Safe and Efficient Fleet Operation for Autonomous Machines: An Actor-based Approach

Ali Jafari
School of Computer Science
Reykjavik University, Iceland
ali@ru.is

Jayasoorya Jayanthi Surendran Nair
School of Innovation Design and Engineering
Mälardalen University, Sweden
jjr15001@student.mdh.se

Stephan Baumgart
System Architecture Department
Volvo Construction Equipment, Sweden
stephan.baumgart@volvo.com

Marjan Sirjani
School of Innovation Design and Engineering
Mälardalen University, Sweden
School of Computer Science
Reykjavik University, Iceland
marjan.sirjani@mdh.se

ABSTRACT

In this paper, we formally model and verify run-time requirements of an application consisting of complex electrified machines called HX autonomous haulers, developed by Volvo Construction Equipment. To model the fleet control, we use Timed Rebeca, an actor-based modeling language, and to analyze the system performance, we use Afra, an integrated environment for modeling and verifying distributed systems modeled by Rebeca or Timed Rebeca language. We run a set of experiments to find the improved configuration in which the total time for machines to complete one operating cycle is minimized.

CCS CONCEPTS

• **Computing methodologies → Model verification and validation**; • **Computer systems organization** → *Embedded software*;

KEYWORDS

Actor model, Embedded system, Timed Rebeca, Model checking, Performance evaluation, Autonomous machine

ACM Reference Format:
Ali Jafari, Jayasoorya Jayanthi Surendran Nair, Stephan Baumgart, and Marjan Sirjani. 2018. Safe and Efficient Fleet Operation for Autonomous Machines: An Actor-based Approach. In *SAC 2018: SAC 2018: Symposium on Applied Computing , April 9–13, 2018, Pau, France.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3167132.3167382

1 INTRODUCTION

Modern construction equipment machines include embedded software, which makes these systems highly sophisticated. Many new functions implemented in the embedded systems are safety critical and functional safety standard compliance need to be ensured. Using Autonomous Guided Vehicles (AGVs) is a recent trend in many domains where parts of the workflow can be automated. Here, we consider the Electric Site Research Project that Volvo Construction Equipment is working on [8]. In this project, the transporting of material in a quarry site is automated using a fleet of autonomous haulers (HX). The path between loading, unloading and charging positions is subdivided into tracks for controlling the distribution of machines and realizing different driving characteristics.

Generally, there are two approaches of centralized and decentralized control for realizing this application. In the case of AGVs, the autonomous machines get the needed information about the path from a centralized controller in a specified time interval. According to missions assigned to machines, they are updated by the controller about the location of other machines to avoid collisions.

An alternative to using a centralized server is that AGVs are controlled in a decentralized manner [9]. In this paper, we design a fleet operation model for autonomous machines, in which the machines and tracks are equipped with sensors and communication equipments to send data to each other. Accordingly, this design is based on the sensor networks that guarantee the safety property of "no-collision" using a distributed approach instead of a centralized controller.

To show the applicability of the proposed distributed fleet operation model, we consider the quarry site described above as the case study. Timed Rebeca [1], an actor-based modeling language, is used to model the quarry site including the operational behaviors of machines at the fleet level. We employ Afra [2], an IDE with a dedicated model checker, for both functional verification and performance evaluation of the model. We show that the critical safety property of "no-collision" is guaranteed in the proposed distributed approach. For performance evaluation, we run experiments to find an improved values for the parameters of the system, for which the total time needed by machines to complete one operating cycle is minimized.

This paper is structured as follows. Section 2 describes the related works. Rebeca and its timed extension, namely Timed Rebeca, are described in section 3. Section 4 explains the case study which is inspired by the electrified quarry site at Volvo. In section 5, the modeling approach and the Timed Rebeca model are presented. Section 6 reports the analysis results for different experiments.

2 RELATED WORK

The quarry site use case we analyze in this paper requires planning of machines and their distribution on the tracks. As an input to this analysis, the speed requirements and spend time at different positions at the site need to be provided. Tasks need to be scheduled and synchronized. Generally, schedules can be planned on-line or off-line. Off-line approaches have the possibility to analyze different potential configurations without being dependent on finding a solution within a limited time. Calculating routes and distributions assumes a constant environment with low probability for changes. On-line scheduling algorithms for AGVs have the advantage to react on unforeseen situations, but require a higher computational power and efficient algorithms to provide answers in time.

