whose behavior (actions and interactions) determines in a non-trivial way the evolution of the overall system. ABMs support the study and analysis of topics like decentralized decisions making, local-global interactions, self-organization, emergence and effects of heterogeneity in the simulated system.

Besides modeling the system, researchers are of course interested in simulating it, hence leading to the Agent-Based Modeling and Simulation (ABMS) approach which is strongly inspired by social simulation and is considered by some scientists as "a third way of doing science" [1].

[3] `http://www.rdso.indianrailways.gov.in/works/uploads/File/TMS%20-%20DRAFT%20SPEC.pdf`, accessed on May, 2013.

The motivation for this paper is to move a concrete step towards the development of a fully automated and decentralized IR to overcome collisions due to manual errors and to improve trains punctuality, hence allowing IR to provide a better Quality of Services (QoS) to the customers.

The chosen approach is that of Agent-Based Modeling and Simulation, which perfectly fits the IR domain: in IR we can identify different entities playing different roles, that are autonomous, distributed, concurrent, and interact via asynchronous communication using a high level communication language. Modeling such entities as agents is a very natural choice: simulating the resulting Multiagent System (MAS) to check whether the model is realistic, and to evaluate the effects of interventions in the model itself, can give important hints on how improving the system's efficiency.

In this paper we describe how we modeled the IR system as a MAS and its simulation and results using NetLogo[4]. The IR MAS was already discussed in [12], that describes the negotiation algorithms in detail and the delay optimization based on the MAX-SUM algorithm [16, 17], and in [3], that describes the agents' features in terms of beliefs, desires and intentions.

With respect to the two papers above, the system's model given in this paper is different being based on concepts inspired by dynamic logic (fluent and persistent functions), and the description of the NetLogo implementation and of its results is original. The simulated system is collision-free under specific constraints and, thanks to the integration of the MAX-SUM algorithm, minimizes the individual train delay.

The paper is structured as follow: Section 2 shortly discusses the exploitation of MASs and ABMS techniques in the railway domain; Section 3 describes our new model of the IR system; Section 4 presents six problems with their solution in our MAS; Section 5 reports the results of the experiments executed with NetLogo and lastly Section 6 concludes and describes the future work.

2 Related Work

As observed in a recent survey [8], railway dispatching or scheduling has been usually modeled using classical technologies, such as operational research and constraint programming. These techniques are suitable to model static situations with complete information, but they lack the ability to cope with the dynamics and uncertainty of real railway traffic management. In order to overcome the limitations of traditional centralized and monolithic approaches, many researchers from the multiagent community started working on this domain. Fischer et al. [10] present a MAS that models a society of transportation companies whose goal is to deliver a set of dynamically issued orders satisfying some given cost and time constraints. The proposed conceptual system together with a further study [11] led to two practical applications, i.e., the TeleTruck system [13] and a railroad scheduling system [14]. Cuppari et al. [9] employ a logic

[4] `http://ccl.northwestern.edu/netlogo/`, accessed on May, 2013.

programming-based environment to prototype a MAS for the management of freight train traffic. The work presented in [4] uses a multiagent approach to the scheduling system for train coupling and sharing. The system is incremental, takes incomplete task specifications into account and generates an initial plan using the contract net protocol: then the post-optimization of the initial solution is achieved by means of the simulated trading protocol.

The agent approach has also been used for other railway applications in the recent years. To model railway access negotiation, Tsang and Ho [18] employ a multiagent approach in which a train services provider (TSP) and an infrastructure provider (IP) are represented by individual software agents. An IP-TSP transaction is simulated by a negotiation protocol. For flexible trains traffic control and conflicts avoidance, Proenca and Oliveira [15] adopt a multiagent railway control system made up of two subsystems: control and learning. The "Control" subsystem is responsible for traffic management and guidance in the network. The "Learning" subsystem has the objective of analyzing the past accumulated situation descriptions and inferring rules that anticipate trains conflicts and improve traffic-control processes. In the set of papers [5–7], Briola et al. describe a multiagent system called FYPA that is used to dynamically manage the allocation of trains on railway tracks inside a station: every train and every railway segment is associated with a dedicated agent, and the real time reallocation of trains is made following a complex negotiation protocol. The system is currently part of the Ansaldo STS (Italy) applications portfolio.

