The analysis of a conveyor belt system : a case study in hybrid systems and timed muCRL

Eindhoven University of Technology
Department of Mathematics and Computing Science

The Analysis of a Conveyor Belt System
a case study in Hybrid Systems ant timed μCRL

by

Tim A.C. Willemse

99/10

ISSN 0926-4515

Reports are available at:
http://www.win.tue.nl/win/cs

Computing Science Reports 99/10
Eindhoven, August 1999

The Analysis of a Conveyor Belt System
a case study in Hybrid Systems and timed μCRL

Tim A.C. Willemse

Eindhoven University of Technology
Department of Mathematics and Computing Science
P.O. Box 513, NL-5600 MB Eindhoven, The Netherlands
timw@win.tue.nl

Abstract

A Conveyor belt system, together with its safety and efficiency objectives is described and studied in the setting of timed μCRL [9]). First, various safety requirements, dealing with the safe transportation of trays from the front end to the back end of the belt, and the speed of the belt are stated. Subsequently, a specification in timed μCRL is given for such a system. This specification is the subject of further calculations and analysis. Using various techniques and theorems, devised within the theory of timed μCRL, such as the Sum Elimination Theorem and the Encapsulation and Expansion Theorem, properties of the system are proved and discussed. Also, a section is devoted to a brief comparison between the theory of Hybrid Automata [14] and timed μCRL.

Contents

1 Introduction

Hybrid systems are systems in which the essence of the system cannot be captured by describing only the continuous or the discrete behaviour, but the combination of both is essential. The interest in these systems has its foundations not solely in the academic world, it also shows in the industry, in which many real-life physical components, including complete factories have to be automated. For many of the hybrid systems that are considered, very strict safety and liveness requirements must hold. Safety requirements basically state that nothing bad ever happens. One can think of requirements that protect people from physical threat (e.g. in aviation, air-traffic control, control of nuclear power plants), or requirements ensuring no products are lost or damaged. The liveness requirements often deal with progress issues, ensuring eventually something good will happen. Apart from these requirements, efficiency is often an issue not to be discarded, especially from an industrial point of view.

On top of these requirements, some of which are very hard to test in real-life, there is the fact that financially it is very interesting to consider models of systems. This especially applies to the more expensive or hazardous physical systems. Traditionally, in the engineering disciplines of mechanical engineering, electrical engineering, chemical engineering, etc. the (continuous) physical systems are described using for instance Differential Algebraic Equations (see f.i. [6]). The description of the discrete parts of the hybrid system, is often a task that best fits in with the disciplines of (discrete) mathematics and computing science. The techniques used within these disciplines vary from process algebras and automata to state charts. It is rather remarkable that only recently, attempts have been made to integrate the techniques used in these disciplines. Already, many methods and models have been proposed ([4], [17], [16], [5]) and investigation into these methods and techniques and their applications still is a hot topic.

These relatively recent developments provide a new area of application for process algebras, having a history of being applied to discrete systems, such as (data) communications protocols. Within the theory of process algebras, various approaches exist, such as CCS [19], LOTOS [15] and ACP [3]. Already a few case studies, in which process algebras have been used to describe a hybrid system, have been made (e.g. [1] and [11]).

In this article, the formalism timed μCRL [9], is used to describe and analyse a hybrid system, consisting of a conveyor belt and its controller. The example is inspired by [20] and [7]. In this example, the conveyor belt is a physical system, governed by certain DAEs, and put to use for transportation of trays from the front end to the back end. The task of the controller, which is discrete in nature, is to make sure the system as a whole adheres to four (simple) requirements, dealing with liveness, safety aspects such as collision and speed controlling, and efficiency aspects. In section 2, both the controller and the conveyor belt are described in greater detail, leading to descriptions in timed μCRL; these descriptions are then simplified, mainly using theorems such as the Sum Elimination Theorem. Section 3 describes the behaviour of the hybrid system as the composition of the specifications derived in section 2; various invariants are formulated proving that, if no deadlock occurs, the system adheres to all safety requirements, provided the system is deadlock free. Section 4 subsequently shows the required absence of deadlock and comments on some operational analysis. Section 5 contains a brief comparison between the formalism of Hybrid Automata [14] and timed μCRL and finishes with some closing remarks on future work.

It turns out the formalism timed μCRL is quite suited for the description of systems which are hybrid in nature, and that it allows for thorough analysis of these systems. However, this case study also shows that still a lot of research needs to be done in the formalism timed μCRL; for instance the rather alarming number of timed deadlocks arising from the application of the Encapsulation and Expansion theorem is something to be looked into.

2 The components of the Conveyor Belt System

The system described in this article is a conveyor belt system, which consists of a *belt* and a *controller*. The system defining the interaction between the belt and the controller is called the *conveyor belt system* (see figure 1).

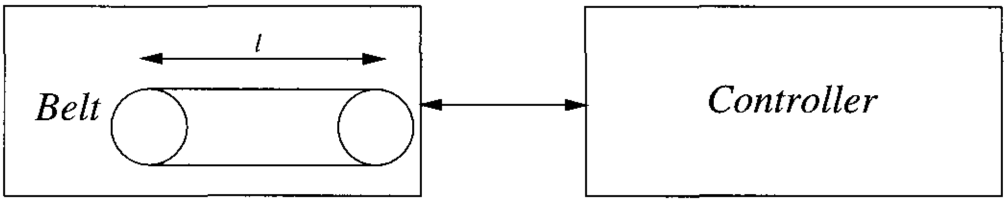

Figure 1: *The conveyor belt system*

It is assumed that the conveyor belt system is put to use for the transportation of *trays*. The remainder of this section is devoted to describing the design criteria for the conveyor belt system.