Different approaches are proposed for planning routes and avoid collision. Secchi et al. [5] use an automated warehouse case for their studies. The authors assume that routes are specified and the autonomous vehicles are distributed across the warehouse. In this work, a new approach is proposed to make decisions at real-time which agent should receive the mission and which routes should be taken based on the current traffic situation. This examples assumes that vehicles are idling and waiting for receiving a new mission. This is different from our case, where the fleet of autonomous haulers is working in a cyclic manner. Another approach for real-time scheduling of AGVs in an industrial context is described by Erol et al. [3]. The authors propose an agent-based scheduling algorithm applying negotiation/bidding concepts for choosing the mission assignment to a specific AGV. When the environment becomes dynamic, routes may change or unknown objects need to be identified and avoided. Zhang et al. [10] use a tool to formally verify an algorithm for safe navigation of robots.

In our use case, we assume that the workflows are to be designed and assurance of the safety of routing is required. Therefore, on-line approaches are not feasible at this stage.

3 REBECA AND TIMED REBECA MODELING LANGUAGES

In this section, we briefly explain the Rebeca language [6, 7], and then we present its extension with timing features to build Timed Rebeca [1].

Rebeca. Rebeca is an actor-based modeling language with formal semantics that is supported by model checking tools. A Rebeca model consists of the definition of reactive classes and the instantiation part which is called main. The main part defines instances of reactive classes, called *rebecs*. The behavior of a rebec is determined by its message servers.

In Rebeca, computation is event-driven, where messages can be seen as events. Each rebec takes a message from its message queue and executes the corresponding message server. Communication takes place by asynchronous message passing, which is non-blocking for both sender and receiver. The behavior of a Rebeca model is defined as the parallel execution of the released messages of the rebecs.

Timed Rebeca. The timing primitives are added to the Rebeca syntax to cover timing features that a modeler might need to address in a message-based, asynchronous and distributed setting. The timing primitives added to the Rebeca syntax are as follows.

- Delay: *delay(t)*, where t is a positive natural number, increases the value of the local clock of the respective rebec by the amount *t*.
- Deadline: *r.m() deadline(t)*, means that the message m is sent to the rebec r and it is put in the message bag. After t units of time the message is not valid any more and is purged from the bag. Deadlines are used to model message expirations (timeouts).
- After: *r.m() after(t)*, the message cannot be taken from the bag before t time units have passed. The *after* primitive is used to model network delays in delivering a message to its destination. Note that *after* primitive can also be used to model periodic events. If we send a message in a loop with *after(t)*, this will cause having the message in the message queue every t units of time. In Timed Rebeca, loops are modeled by sending a message to itself.

In Timed Rebeca, each rebec has its own local clock, but there is also a notion of global time based on synchronized distributed clocks of all the rebecs. Messages that are sent to a rebec are put in its message bag together with their arrival time, and their deadline. Methods are executed atomically, but the passing of time during the execution of methods can be modeled.

4 CASE STUDY

Our case study is inspired by the quarry site used in the electrified site project at Volvo Construction Equipment, where autonomous haulers (HX) are used for transporting material in the site. HX machines are intended to perform tasks such as material transport, loading, unloading, and charging in a cyclic manner with predefined timing constraints and task priorities.

Figure 1 shows an exemplified generic workflow. In order to be generic, we call different possible tasks or resources in the quarry as events. Loading an autonomous machine, unloading or charging are examples of different events. The workflow, type of loading equipment, unloading procedures or charging stations may differ as well as the number of autonomous machines, when different sites are considered.

We explain the operational behavior of HX machines in the site, for which a TRebeca model is developed and performance analysis is carried out and presented in the next sections. The entire path to be traveled by HX machines is composed of several tracks which are named from $S1$ to $S7$ in Figure 1. Each track is divided into an arbitrary number of sub-tracks for which speed profiles are defined. The machines traveling in a sub-track have to comply the corresponding speed profile.