3 Agent-Based Model of the IR System

Model of the IR physical network

The IR physical network (*IRN*) can be suitably modeled as a mixed multigraph, namely a graph where some arcs are directed and some are undirected ("mixed graph"), and where multiple arcs with the same source and target vertices are allowed ("multigraph").

For sake of clarity, we will present *IRN* as if it were a direct multigraph. Undirected arcs impact only in a couple of points of our presentation, where we will deal with them explicitly.

Vertices in V can model either stations (S) or junctions (J), and stations' identifiers are disjoint from junction' ones: $V = S \cup J \wedge S \cap J = \emptyset$. The difference between a station and a junction is that only one train at a time can cross a junction and it cannot stop over it, whereas a station can hold many trains (usually the station's capacity is ≥ 1) and trains can stop on it. Vertices will be identified by v throughout the paper. If needed, we will use subscripts and we will explicitly state whether v_i models a station or a junction by writing s_i and j_k, respectively (since stations and junctions are disjoint, if $v_i = s_i$ represents a station and $v_k = j_k$ represents a junction, then $i \neq k$). Edges E are identified by $e_{(i,j,k)}$, where i is the index of the source vertex v_i, j is the index of the destination vertex v_j, and k ranges on $1, 2..., e$ where e is the total number of edges directed from v_i to v_j. If there is no need to give details on an edge's source, destination and number, we will use e to represent it.

Agents

Each junction and station in *IRN* has an agent associated with it. We identify the station agent associated with s_i, where $s_i \in S$, with $sa_i \in SA$ (the set of station agents), and the junction agent associated with j_k, where $j_k \in J$, with $ja_k \in JA$ (the set of junction agents).

Each train $t \in T$ is associated with a train agent $ta \in TA$ as soon as it is created in the IR system. Train agents are identified by ta subscripted, if needed, by a natural number i ranging from 1 to the maximum number of trains in *IRN*.

By using the same subscript for the agent and for the physical element of the IR system it is associated with, we can leave the bijective function that maps physical elements to agents implicit: throughout the paper, station agent sa_i will always be associated with station s_i, junction agent ja_i will be associated with junction j_i, and train agent ta_i will be associated with train t_i.

Fluent and persistent properties of the IR system

Properties of the IR system can be expressed by fluents (functions whose returned value changes over time), and by persistent functions which state structural properties of the system, not subject to changes over time. We assume that time is discrete and is represented by the number of time units that elapsed from a conventional starting instant. We express the time argument of fluents as a subscript of the function name for readability.

For each function, we write in square brackets and bold font the name of the only agent in the system which knows the function's value on the given arguments. This is a relevant aspect for emphasizing that data are really decentralized, and hence computation can be decentralized as well.

Besides all the standard operations on multigraphs (adjacency list, degree of a vertex, and so on), the only persistent function representing a relevant aspect of the IR physical network is

- $\mathit{max\_capacity} : S \to \mathbb{N}$
  $\mathit{max\_capacity}(s_i)[\mathbf{sa_i}] = n$ iff station s_i can host at most n trains. This information is only available to station agent sa_i.

Fluents representing IR's features are

- $\mathit{current\_capacity} : \mathbb{N} \times S \to \mathbb{N}$
  $\mathit{current\_capacity}_t(s_i)[\mathbf{sa_i}] = n$ iff station s_i has room for n more trains at time t. This information is only available to station agent sa_i.
- $\mathit{running\_on} : \mathbb{N} \times E \to T^*$
  $\mathit{running\_on}_t(e_{(i,j,k)})[\mathbf{sa_i}] = t_n ... t_1$ iff $t_n, ..., t_1$ are the trains currently running on edge $e_{(i,j,k)}$, in the order they appear in the string $t_n ... t_1$ (namely, t_1 is the first train that left the station and it is the most distant from it, whereas

t_n is the last train that left it, and it is the closest). This information is only available to the station agent sa_i in charge of the station from which the edge exits.

- $\mathit{is\_free} : \mathbb{N} \times J \to \mathit{Bool}$
  $\mathit{is\_free}_t(j_i)[\mathbf{ja_i}]$ is true if junction j_i is free at time t, and false otherwise. This information is only available to junction agent ja_i.

As far as trains are concerned, persistent functions representing their official schedule and technical features are described below. Apart from the train's schedule, that is public, the only agent that possesses information about train t_i, is ta_i.