2.1 The control objectives

For reasons of convenience, the belt is assumed to have a fixed length $0 < l$. The belt is 0 with dedicated sensors, positioned at $0, \frac{1}{2}l$ and l. A sensor is triggered the moment an object passes it. Furthermore, it is a reasonable assumption that the trays that need transportation, have a fixed length of $0 < n < \frac{1}{2}l$. This way, at least two trays fit on the belt. The control objectives of the system are as follows:

- Trays should be transported from the front end to the back end environment.
- Trays should not collide.
- The velocity of the belt must be in the range of 0 to v_{max}, where $0 < v_{max}$.
- Support a mechanism to make a trade off between the throughput of trays and the energy efficiency of the system.

The first requirement is classified as a liveness property, since in the end we wish to put the conveyor belt system to use. The second and third objective can be classified as safety requirements, since the violation of these requirements will result in physical damage of either system or product. The fourth requirement forces no design, however, it is a control objective since it does affect the behaviour of the conveyor belt system.

2.2 Narrowing the choices

Since the control objectives leave a lot of freedom, one additional requirement is formulated, which proves to be sufficient to narrow the choices for a decent model of the conveyor belt system. An obvious choice is to allow for up to two trays on a belt, since a belt is capable of carrying two trays at a time with the system still adhering to the second control objective. The belt must then be able to accept a new tray when the previous tray has reached the mid point of the belt.

2.2.1 The discrete model

The discrete model, depicted in figure 2 is, with respect to the previous sections, the model that is intuitively enforced.
The transitions between states are labelled with events, which basically have the following meaning associated with them:

- The arrival of a tray at the front end of the belt (*ar*).
- The arrival of a tray at the mid point of the belt (*mid*).

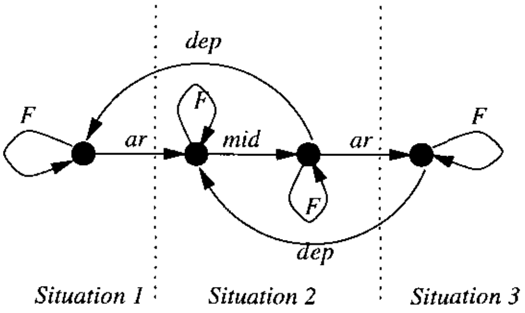

Figure 2: *A transition diagram*

- The departure of a tray from the belt (*dep*).
- The adjusting of the *torque* applied to the motor of the belt (F) (see also section 2.2.2).

The model depicted in figure 2 furthermore makes a distinction towards the number of trays on the belt. In words, the following situations are modelled:

- No tray is present on the belt (situation 1),
- One tray is present on the front end OR the back end of the belt (situation 2),
- Two trays are present on the belt (situation 3).

2.2.2 The continuous model

Since the position of a tray is an essential part of the discrete model, this somehow has to be incorporated into a hybrid model combining both the discrete and the continuous model. Basic mechanical engineering shows that a model for the position of a tray can be written as a Differential Algebraic Equation (DAE) (see for instance [6]).

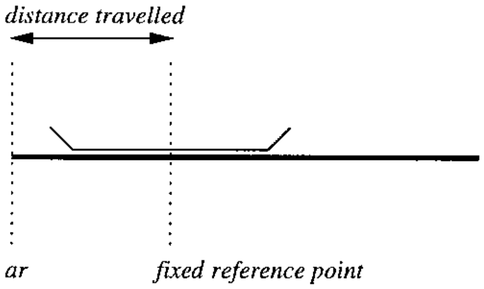

Figure 3: *The fixed reference point*

Although a number of factors can have an impact on the displacement of the trays, the following modelling assumptions are made:

- The velocity of the belt (and thus of the trays on the belt), is (indirectly) controlled by the *torque* applied to the motor of the belt.
- A tray, together with its load, is assumed to have a fixed mass, which is too small to affect the velocity of the belt in a substantial manner.
- A tray is assumed to have a fixed reference point (see figure 3).
- A tray is only carried by the belt that carries the reference point.

The third assumption is basically the one that allows us to write the displacement as a simple DAE, the other three assumptions are introduced in order to keep the model as simple as possible.

The system can now be described using the following three DAE's:

$$x_0'(\zeta) = bf(\zeta) \tag{2.1}$$

$$x_1'(\zeta) = Ax_1(\zeta) + Bf(\zeta) \tag{2.2}$$

$$x_2'(\zeta) = Cx_2(\zeta) + Df(\zeta) \tag{2.3}$$

The constants A, B, C and D are matrices of appropriate dimension, defined in equation 2.4. The function x_i is a function from the domain **Time** to $\mathbb{R}^{i+1}$. In words, the model expresses how the speed and the position of a tray are related in time. It is assumed that the function f : **Time** $\to \mathbb{R}$ is prescribed by the controller, and represents the value of the torque in time. Intuitively, the functions model the following:

- The value of $x_0(t)$ describes the velocity of the belt at time $t \in$ **Time** when no tray is present on the belt.
- For each $t \in$ **Time**, the first component of $x_1(t)$ expresses the distance covered by the tray at time t and the second component expresses the velocity of the belt at time t, when exactly *one* tray is present on the belt.
- For each $t \in$ **Time**, the first component of $x_2(t)$ expresses the distance covered by the foremost tray at time t, the second component expresses the velocity of the belt at time t and the third component represents the distance covered by the subsequent tray at time t, provided that there are exactly two trays present on the belt.

Note: Throughout this article, the constants A, B, C and D are matrices, defined as follows:

$$A = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}, B = \begin{pmatrix} 0 \\ b \end{pmatrix}, C = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \text{ and } D = \begin{pmatrix} 0 \\ b \\ 0 \end{pmatrix} \tag{2.4}$$

The constant b in B and D is present to account for a number of physical phenomena, such as friction and a fraction for the masses of the tray.

2.3 Modelling the Conveyor Belt System

The natural problem that arises when constructing a model of a real system, or a system that is yet to be built, is how to abstract from reality and capture the essence of the system. A few helpful paradigms are separation of concerns and orthogonality. These paradigms basically state that when something is irrelevant to one system, this system should not be bothered by this. When modelling the conveyor belt system, it is of importance to apply these two paradigms to the models. A natural description of the system is by decomposing it into the real-life objects, i.e. the belt and the controller. The objective now is to model these entities as realistic as possible, without having both affect each other.