The workflow can start with *Event 4*, where the autonomous machines receive a task to either travel to *Event 1* or to *Event 2. Event 1* can be loading the machines with a wheel loader (WL). *Event 2* can be a direct loading of the machines from a stone crusher (SC). Track $S4$ requires reduced speed to maneuver towards the branching point. Tracks $S5$ and $S6$ are high speed tracks, where the machines can travel in maximum speed. When loaded, the machines shall transport the material to the dumping spot (DS) at *Event 3*. Tracks *S*1 and *S*7 are slow driving areas, because of approaching the point where both tracks join again to travel with high speed on track *S*2. Electric autonomous machines need to recharge the battery, which may be done during each cycle for simplicity. Nonetheless, more complex scenarios can be modeled by the abstraction presented above.

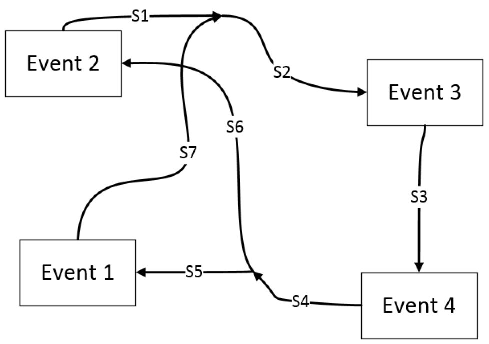

Figure 1: An abstract presentation of tracks and events in a Fleet Operation Site

5 FORMAL MODELING OF FLEET OPERATION

In this section, we first describe the proposed modeling approach of track-based applications. According to this approach, we model the case study explained in Section 4 using the TRebeca language.

To model the case study shown in Figure 1 in TRebeca, we consider tracks and events as rebecs (actors), and machines as messages being sent among rebecs. To be able to assure safety of the site, and control the distance of machines from each other, we further refine this modeling approach by dividing each track to sub-tracks. Each sub-track is dealt with as a critical section, only one machine can be on a sub-track at the same time. Sub-tracks and machines are equipped with sensors and communication devices. A machine traveling in a sub-track sends a request to the next sub-track when approaching it. If the next sub-track is free, i.e., if it is not occupied by another machine, the machine gets a permission signal and is able to exit the current sub-track and enter the next one. Otherwise, the machine stops in a "safe distance" from the next sub-track (or slow down) until it gets the permission. Since we assure the existence of only one machine inside a sub-track, collision avoidance is guaranteed by design. Therefore, there is no need to calculate the distance of machines from each other to avoid collisions.

This design decision has other advantages too. Each sub-track has its own speed profile and length. We are able to model different physical situations in some part of the site. For example, if a part of track S7 is icy in winter and the machines must keep their speed below 20 km/h, this can be modeled by specifying this part as a sub-track with predefined length and permitted speed.

More importantly, as we are able to set the length and the permitted speed for every sub-track as well as the safe distance from the next sub-track, this approach can be generalized and used for modeling any other track-based fleet control applications.

Timed Rebeca Model. The TRebeca model consists of seven different reactive classes: *StoneCrusher*, *DumpingSpot*, *WheelLoader*, *StartingPoint*, *SubTrack*, *PrePoint*, and *CrossController*.

The first four reactive classes model the resources (events) in the site. The *SubTrack* is used to model sub-tracks. As previously explained, a sequence of sub-tracks constructs a track in Figure 1, which is used to connect resources to each other. Tracks are used by machines to navigate in the site. Track S5 has three sub-tracks; the remaining tracks have 5 sub-tracks.

The *StartingPoint (Event 4)* sends machines to *PrePoint*, where sub-track *S*4 is forked into *S*5 and *S*6. *PrePoint* manages the machines and sends them towards either *StoneCrusher (Event 2)* or *WheelLoader (Event 1)*. The machines are sent to these two Events alternatingly. If the sub-track next to *PrePoint* and towards the chosen target is occupied then *PrePoint* chooses the alternate target. These policies can easily be changed in the TRebeca model.

The reactive class *CrossController* is responsible for controlling the critical section created in the crossing point of tracks *S*6 and *S*7. The crossing point is the common part of two sub-tracks belonging to tracks *S*6 and *S*7. The *CrossController* allows only one machine to pass the crossing at a given time. After getting unloaded at the *DumpingSpot (Event 3)*, a machine moves towards the starting point where it completes one operating cycle. The complete TRebeca model can be found in [4].