- $\mathit{route} : T \times \mathbb{N} \to S$
  $\mathit{route}(t_i, \mathit{ind})[\mathbf{ta_i}] = s$ iff the ind-th station in t_i's scheduled path is s (ind ranges between 1 and the maximum number of stations that t_i is expected to traverse).
- $\mathit{scheduled\_arrival} : T \times S \to \mathbb{N}$
  $\mathit{scheduled\_arrival}(t_i, s_j)[\mathbf{sa_j}] = n$ iff n is the time instant when t_i should arrive in s_j according to the planned schedule.
- $\mathit{max\_speed} : T \to \mathbb{R}$
  $\mathit{max\_speed}(t_i)[\mathbf{ta_i}] = r$ iff the maximum allowed speed for t_i is r, expressed in some suitable speed unit measure.
- $\mathit{stop\_value} : T \times S \to \mathit{Bool}$
  $\mathit{stop\_value}(t_i, s)[\mathbf{ta_i}]$ is true if t_i will stop in station s and false otherwise.

The fluents modeling train's features are

- $\mathit{current\_position} : \mathbb{N} \times T \to (V \cup E) \times \mathbb{R}$
  $\mathit{current\_position}_t(t_i)[\mathbf{ta_i}] = (v/e, r)$ iff t_i is currently either on vertex v, in which case r is 0, or on edge e, in which case r is the distance from the edge origin expressed in a suitable distance measure unit.
- $\mathit{current\_speed} : \mathbb{N} \times T \to \mathbb{R}$
  $\mathit{current\_speed}_t(t_i)[\mathbf{ta_i}] = r$ iff at time t, t_i is running at speed r, expressed in a suitable speed measure unit.

Finally, two fluents involve both trains and stations:

- $\mathit{expected\_waiting\_time} : \mathbb{N} \times T \times S \to \mathbb{N}$
  $\mathit{expected\_waiting\_time}_t(t_i, s_j)[\mathbf{ta_i}, \mathbf{sa_j}] = n$ iff n is the amount of time units that t_i should wait in station s_j.
- $\mathit{expected\_arrival} : \mathbb{N} \times T \times S \to \mathbb{N}$
  $\mathit{expected\_arrival}_t(t_i, s_j)[\mathbf{ta_i}, \mathbf{sa_j}] = n$ iff the time instant when t_i is expected to actually arrive in s_j, is n.

4 Tackled Problems and Proposed Solutions

In order to decide which train should access a resource first (the right to cross a station or a junction, the right to move on an edge), we introduce the notion of payoff of a train t_i that wants to enter a station s_j at time t.

$$\begin{aligned} &\mathit{payoff} : \mathbb{N} \times T \times S \to \mathbb{R} \\ &\mathit{payoff}_t(t_i, s_j) = \mathit{deviation\_from\_schedule}_t(t_i, s_j) * \mathit{deviation\_cost} + \\ &\qquad\qquad \mathit{expected\_waiting\_time}_t(t_i, s_j) * \mathit{waiting\_cost} \end{aligned}$$

where

- $\mathit{deviation\_from\_schedule}_t(t_i, s_j) =$ $|\mathit{expected\_arrival}_t(t_i, s_j) - \mathit{scheduled\_arrival}(t_i, s_j)|$ is the absolute value of the difference between the time when t_i should enter s_j according to its scheduled timetable, and the time when it is actually expected to enter it.
- *deviation_cost* is the penalty for a 1 time unit deviation from the scheduled timetable.
- $\mathit{expected\_waiting\_time}_t(t_i, s_j)$ is a value that we compute by exploiting the MAX-SUM algorithm [16, 17], as described in [12].
- *waiting_cost* is the penalty for a 1 time unit expected waiting in a station.

In our model, $\mathit{waiting\_cost} > \mathit{deviation\_cost}$, and both can be established by the system's administrator.

The same notion of payoff can be defined for junctions, instead of stations. Due to space limitations, we cannot give here more details about the MAX-SUM algorithm. We only point out that this algorithm uses the current delay of a train to increment the train's payoff: in this way, the payoff increases with the increase of the delay, and train starvation is avoided because sooner or later the train will reach a payoff higher than that of the other conflicting trains, getting the right to access a shared resource. Payoffs are used to resolve conflicts on shared resources according to the following criteria:

1. if more trains want to access the same resource at the same time, the resource will be assigned to the train with higher payoff;
2. a station agent associated with a station with few available platforms, can decide not to authorize a train to enter the station if its payoff is lower than the payoff of other trains that want to enter the station.