2.3.1 A hybrid model for the belt

The idea is to combine both the discrete model and the continuous model described in the previous sections into one model. This hybrid model then has to be intuitive, readable and concise. Using the formalism timed μCRL [9], these goals can easily be achieved: the discrete model is easily translated to a model in μCRL [10], and the timed features of timed μCRL allow for a synchronisation of the continuous model and the discrete model in an intuitive fashion.

For modelling the belt in a more detailed fashion, it is assumed that the following requirements hold:

- The belt is equipped with sensors, which are triggered the moment the fixed reference point of the tray "hits" it.

- The belt has no means of controlling and influencing its own speed.
- The belt is always able to accept new trays, even if this would mean that trays would collide (in a more pragmatic way: the belt has no means to influence the environment).

The model built around these requirements consists of three different processes, viz. $Belt_0$, $Belt_1$ and $Belt_2$, which are subsequently explained and described. One additional abbreviation is introduced: θ_i is written for the function space $\mathbf{Time} \to \mathbb{R}^{i+1}$.

In order to describe the discrete model in timed μCRL, the following actions are introduced:

- Action ar_b signals the arrival of a tray at the front end of the belt.
- Action mid_b signals the arrival of a tray at the mid point of the belt.
- Action dep_b signals the departure of a tray from the belt.
- Action $F_b(\gamma)$ represents the receiving of the setting of the torque of the motor to γ, where $\gamma \in Torque = \{-T_{max}, 0, +T_{max}\}$ and $0 < T_{max}$.

Information about a previous (continuous) state is registered in dedicated parameters. The parameter t is used to denote the time the system last performed an action, parameter x_s denotes the the continuous state of the belt at time t (i.e. speed and positions of trays if applicable) and the parameter f denotes the torque applied to the motor of the belt at time t.

Process $Belt_0$ describes the situation in which no tray is occupying the belt, and in which the belt is essentially waiting for new trays to arrive. Furthermore, the torque of the motor can be set in this state. The process description is as follows:

$$\textbf{proc } Belt_0(t : \mathbf{Time}, x_s : \mathbb{R}, f : Torque) =$$

$$\overline{\Sigma}_{u:\mathbf{Time},x_0:\theta_0}\ ar_b\text{‘}u\ Belt_1(u,(0,x_0(u)),f) \lhd x_0'(\zeta) = bf \wedge x_s = x_0(t) \rhd \delta\text{‘}\mathbf{0}+$$

$$\overline{\Sigma}_{u:\mathbf{Time},x_0:\theta_0,\gamma:Torque}\ F_b(\gamma)\text{‘}u\ Belt_0(u,x_0(u),\gamma) \lhd x_0'(\zeta) = bf \wedge x_s = x_0(t) \rhd \delta\text{‘}\mathbf{0}$$

Process $Belt_1$ describes the situation in which one tray is occupying the belt, which is waiting to be transported to the end of the belt. The events that can take place are the setting of the torque of the motor, the arrival of a tray at the mid point of the belt and the departure of the tray from the belt. Moreover, a new tray can arrive in this state. The function π_i is introduced to denote the projection onto the i-th argument of a tuple.

$$\textbf{proc } Belt_1(t : \mathbf{Time}, x_s : \mathbb{R}^2, f : Torque) =$$

$$\begin{aligned}&\overline{\Sigma}_{u:\mathbf{Time},x_1:\theta_1}\ mid_b\text{‘}u\ Belt_1(u,x_1(u),f)\\ &\quad \lhd x_1'(\zeta) = Ax_1(\zeta) + Bf \wedge x_s = x_1(t) \wedge \pi_1(x_1(u)) = \tfrac{1}{2}l \rhd \delta\text{‘}\mathbf{0}+\end{aligned}$$

$$\overline{\Sigma}_{u:\mathbf{Time},x_1:\theta_1}\ ar_b\text{‘}u\ Belt_2(u,(x_1(u),0),f) \lhd x_1'(\zeta) = Ax_1(\zeta) + Bf \wedge x_s = x_1(t) \rhd \delta\text{‘}\mathbf{0}+$$

$$\begin{aligned}&\overline{\Sigma}_{u:\mathbf{Time},x_1:\theta_1}\ dep_b\text{‘}u\ Belt_0(u,\pi_2(x_1(u)),f)\\ &\quad \lhd x_1'(\zeta) = Ax_1(\zeta) + Bf \wedge x_s = x_1(t) \wedge \pi_1(x_1(u)) = l \rhd \delta\text{‘}\mathbf{0}+\end{aligned}$$

$$\overline{\Sigma}_{u:\mathbf{Time},x_1:\theta_1,\gamma:Torque}\ F_b(\gamma)\text{‘}u\ Belt_1(u,x_1(u),\gamma) \lhd x_1'(\zeta) = Ax_1(\zeta) + Bf \wedge x_s = x_1(t) \rhd \delta\text{‘}\mathbf{0}$$

Process $Belt_2$ describes the situation in which two trays are present on the belt. Because one of the requirements is that at most two trays can be on the belt at any moment in time, this results in the absence of an ar_b action. The action mid_b is also not allowed, since this would imply that the trays are less than $\frac{1}{2}l$ distance apart. The actions that are again possible are the action dep_b, allowing for trays to depart from the belt, and F allowing for a change of velocity of the belt.