6 ANALYSIS RESULTS

In this section, we use the Afra tool for functional verification and performance evaluation of the Timed Rebeca model explained in Section 4. The model checker of Afra checks "deadlock-freedom" and "no deadline-miss" properties for the model automatically.

For performance evaluation, the state space (in the form of timed transition system) generated by the model checker of Afra is fed into an script to calculate the *spent time*, which is the total time needed for all machines to complete one operating cycle. For each machine, the operating cycle starts from a parking slot at the starting point, continues by loading, unloading tasks, and finishes when the machine gets back to the starting point again.

We run some experiments, each of which consisting of different configurations, in order to understand the effect of the system parameters on the spent time. The goal is to find the improved configuration in which the spent time is minimized.

The following parameters are considered in the TRebeca model, and specify different characteristics of the system. The value of these parameters can affect the value of the spent time: PERIOD, NORMAL_SPEED, REDUCED_SPEED, SAFE_DISTANCE, SUB_TRACK_LENGTH, LOADING_TIME_SC, LOADING_TIME_WL, UNLOADING_TIME_DS, NUMBER_VEHICLES.

Some of these parameters are defined. 1) PERIOD: the machine waiting behind the next sub-track to get entrance permission resends its request again after a specified time, which is called period. 2) NORMAL_SPEED: in most sub-tracks, machines are traveling with a normal and predefined speed. 3) REDUCED_SPEED: in a few sub-tracks, machines must travel with a lower speed. 4)

SAFE_DISTANCE: when a machine wants to exit its current sub-track, it must stop in a safe distance from the next sub-track as there may be another machine at the beginning of the next sub-track.

Tracks *S*1, *S*4 and *S*7 have five sub-tracks. Considering Figure 1, in all experiments the last three sub-tracks of *S*1 and *S*4, and the last two sub-tracks of *S*7 have the speed limit of REDUCED_SPEED. This way, we can model areas in which machines should be driven slowly for some reason.

The value of the following parameters are fixed in all experiments.

- SUB_TRACK_LENGTH = 200 meters
- SAFE_DISTANCE = 20 meters
- LOADING_TIME_SC = 60 seconds
- LOADING_TIME_WL = 60 seconds
- UNLOADING_TIME_DS = 30 seconds
- REDUCED_SPEED = NORMAL_SPEED /2

In the first experiment, we examine the effect of changing the "normal speed" and "period" on the spent time. To this end, the normal speed changes from 10 to 20 m/s and the period varies from 15 to 25 seconds. More specifically, we run the TRebeca model with 121 different values for the normal speed and the period, and keep the value of other parameters fixed. The spent time is calculated for each setting. The results are shown in Figure 2 for the system with six machines traveling in the site. The minimum value of the spent time is 561 seconds when the normal speed is 20 m/s and the period equals to 22 seconds.

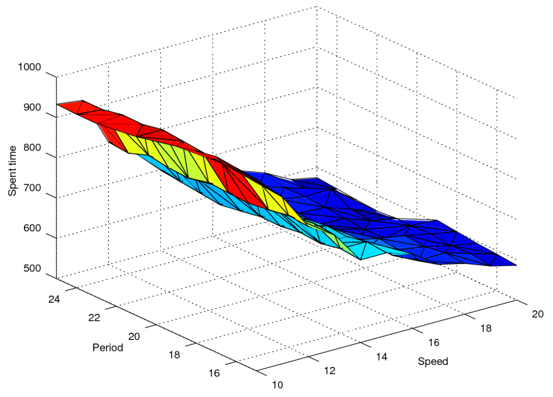

Figure 2: The effect of "speed" and "period" on the spent time with six machines in the system

In the second experiment, we investigate the effect of period and the number of machines on the spent time. To this end, the values of the normal speed and the reduced speed are set to 18 m/s and 9 m/s, respectively. The value of period is chosen from integer interval [15, 25], and the number of machines varies from 3 to 6. The experiment includes running the TRebeca model with 44 different settings. The analysis results are shown in Figure 3.