Now we can discuss the IR problems we faced and how we solved them.

Problem 1: if two trains are running on the same directed edge (obviously in the same direction) and their distance becomes lower than a critical distance cd, then a collision might take place.
When a train t_i leaves a station s_j on edge e, the station agent sa_j tells the train agent ta_i the identifier k of the train running ahead of it on edge e. In this way, ta_i can contact ta_k and coordinate with it in order to avoid collisions. The coordination is based on a simple rule: if there is a risk of collision computed based on the current speed and position of both trains, and the train which is ahead can accelerate (it is not running at its maximum allowed speed), than it does accelerate. Otherwise, the train which is behind decelerates. When the train which is ahead reaches the next station or leaves that edge, it informs both the station from which it departed and the train which is behind it. The station agent can remove that train from the list of trains running on e by updating the value of $\mathit{running\_on}_t(e)$.

The proposed solution works only if the change of speed of one train removes the risk of collision with the train which is ahead (resp. behind) without introducing a collision risk with the train that is behind (resp. ahead). In other words, we are assuming that if t_3, t_2 and t_1 are running on the same edge and t_2 has to accelerate to avoid a collision with t_3, this acceleration will not cause a collision with t_1. This is a very strong assumption and we are extending our negotiation algorithm in order to overcome it. The extended algorithm will fire a message exchange with the (at most two) adjacent trains, whenever a train changes its speed. In this way changing the speed propagates the collision risk, but also the communication among agents to prevent it.

Problem 2: if the number of trains hosted by a station is equal to its maximum capacity, then no more trains can enter it otherwise a collision might take place.
When a train t_i wants to enter a station s_j, ta_i sends a message to sa_j asking a permission to enter. If $current\_capacity_t(s_j) \geq 0$, then sa_j answers to ta_i that t_i can enter the station, otherwise t_i will need to stop and wait that one train leaves s_j. We assume that t_i can stop outside s_j without creating any collision with the trains behind it, if any, running toward s_j. In case there is only one bidirectional edge entering s_j (recall that IRN is a mixed graph), it must be used both by t_i for entering s_j and by the trains already in s_j for exiting, leading to a deadlock. This is a borderline situation that we assume will not take place. We will relax these constraints in the forthcoming version of the negotiation algorithm.

In case of two simultaneous requests involving one train that would stop in the station and another that just needs to traverse it, then the train that will not stop in s_j gets the right to enter. If the trains have the same *stop_value*, the agent with higher payoff will be allowed to enter the station.

Problem 3: if two trains are leaving a station on the same edge at the same time (modulo a small time difference), a collision might occur.
To resolve this problem, when a station agent allows a train t_i to leave on the edge e, it must not allow other trains to leave on e until t_i moves of at least cd (the critical distance). In this situation, the train that should arrive to the next station first is allowed to move on the edge first.

Problem 4: when a train t enters a junction or a station where it should not stop, planning to exit on edge e, and another train is either already moving on e within the critical distance cd, or is planning to move on it, then a collision might occur.
This situation can be faced considering two cases:
1. t_i wants to cross a junction or a station v (without stopping on it), but the edge e where it plans to move is already occupied by t_j whose distance from v is lower than cd. The only action that the agent associated with v can do is to prevent t_i from entering v until the distance between t_j and v becomes greater than cd.
2. t_i wants to cross a station s exiting on edge e, and t_j is waiting in s and is ready to leave it, moving on edge e. Since trains that do not stop in a station

have a higher priority w.r.t. those that stop, t_j is asked to wait, and t_i is allowed to cross s and exit on e.

In both cases we assume that stopping a train on an edge outside a station or a junction does not cause a collision with the train behind that is arriving on that edge. Also this assumption will be removed in the refinement of the current algorithm.

Problem 5: if a bidirectional edge exists between two vertices v_i and v_j and t_h would like to move from v_i to v_j, while t_k would like to move from v_j to v_i, then a collision might occur.
If both trains are still waiting in their stations, this problem is solved by allowing the train with higher payoff to use the bidirectional edge between v_i and v_j first. If one train is already moving on the bidirectional edge, the other train cannot move on it until the edge becomes free again.