This is in accordance to the discrete transition diagram in figure 2.

proc $Belt_2(t : \mathbf{Time}, x_s : \mathbb{R}^3, f : Torque) =$

$$\overline{\Sigma}_{u:\mathbf{Time},x_2:\theta_2}\ dep_b{}^{\backprime}u\ Belt_1(u, (\pi_3(x_2(u)), \pi_2(x_2(u))), f)$$
$$\triangleleft x_2'(\zeta) = Cx_2(\zeta) + Df \wedge x_s = x_2(t) \wedge \pi_1(x_2(u)) = l \triangleright \delta{}^{\backprime}\mathbf{0}+$$

$$\overline{\Sigma}_{u:\mathbf{Time},x_2:\theta_2,\gamma:Torque}\ F_b(\gamma){}^{\backprime}u\ Belt_2(u, x_2(u), \gamma) \triangleleft x_2'(\zeta) = Cx_2(\zeta) + Df \wedge x_s = x_2(t) \triangleright \delta{}^{\backprime}\mathbf{0}$$

The model described above is a reasonable model with respect to the requirements posed so far.

Note: this model already incorporates some of the meta knowledge that has been written down. For instance, the decision to not allow for a new tray to arrive when already two trays are present on the belt is debatable, since this assumes that the belt itself has some way of rejecting new trays, which is not completely according to the third requirement. Similarly, one could argue that the process $Belt_2$ is too restrictive in that it does not allow for a tray to arrive at the mid point of the belt. However, in order to keep the model as concise as possible, the above solution was adopted.

2.3.2 A hybrid model for the controller

Again, in this section, the discrete model presented in section 2.2.1 and the continuous model presented in section 2.2.2 are combined into one hybrid model, using the formalism timed μCRL. The emphasis of the controller is on adjusting the velocity of the belt. For modelling the controller in a more detailed fashion, it is assumed that the following requirements are met:

- The controller has information about the velocity of the belt.
- When no trays are present on the belt, the controller allows for some time to pass, say k time-units, before the belt is decelerated. This delay is introduced to allow for a trade off between energy efficiency and throughput efficiency, such that the fourth requirement in section 2.1 can be met.
- The controller is able to act instantaneously on events.

In order to describe the discrete model in timed μCRL, the following actions are introduced:

- Action ar_c signals the arrival of a tray at the front end of the belt.
- Action mid_c signals the arrival of a tray at the mid point of the belt.
- Action dep_c signals the departure of a tray from the belt.
- Action $F_c(\gamma)$ represents the setting of the torque of the motor to γ, where $\gamma \in Torque$.

Information about a previous (continuous) state is registered in dedicated parameters. Again, parameter t denotes the time the system last performed an action, parameter x_s denotes the relevant continuous state of the belt (in case of the controller only the velocity of the belt), and the parameter f is used to denote the torque applied to the motor of the belt at time t.

Process $Cont_0^{nd}$ describes the situation when there is no tray present on the belt. If the controller is in this state, it assumes that the belt is, if it did not already come to a stand still, decelerating. At any time, the controller must be able to receive a signal a new tray arrived, and at the same time, monitor and set the velocity of the belt.

proc $Cont_0^{nd}(t : \mathbf{Time}, x_s : \mathbb{R}, f : Torque) =$

$$\overline{\Sigma}_{u:\mathbf{Time},x_0:\theta_0}\ ar_c{}^{\backprime}u\ Cont_1^{nac,ds}(u, x_0(u), f) \triangleleft x_0'(\zeta) = bf \wedge x_s = x_0(t) \wedge 0 \leq x_0(u) \leq v_{max} \triangleright \delta{}^{\backprime}\mathbf{0}+$$

$$\overline{\Sigma}_{u:\mathbf{Time},x_0:\theta_0}\ F_c(0)\text{‘}u\ Cont_0^{nd}(u,0,0) \lhd x_0'(\zeta) = bf \land x_s = x_0(t) \land x_0(u) = 0 \land f \neq 0 \rhd \delta\text{‘}\mathbf{0}$$

Process $Cont_0^d$ is in some sense the complementary process to process $Cont_0^{nd}$. It describes the situation when there is no tray occupying the belt, yet, in this state it assumes that the belt is, if it did not already reach v_{max}, accelerating. The reason for this could be that a tray was on the belt less than k time units ago. The controller tasks are essentially the same as for the situation described by process $Cont_0^{nd}$.

proc $Cont_0^d(t, d : \mathbf{Time}, x_s : \mathbb{R}, f : Torque) =$

$$\overline{\Sigma}_{u:\mathbf{Time},x_0:\theta_0}\ ar_c\text{‘}u\ Cont_1^{nac,am}(u, x_0(u), f)$$
$$\lhd x_0'(\zeta) = bf \land x_s = x_0(t) \land 0 \leq x_0(u) \leq v_{max} \land u \leq t + d \rhd \delta\text{‘}\mathbf{0}+$$

$$\overline{\Sigma}_{u:\mathbf{Time},x_0:\theta_0}\ F_c(0)\text{‘}u\ Cont_0^d(u, d + t - u, v_{max}, 0)$$
$$\lhd x_0'(\zeta) = bf \land x_s = x_0(t) \land x_0(u) = v_{max} \land f \neq 0 \land u < t + d \rhd \delta\text{‘}\mathbf{0}+$$

$$\Sigma_{x_0:\theta_0}\ F_c(-T_{max})\text{‘}(t + d)\ Cont_0^{nd}(t + d, x_0(t + d), -T_{max})$$
$$\lhd x_0'(\zeta) = bf \land x_s = x_0(t) \land 0 < x_0(t + d) \leq v_{max} \rhd \delta\text{‘}\mathbf{0}$$

Process $Cont_1^{nac,ds}$ describes the situation when exactly one tray is occupying the belt. According to the latest information, the belt is not yet able to accept a new tray and the belt is either still decelerating or it has already come to a stand still. The main task in this situation for the controller is to set the torque of the motor such that v_{max} is maintained or reached.

proc $Cont_1^{nac,ds}(t : \mathbf{Time}, x_s : \mathbb{R}, f : Torque) =$

$$F_c(0)\text{‘}t\ Cont_1^{nac,am}(t, v_{max}, 0) \lhd x_s = v_{max} \rhd \delta\text{‘}\mathbf{0}+$$

$$F_c(T_{max})\text{‘}t\ Cont_1^{nac,am}(t, s, T_{max}) \lhd x_s < v_{max} \rhd \delta\text{‘}\mathbf{0}$$