To decrease the total spent time, The machines can increase their speed. But, higher speed uses more energy in machines. Using our methodology, we are able to find a setting in which the total spent time is below a desired threshold, and the energy consumption is minimized. As an example, in the first experiment, it's enough to find the minimum speed while the time threshold is preserved.

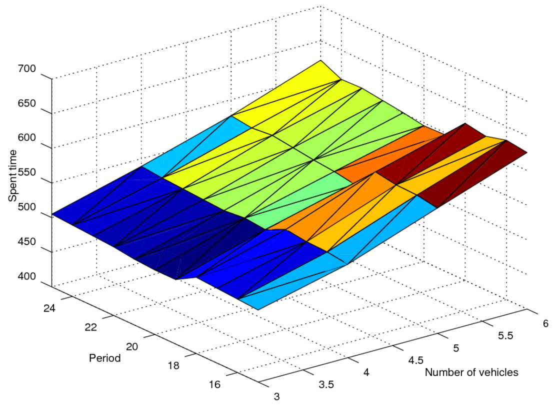

Figure 3: The effect of "period" and "number of machines" on the spent time.

Acknowledgments. The work of the first author, and a fraction of work of the forth author are supported by the project "Self-Adaptive Actors: SEADA" (163205-051) of the Icelandic Research Fund. Work of the forth author is also supported in part by DPAC Project (Dependable Platforms for Autonomous Systems and Control) at Mälardalen University, Sweden. The third author acknowledges the funding support received for this research from the KKS-funded ITS- EASY Post Graduate School for Embedded Software and Systems and the Electric Site Project funded by the Swedish Energy Agency.

REFERENCES

[1] Luca Aceto, Matteo Cimini, Anna Ingólfsdóttir, Arni Hermann Reynisson, Steinar Hugi Sigurdarson, and Marjan Sirjani. 2011. Modelling and Simulation of Asynchronous Real-Time Systems using Timed Rebeca. In *FOCLASA'11*. 1–19.

[2] Afra. 2017. Afra: an integrated environment for modeling and verifying Rebeca family designs. (2017). Retrieved 2017-12-09 from https://rebeca-lang.github.io/alltools/Afra

[3] Rizvan Erol, Cenk Sahin, Adil Baykasoglu, and Vahit Kaplanoglu. 2012. A multi-agent based approach to dynamic scheduling of machines and automated guided vehicles in manufacturing systems. *Applied Soft Computing* 12, 6 (2012), 1720 – 1732. https://doi.org/10.1016/j.asoc.2012.02.001

[4] Rebeca. 2017. Rebeca Homepage. (2017). Retrieved 2017-12-09 from http://www.rebeca-lang.org

[5] C. Secchi, R. Olmi, F. Rocchi, and C. Fantuzzi. 2015. A dynamic routing strategy for the traffic control of AGVs in automatic warehouses. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*. 3292–3297. https://doi.org/10.1109/ICRA.2015.7139653

[6] M. Sirjani and A. Movaghar. 2001. *An Actor-Based Model for Formal Modelling of Reactive Systems: Rebeca.* Technical Report CS-TR-80-01. Tehran, Iran.

[7] M. Sirjani, A. Movaghar, A. Shali, and F.S. de Boer. Dec. 2004. Modeling and Verification of Reactive Systems using Rebeca. *Fundamenta Informatica* 63, 4 (Dec. 2004), 385–410.

[8] Volvo Construction Equipment. 2017. Innovation at Volvo Construction Equipment. https://www.volvoce.com/global/en/this-is-volvo-ce/what-we-believe-in/innovation/. (2017). [Online; accessed 24-August-2017].

[9] Danny Weyns, Tom Holvoet, Kurt Schelfthout, and Jan Wielemans. 2008. Decentralized Control of Automatic Guided Vehicles: Applying Multi-agent Systems in Practice. In *Companion to the 23rd ACM SIGPLAN Conference on Object-oriented Programming Systems Languages and Applications (OOPSLA Companion '08)*. ACM, New York, NY, USA, 663–674. https://doi.org/10.1145/1449814.1449819

[10] M. Zhang and X. Zhang. 2016. Formally verifying navigation safety for ground robots. In *2016 IEEE International Conference on Mechatronics and Automation*. 1000–1005. https://doi.org/10.1109/ICMA.2016.7558699