Problem 6: given that avoiding collisions is the first goal of the IR system, how could we minimize the total delay of trains running in the IR network?
The delay optimization algorithm we have selected for solving the last problem is the MAX-SUM algorithm described in [16, 17], with the payoff introduced at the beginning of this section. Since the MAX-SUM algorithm is not new, and its instantiation to the IR system has been described in [12], we do not enter into its details here. By adopting the MAX-SUM algorithm, our system is guaranteed to converge into an optimal solution, which minimizes the individual train delay as well as the total system delay, and which additionally reduces the communication and computation cost also.

5 Experiments

We simulated our MAS using NetLogo because of its support for the rapid development of a graphical interface by which an IR administrator could visualize the whole network and the movement of each individual train. The NetLogo program needs three text files as an input:
i) Path.txt, which contains information on the modeled stations
ii) Time_table.txt, which contains the trains' scheduled timetable
iii) Route.txt, which contains a list of stations that a train must follow in its journey (namely, the values of $route(t_i, ind)[\mathbf{ta_i}] = s$, for each *ind* from 1 to the number of stations that t_i must traverse). As NetLogo does not support any message passing protocol between agents, we implemented it by ourselves.

In the next sections we discuss our case studies and the results we obtained by the simulations we executed.

Case studies

The data we used for our experiments have been provided to us by the Eastern India Railway officials. We simulated a sub-network consisting of 42 Km of track with 11 stations, whose features are shown in Table 1.

Station id	Num. of platforms	Adj. stations (with distance in Km)
1	5	2 (8.9)
2	4	3 (7.6), 1 (8.9)
3	6	4 (7.5), 7 (3.0), 2 (7.6)
4	4	5 (12.5), 3 (7.5)
5	4	6 (5.5), 4 (12.5)
6	7	5 (5.5)
7	2	8 (2.3), 3 (3.0)
8	2	9 (4.5), 7 (2.3)
9	2	10 (2.5), 8 (4.5)
10	2	11 (2.5), 9 (2.5)
11	2	10 (2.5)

Table 1. Stations in the simulated *IR* sub-network.

The simulation involves 128 trains: 55 trains run from station 1 to station 6 and 55 trains run from station 6 to station 1. As station 11 lies in a very remote area, only 2 trains are running from station 1 to station 11 via station 3 and vice versa. Similarly, only 8 trains are scheduled from station 3 to 11 and 6 trains are running from station 11 to station 3.

We simulated: the existing non-automated system (without modeling the manual interventions and communication used in order to coordinate the train movements), the MAS where the solutions to the first five problems highlighted in Section 4 have been implemented, and the MAS where the sixth problem has been faced by means of the MAX-SUM algorithm.

1. Simulation of the non-automated system. In the first experiment, we simulated the existing IR sub-network with neither automation nor manual intervention.

2. Simulation of the proposed MAS, without delay optimization. In the second experiment, we simulated the same sub-network, with the same trains running over it as in the first experiments, but where coordination was guaranteed by the MAS described in the previous two sections.

3. Simulation of the proposed MAS, with delay optimization. In the third experiment, we simulated the same sub-network as in the first and second experiments, but where delay is minimized thanks to the MAX-SUM algorithm. With respect to the previous experiments, we changed the number of trains: we created six different configurations with 50, 100, 150, 200, 150 and 300 trains by introducing some random delay and simulated them in NetLogo.

Results

1. Simulation of the non-automated system. The result of the first experiment was that trains got 3.8 hours of cumulative delay in 24 simulated hours (second column of Table 2) and 4 collisions took place. Of course (and luckily!), in the real system such collisions did not take place because of the manual coordination and communication among the IR staff. However, we could not really compare our simulated automated system with the (simulated) actual one, since the IR staff human interventions are so complex and dependent on a large set of environmental variables that they are not reproducible within a simulation. Anyway, the IR staff main objective is avoiding collisions, with little care towards reducing trains delay. Hence, as far as delay reduction is concerned, the actual IR system is almost close to the non-automated one that we simulated and our comparison between the automated and non-automated systems discussed in the two paragraphs below is realistic.