Process $Cont_1^{nac,am}$ describes the situation when exactly one tray is occupying the belt. Furthermore, it is assumed that in this state, the belt is accelerating or has already reached maximal speed. Since the controller did not yet receive a mid_c action, the controller must make sure, no new trays can arrive.

proc $Cont_1^{nac,am}(t : \mathbf{Time}, x_s : \mathbb{R}, f : Torque) =$

$$\overline{\Sigma}_{u:\mathbf{Time},x_0:\theta_0}\ mid_c\text{‘}u\ Cont_1^{ac}(u, x_0(u), f) \lhd x_0'(\zeta) = bf \land x_s = x_0(t) \land 0 \leq x_0(u) \leq v_{max} \rhd \delta\text{‘}\mathbf{0}+$$

$$\overline{\Sigma}_{u:Time,x_0:\theta_0}\ F_c(0)\text{‘}u\ Cont_1^{nac,am}(u, v_{max}, 0)$$
$$\lhd x_0'(\zeta) = bf \land x_s = x_0(t) \land x_0(u) = v_{max} \land f \neq 0 \rhd \delta\text{‘}\mathbf{0}$$

Process $Cont_1^{ac}$ describes the situation when exactly one tray is occupying the belt *and* it is possible to accept a new tray, without having trays collide. Furthermore, the controller must be able to receive a dep_c action which informs the controller that the tray has left the belt.

proc $Cont_1^{ac}(t : \mathbf{Time}, x_s : \mathbb{R}, f : Torque) =$

$$\overline{\Sigma}_{u:\mathbf{Time},x_0:\theta_0}\ ar_c\text{‘}u\ Cont_2(u, x_0(u), f) \lhd x_0'(\zeta) = bf \land x_s = x_0(t) \land 0 \leq x_0(u) \leq v_{max} \rhd \delta\text{‘}\mathbf{0}+$$

$$\overline{\Sigma}_{u:\mathbf{Time},x_0:\theta_0}\ dep_c\text{‘}u\ Cont_0^d(u, k, x_0(u), f) \lhd x_0'(\zeta) = bf \land x_s = x_0(t) \land 0 \leq x_0(u) \leq v_{max} \rhd \delta\text{‘}\mathbf{0}+$$

$$\overline{\Sigma}_{u:\mathbf{Time},x_0:\theta_0}\ F_c(0)\text{‘}u\ Cont_1^{ac}(u, v_{max}, 0) \lhd x_0'(\zeta) = bf \land x_s = x_0(t) \land x_0(u) = v_{max} \land f \neq 0 \rhd \delta\text{‘}\mathbf{0}$$

Process $Cont_2$ describes the situation where there are two trays occupying the belt. Since in this state, the belt cannot accept a new tray because otherwise this tray could collide with the trays already on the belt.

proc $Cont_2(t : \mathbf{Time}, x_s : \mathbb{R}, f : Torque) =$

$$\overline{\Sigma}_{u:\mathbf{Time},x_0:\theta_0}\ dep_c{}^{\backprime}u\ Cont_1^{nac,am}(u, x_0(u), f) \triangleleft x_0'(\zeta) = bf \wedge x_s = x_0(t) \wedge 0 \leq x_0(u) \leq v_{max} \triangleright \delta{}^{\backprime}\mathbf{0} +$$

$$\overline{\Sigma}_{u:\mathbf{Time},x_0:\theta_0}\ F_c(0){}^{\backprime}u\ Cont_2(u, v_{max}, 0) \triangleleft x_0'(\zeta) = bf \wedge x_s = x_0(t) \wedge x_0(u) = v_{max} \wedge f \neq 0 \triangleright \delta{}^{\backprime}\mathbf{0}$$

The task the controller has to perform can be regarded as a rather simple task, yet, already it is interesting to see that this translates to a rather complex process, which is hard to check by hand. The essential observation here is that the main task of the controller should be, by adjusting the torque of the motor at appropriate times, to maintain the following expression invariant:

$$\forall_{t:\mathbf{Time}} 0 \leq v(t) \leq v_{max} \tag{2.5}$$

where $v(t)$ represents the velocity of the belt at time t. This invariant is actually used in the process descriptions to synthesise the controller.

2.4 Simplification of the hybrid models

A closer inspection of the timed μCRL process descriptions given in the previous section yields the observation that some simplifications can be done. Taking a closer look to process $Belt_1$, it is easy to notice that the following condition occurs in every alternative:

$$x_1'(\zeta) = Ax_1(\zeta) + Bf \wedge x_s = x_1(t)$$

Now, what immediately comes to mind is some elementary mathematics, which basically state that when we have an infinite number of functions defined by:

$$x_1'(\zeta) = Ax_1(\zeta) + Bf \wedge x_1(t) = x_s,$$

in most cases there is exactly one function that satisfies these requirements, and serves as the solution for this function space. Since the used functions are all in C^1, this definitely holds for the function described above, as some elementary calculations show:

$$\begin{aligned}
& x_1'(\zeta) = Ax_1(\zeta) + Bf \wedge x_1(t) = x_s \\
\equiv\ & \{ \text{ definition of } A \text{ and } B\text{, write } x_1 \text{ as the vector } (x, y) \text{ and } x_s \text{ as the vector } (p, s)\} \\
& \begin{pmatrix} x'(\zeta) \\ y'(\zeta) \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} x(\zeta) \\ y(\zeta) \end{pmatrix} + \begin{pmatrix} 0 \\ b \end{pmatrix} f \wedge \begin{pmatrix} x(t) \\ y(t) \end{pmatrix} = \begin{pmatrix} p \\ s \end{pmatrix} \\
\equiv\ & \{ \text{ Matrix calculus, rewrite to a set of equations } \} \\
& \begin{cases} x'(\zeta) = y(\zeta) \wedge x(t) = p \\ y'(\zeta) = bf \wedge y(t) = s \end{cases} \\
\equiv\ & \{ \text{ differential calculus } \} \\
& \begin{cases} x'(\zeta) = y(\zeta) \wedge x(t) = p \\ y(\zeta) = bf\zeta + C_1 \wedge y(t) = s \end{cases} \\
\Rightarrow\ & \{ \text{ Calculus: instantiation of } t \text{ for } \zeta \ \} \\
& \begin{cases} x'(\zeta) = y(\zeta) \wedge x(t) = p \\ y(\zeta) = bf(\zeta - t) + s \end{cases} \\
\equiv\ & \{ \text{ substitution of } y(\zeta)\} \\
& \begin{cases} x'(\zeta) = bf(\zeta - t) + s \wedge x(t) = p \\ y(\zeta) = bf(\zeta - t) + s \end{cases}
\end{aligned}$$

$\equiv$ { differential calculus }

$$\begin{cases} x(\zeta) = \frac{1}{2}bf\zeta^2 - (bft - s)\zeta + C_2 \wedge x(t) = p \\ y(\zeta) = bf(\zeta - t) + s \end{cases}$$

$\Rightarrow$ { Calculus: instantiation of t for ζ }

$$\begin{cases} x(\zeta) = (\zeta - t)(\frac{1}{2}bf(\zeta - t) + s) + p \\ y(\zeta) = bf(\zeta - t) + s \end{cases}$$

□

Naturally, similar calculations can be done for the functions occurring in processes $Belt_0, Belt_2$ and all controller ($Cont_i$) processes. Using these simplifications in combination with the Sum Elimination Theorem (see [12]):

$$\Sigma_{d:D}\ p \lhd d = e \rhd \delta\text{‘}\mathbf{0} \ = \ p[e/d],$$

which states that in this case the quantification over the infinite number of functions in the models can be eliminated. Another application of the Sum Elimination Theorem removes several quantifications over the **Time** domain as well.

2.4.1 A simplified hybrid model for the belt

After the application of the Sum Elimination Theorem, as explained in the previous section, this section describes the processes that define the operation of the belt. For convenience, some alpha conversion is applied to the processes: the different vectors x_s are replaced by ordinary variables of type $\mathbb{R}$. The variable a is replacing bf, since this has become a common factor representing the acceleration. The variable f in the process definitions is eliminated.

proc $Belt'_0(t : \mathbf{Time}, s, a : \mathbb{R}) =$

$$\Sigma_{u:\mathbf{Time}}\ ar_b\text{‘}u\ Belt'_1(u, 0, a(u - t) + s, a)\ +$$

$$\overline{\Sigma}_{u:\mathbf{Time},\gamma:Torque}\ F_b(\gamma)\text{‘}u\ Belt'_0(u, a(u - t) + s, b\gamma)$$

proc $Belt'_1(t : \mathbf{Time}, x, s, a : \mathbb{R}) =$

$$mid_b\text{‘}(t + \tfrac{1}{a}(\sqrt{s^2 + a(l - 2x)} - s))\ Belt'_1(t + \tfrac{1}{a}(\sqrt{s^2 + a(l - 2x)} - s), \tfrac{1}{2}l, \sqrt{s^2 + a(l - 2x)}, a) \\ \lhd a \neq 0 \rhd \delta\text{‘}\mathbf{0} +$$

$$mid_b\text{‘}(t + \tfrac{1}{2s}(l - 2x))\ \ Belt'_1(t + \tfrac{1}{2s}(l - 2x), \tfrac{1}{2}l, s, a) \lhd a = 0 \wedge s \neq 0 \rhd \delta\text{‘}\mathbf{0} +$$

$$dep_b\text{‘}(t + \tfrac{1}{a}(\sqrt{s^2 + 2a(l - x)} - s))\ \ Belt'_0(t + \tfrac{1}{a}(\sqrt{s^2 + 2a(l - x)} - s), \sqrt{s^2 + 2a(l - x)}, a) \\ \lhd a \neq 0 \rhd \delta\text{‘}\mathbf{0} +$$

$$dep_b\text{‘}(t + \tfrac{1}{s}(l - x))\ \ Belt'_0(t + \tfrac{1}{s}(l - x), s, a) \lhd a = 0 \wedge s \neq 0 \rhd \delta\text{‘}\mathbf{0} +$$

$$\Sigma_{u:\mathbf{Time}}\ ar_b\text{‘}u\ Belt'_2(u, (u - t)(\tfrac{1}{2}a(u - t) + s) + x, a(u - t) + s, 0, a) +$$

$$\overline{\Sigma}_{u:\mathbf{Time},\gamma:Torque}\ F_b(\gamma)\text{‘}u\ Belt'_1(u, (u - t)(\tfrac{1}{2}a(u - t) + s) + x, a(u - t) + s, b\gamma)$$

proc $Belt'_2(t : \mathbf{Time}, x, s, y, a : \mathbb{R}) =$

$$dep_b\text{‘}(t + \tfrac{1}{a}(\sqrt{s^2 + 2a(l - x)} - s))\ \ Belt'_1(t + \tfrac{1}{a}(\sqrt{s^2 + 2a(l - x)} - s), y + (l - x), \sqrt{s^2 + 2a(l - x)}, a) \\ \lhd a \neq 0 \rhd \delta\text{‘}\mathbf{0} +$$

$$dep_b\text{‘}(t + \tfrac{1}{s}(l - x)) \;\; Belt'_1(t + \tfrac{1}{s}(l - x), y + (l - x), s, a) \lhd a = 0 \wedge s \neq 0 \rhd \delta\text{‘}\mathbf{0} +$$

$$\overline{\Sigma}_{u:\mathbf{Time},\gamma:Torque} \; F_b(\gamma)\text{‘}u$$
$$\quad Belt'_2(u, (u - t)(\tfrac{1}{2}a(u - t) + s) + x, a(u - t) + s, (u - t)(\tfrac{1}{2}a(u - t) + s) + y, b\gamma)$$

The changes with respect to the model described in section 2.3.1 are obvious: on the one hand the conditions in the process descriptions have become elementary, and on the other hand, the number of alternatives decreased. Only a few quantifications over the time domain are left, yet, the actions that relate to these quantifications are not in the category of actions that are affected by the continuous dynamics of the belt. One of the essentials is that the understanding of the belt is more intuitive using the description above.