2. Simulation of the proposed MAS, without delay optimization. The result was that trains got 1.2 hours of cumulative delay in 24 simulated hours (third column of Table 2) and no collisions took place.

3. Simulation of the proposed MAS, with delay optimization. The results of the six experiments we run, with different number of trains, are shown in Table 3 and demonstrate that the MAX-SUM algorithm achieves its goal of optimizing the overall delay.

Simulated hour	**Delay in non-autom. system**	**Delay in proposed MAS**
0	0	0
2	0.125	0
4	0.27	0
6	0.35	0
8	0.48	0.02
10	0.56	0.053
12	0.67	0.18
14	0.75	0.2
16	0.83	0.22
18	0.91	0.26
20	0.98	0.32
22	0.99	0.33
24	3.8	1.2

Table 2. Delay Comparison between non-automated system and MAS (in hours).

Discussion

As shown in Figures 1, where the red line (dark grey in b/w images) represents the non-automated system and the green line (light grey in b/w images) represents our MAS without delay optimization, the automation of the IR system according to our model and algorithms gives an high advantage in terms of

Num. of trains	Delay without MAX-SUM	Delay with MAX-SUM
50	0.33	0.33
100	2.33	2
150	9.83	4.67
200	32.83	4.5
250	39.5	5.17
300	47.33	28

Table 3. Delay Comparison between MAS without and with MAX-SUM (in minutes).

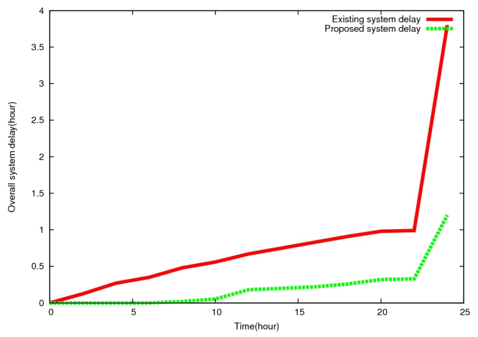

Fig. 1. Non-automated vs automated system: delay comparison.

global delay with respect to a non-automated system. Furthermore, no collisions happen.

Figure 2 shows that the gain in reduced delay when using the MAX-SUM algorithm increases when the number of trains grows up (red line represents the MAS without MAX-SUM and the green line represents the MAS where MAX-SUM has been exploited).

Despite the strong assumptions that we made, the results we obtained in our attempt to model and simulate the huge and complex Indian Railways system are very encouraging both in the reduction of the trains' delay and, most important, in the complete avoidance of collisions.

6 Conclusions and Future Work

In this paper we have presented a model and negotiation algorithms for automatizing the IR signaling and control tasks. Given the features of the IR system, modeling it as a MAS was a very natural choice, and using the NetLogo widespread MAS simulation tool for carrying out our experiments was a further natural consequence of the selected agent-based approach. The results of our experiments are very promising: the system is collision free and the overall delay is reduced. An implementation of the system in Jade[5] has been designed and a

[5] `jade.tilab.com/`, accessed on May, 2013.

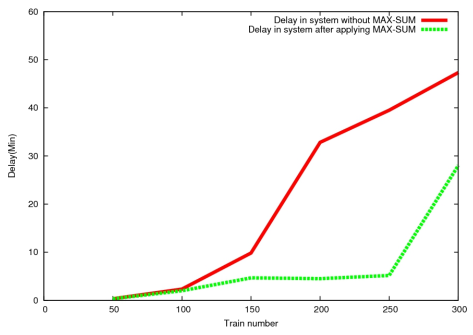

Fig. 2. Delay reduction after the application of MAX-SUM.

first prototype has been implemented. As for the FYPA system, the exploitation of Jade gives the possibility of integrating real databases and existing software components by building a Java bridge towards them, which is usually possible (even if not always easy).

Although we are still far from thinking that IR could adopt our solution, the integration of our MAS on top of the existing IR framework would be feasible and could be done with a low capital investment because the MAS could be implemented using open source software as Jade and could be installed in the existing railway control rooms. The major cost of this technological shift would be due to the improvement of the IR information technology and physical infrastructures to get more reliable and real-time data from the network, which is a requirement for the adoption of our solution. The estimation of this investment is out of our competencies and can be only done by the IR.