2.4.2 A simplified hybrid model for the controller

A similar exercise as for the belt is undertaken for the controller. Since only the torque is adjusted by the controller, the process looks rather similar to the one defined in section 2.3.2. Also, the variable f in the process definitions is eliminated and the variable a is introduced to replace the more commonly occurring factor bf. The speed x_s is now replaced by the variable s.

$$\mathbf{proc}\; Cont'^{nd}_0(t : \mathbf{Time}, s, a : \mathbb{R}) =$$

$$\Sigma_{u:\mathbf{Time}} \; ar_c\text{‘}u \; Cont'^{nac,ds}_1(u, a(u - t) + s, a) \lhd 0 \leq a(u - t) + s \leq v_{max} \rhd \delta\text{‘}\mathbf{0} +$$

$$F_c(0)\text{‘}(t - \tfrac{s}{a}) \; Cont'^{nd}_0(t - \tfrac{s}{a}, 0, 0) \lhd a \neq 0 \rhd \delta\text{‘}\mathbf{0}$$

$$\mathbf{proc}\; Cont'^{d}_0(t, d : \mathbf{Time}, s, a : \mathbb{R}) =$$

$$\Sigma_{u:\mathbf{Time}} \; ar_c\text{‘}u \; Cont'^{nac,am}_1(u, a(u - t) + s, a) \lhd 0 \leq a(u - t) + s \leq v_{max} \wedge u \leq t + d \rhd \delta\text{‘}\mathbf{0} +$$

$$F_c(0)\text{‘}(t + \tfrac{1}{a}(v_{max} - s)) \; Cont'^{d}_0(t + \tfrac{1}{a}(v_{max} - s), d - \tfrac{1}{a}(v_{max} - s), v_{max}, 0)$$
$$\quad \lhd a \neq 0 \wedge \tfrac{1}{a}(v_{max} - s) < d \rhd \delta\text{‘}\mathbf{0} +$$

$$F_c(-T_{max})\text{‘}(t + d) \; Cont'^{nd}_0(t + d, ad + s, -bT_{max}) \lhd 0 < ad + s \leq v_{max} \rhd \delta\text{‘}\mathbf{0}$$

$$\mathbf{proc}\; Cont'^{nac,ds}_1(t : \mathbf{Time}, s, a : \mathbb{R}) =$$

$$F_c(0)\text{‘}t \; Cont'^{nac,am}_1(t, v_{max}, 0) \lhd s = v_{max} \rhd \delta\text{‘}\mathbf{0} +$$

$$F_c(T_{max})\text{‘}t \; Cont'^{nac,am}_1(t, s, bT_{max}) \lhd s < v_{max} \rhd \delta\text{‘}\mathbf{0}$$

$$\mathbf{proc}\; Cont'^{nac,am}_1(t : \mathbf{Time}, s, a : \mathbb{R}) =$$

$$\Sigma_{u:\mathbf{Time}} \; mid_c\text{‘}u \; Cont'^{ac}_1(u, a(u - t) + s, a) \lhd 0 \leq a(u - t) + s \leq v_{max} \rhd \delta\text{‘}\mathbf{0} +$$

$$F_c(0)\text{‘}(t + \tfrac{1}{a}(v_{max} - s)) \; Cont'^{nac,am}_1(t + \tfrac{1}{a}(v_{max} - s), v_{max}, 0) \lhd a \neq 0 \rhd \delta\text{‘}\mathbf{0}$$

$$\mathbf{proc}\; Cont'^{ac}_1(t : \mathbf{Time}, s, a : \mathbb{R}) =$$

$$\Sigma_{u:\mathbf{Time}} \; ar_c\text{‘}u \; Cont'_2(u, a(u - t) + s, a) \lhd 0 \leq a(u - t) + s \leq v_{max} \rhd \delta\text{‘}\mathbf{0} +$$

$$\Sigma_{u:\mathbf{Time}} \; dep_c\text{‘}u \; Cont'^{d}_0(u, k, a(u - t) + s, a) \lhd 0 \leq a(u - t) + s \leq v_{max} \rhd \delta\text{‘}\mathbf{0} +$$

$$F_c(0)\text{‘}(t + \tfrac{1}{a}(v_{max} - s)) \; Cont'^{ac}_1(t + \tfrac{1}{a}(v_{max} - s), v_{max}, 0) \lhd a \neq 0 \rhd \delta\text{‘}\mathbf{0}$$

proc $Cont'_2(t : \mathbf{Time}, s, a : \mathbb{R}) =$

$$\Sigma_{u:\mathbf{Time}}\ dep_c{}^{\varsigma}u\ Cont'^{nac,am}_{1}(u, a(u-t)+s, a) \lhd 0 \leq a(u-t)+s \leq v_{max} \rhd \delta{}^{\varsigma}\mathbf{0} +$$

$$F_c(0){}^{\varsigma}(t + \tfrac{1}{a}(v_{max} - s))\ Cont'_2(t + \tfrac{1}{a}(v_{max} - s), v_{max}, 0) \lhd a \neq 0 \rhd \delta{}^{\varsigma}\mathbf{0}$$

The observation, made earlier with respect to the model of the belt, still holds, i.e. after elimination of the quantifiers, the model gets more intuitive, and the conditions become elementary. Surprising, however, is the still very large descriptions compared to the rather simple nature of the problem. Perhaps this is also the pitfall for many systems that are to be automated: simple ideas turn out to behave in a complex manner.

3 Composition of the Conveyor Belt System

Until now, the focus has been on the individual components of the conveyor belt system. However, the interesting part is the co-operation between these independent parts, together defining the conveyor belt system. To this end, it is essential that the communications between the belt and the controller are defined. These communications are then what make up the conveyor belt system.