The goals we plan to pursue in the future include:

– collecting real data on the railway traffic during some days (delays and, we hope none, collisions), to make a comparison between the solution proposed by our system, simulating the same days, with the actual human solution, instead of using the non-automated system as a reference;

– eliminating the strongest assumptions we made in our model, in order to make our system closer to the real scenario;

– incorporating some intelligent rule engine like JESS[6] to make the agents reasoning and coordination activities smarter than now;

– integrating an ontology representing the domain to better model communication semantics.

References

1. R. Axelrod. Advancing the art of simulation in the social sciences. *Complex.*, 3(2):16–22, Nov. 1997.

[6] `http://herzberg.ca.sandia.gov/`, accessed on May, 2013.

2. S. Bandini, S. Manzoni, and G. Vizzari. Agent based modeling and simulation: An informatics perspective. *Journal of Artificial Societies and Social Simulation*, 12(4):4, 2009.
3. A. Bhardwaj, S. Ghosh, and A. Dutta. Modeling of multiagent based railway system using BDI logic. *International Conference on Future Trends in Computing and Communication*, 2013. To appear.
4. J. Böcker, J. Lind, and B. Zirkler. Using a multi-agent approach to optimise the train coupling and sharing system. *European Journal of Operational Research*, 131(2):242 – 252, 2001.
5. D. Briola and V. Mascardi. Design and implementation of a NetLogo interface for the stand-alone FYPA system. In *WOA 2011, Proceedings*, pages 41–50, 2011.
6. D. Briola, V. Mascardi, and M. Martelli. Intelligent agents that monitor, diagnose and solve problems: Two success stories of industry-university collaboration. *Journal of Information Assurance and Security*, 4(2):106–116, 2009.
7. D. Briola, V. Mascardi, M. Martelli, R. Caccia, and C. Milani. Dynamic resource allocation in a MAS: A case study from the industry. In *From Objects to Agents Workshop, WOA 2009, Proceedings*, 2009.
8. B. Chen and H. H. Cheng. A review of the applications of agent technology in traffic and transportation systems. *Trans. Intell. Transport. Sys.*, 11(2):485–497, 2010.
9. A. Cuppari, P. L. Guida, M. Martelli, V. Mascardi, and F. Zini. Prototyping freight trains traffic management using multi-agent systems. In *In Proc. of IEEE International Conference on Information, Intelligence and Systems*. IEEE, 1999.
10. K. Fischer, N. Kuhn, H. Muller, J. Muller, and M. Pischel. Sophisticated and distributed: The transportation domain. In *Artificial Intelligence for Applications, 1993. Proceedings., Ninth Conference on*, page 454, 1993.
11. K. Fischer, N. Kuhn, and J. Muller. Distributed, knowledge-based, reactive scheduling of transportation tasks. In *Artificial Intelligence for Applications, 1994., Proceedings of the Tenth Conference on*, pages 47–53, 1994.
12. S. Ghosh and A. Dutta. Multi-agent based railway track management system. *In Proc. of 3rd IEEE International Conference on Advance Computing & Communication*, pages 1408–1413, 2013.
13. G. V. H-J Bürckert, K Fischer. Transportation scheduling with holonic mas - the teletruck approach. In *Proceedings of the 14 th European Meeting on Cybernetics and Systems Research*, 1998.
14. J. Lind, K. Fischer, J. Böcker, and B. Zirkler. Transportation scheduling and simulation in a railroad scenario: A multi-agent approach. In *IN PAAM99*, pages 325–344. Springer, 1999.
15. H. Proença and E. Oliveira. Marcs multi-agent railway control system. In *IBERAMIA*, pages 12–21, 2004.
16. R. Stranders, A. Farinelli, A. Rogers, and N. R. Jennings. Decentralised coordination of continuously valued control parameters using the Max-Sum algorithm. *Proc. of 8th Int. Conf. on Autonomous Agents and Multiagent Systems*, 2009.
17. R. Stranders, A. Farinelli, A. Rogers, and N. R. Jennings. Decentralised coordination of mobile sensors using the max-sum algorithm. *Proceedings of the 21st international jont conference on Artifical intelligence*, 2009.
18. C.-W. Tsang and T.-K. Ho. Optimal track access rights allocation for agent negotiation in an open railway market. *Intelligent Transportation Systems, IEEE Transactions on*, 9(1):68–82, 2008.