3.1 Defining the communications

This section will go into more detail in the subject of the communication between the controller and the belt. The choices made in this section are an obvious extension to the models and their design criteria as they are defined in sections 2.4.1 and 2.4.2.

The events that need to be synchronised, in order to have a correct functioning of the conveyor belt system, are the following:

- The arrival of a new tray at the beginning of the belt.
- The arrival of a tray at the mid point of the belt.
- The departure of a tray from the belt.
- The setting of the torque of the motor to the belt.

In short: for every action that occurs at the controller, there should be an action that occurs at the belt at the same time and vice versa. To this end, the communications are defined as follows:

comm

$$\begin{array}{rcl} ar & = & ar_b \mid ar_c \\ mid & = & mid_b \mid mid_c \\ dep & = & dep_b \mid dep_c \\ F(\gamma) & = & F_b(\gamma) \mid F_c(\gamma) \end{array}$$

These communications define that if both actions can occur at the same time, they can communicate, in which case a new action is visible. However, it is also possible that actions such as ar_b and ar_c can occur individually, i.e. without first synchronising. The set of actions that are not allowed to take place individually is: $H = \{ar_b, ar_c, mid_b, mid_c, dep_b, dep_c, F_b, F_c\}$. Using the encapsulation operator, the actions in set H are captured and translated to deadlocks, meaning they will not occur.

3.2 A hybrid model for the Conveyor Belt System

In the formalism timed μCRL, it is possible to compose a new process by putting other processes in parallel to each other. Since this is essentially what will happen with the controller and the belt in real life, it is an obvious choice to define the hybrid model for the conveyor belt system as the parallel composition of the models for the belt and the controller. If we take the communications as defined in section 3.1, and the set of actions, defined in H, as the set of actions that need to be encapsulated (i.e. we do not want to see them occurring individually), the model for the conveyor belt system is defined by:

$$\textbf{proc } CBS(t_b, t_c : \textbf{Time}, s_b, s_c, a_b, a_c : \mathbb{R}) =$$

$$\partial_H(Belt'_0(t_b, s_b, a_b) \mid\mid Cont'^{nd}_0(t_c, s_c, a_c)\ )$$

This process will be the subject of investigation in the subsequent sections.

3.3 Expansion of the hybrid model for the conveyor belt system

In order to do some elementary calculations within the theory of timed μCRL on the process CBS, this process must first be expanded. In this case, expansion means that the parallel composition has to be eliminated. A rather straightforward application of the Expansion and Encapsulation Theorem (see [12]), however, is not possible due to the fact that for instance processes $Belt'_0$ and $Cont'^{nd}_0$ are not in the form of a *linear process expression (LPE)* (also see [12]). The next sections are devoted to the transformation of the processes $Belt'_0$ and $Cont'^{nd}_0$ into LPE's.

3.3.1 Linearisation of process $Belt'_0$

Linearising a process is a rather straightforward task, especially when the processes already resemble the LPE format. For untimed processes, this procedure is already automated, however, for timed processes no such tool exists yet. In case of process $Belt'_0$, this means that all data should be united into a single process $Belt$. We take the set $State = \{0,1,2\}$ to represent the number of trays on the belt. This allows for a rather straightforward translation to the following process:

$\textbf{proc } Belt(\sigma : State, t : \textbf{Time}, x, s, y, a : \mathbb{R}) =$

(B1) $\Sigma_{u:\textbf{Time}}\ ar_b{}^{\backprime}u\ Belt(1, u, 0, a(u-t)+s, y, a)$
$\quad \triangleleft \sigma = 0 \triangleright \delta{}^{\backprime}\mathbf{0}+$
(B2) $mid_b{}^{\backprime}(t+\frac{1}{a}(\sqrt{s^2+a(l-2x)}-s))\ Belt(1, t+\frac{1}{a}(\sqrt{s^2+a(l-2x)}-s), \frac{1}{2}l, \sqrt{s^2+a(l-2x)}, y, a)$
$\quad \triangleleft \sigma = 1 \wedge a \neq 0 \triangleright \delta{}^{\backprime}\mathbf{0}+$
(B3) $mid_b{}^{\backprime}(t+\frac{1}{2s}(l-2x))\ Belt(1, t+\frac{1}{2s}(l-2x), \frac{1}{2}l, s, y, a)$
$\quad \triangleleft \sigma = 1 \wedge a = 0 \wedge s \neq 0 \triangleright \delta{}^{\backprime}\mathbf{0}+$
(B4) $dep_b{}^{\backprime}(t+\frac{1}{a}(\sqrt{s^2+2a(l-x)}-s))\ Belt(0, t+\frac{1}{a}(\sqrt{s^2+2a(l-x)}-s), x, \sqrt{s^2+2a(l-x)}, y, a)$
$\quad \triangleleft \sigma = 1 \wedge a \neq 0 \triangleright \delta{}^{\backprime}\mathbf{0}+$
(B5) $dep_b{}^{\backprime}(t+\frac{1}{s}(l-x))\ Belt(0, t+\frac{1}{s}(l-x), x, s, y, a)$
$\quad \triangleleft \sigma = 1 \wedge a = 0 \wedge s \neq 0 \triangleright \delta{}^{\backprime}\mathbf{0}+$
(B6) $\Sigma_{u:\textbf{Time}}\ ar_b{}^{\backprime}u\ Belt(2, u, (u-t)(\frac{1}{2}a(u-t)+s)+x, a(u-t)+s, 0, a)$
$\quad \triangleleft \sigma = 1 \triangleright \delta{}^{\backprime}\mathbf{0}+$
(B7) $dep_b{}^{\backprime}(t+\frac{1}{a}(\sqrt{s^2+2a(l-x)}-s))$
$\quad Belt(1, t+\frac{1}{a}(\sqrt{s^2+2a(l-x)}-s), y+(l-x), \sqrt{s^2+2a(l-x)}, y, a)$
$\quad \triangleleft \sigma = 2 \wedge a \neq 0 \triangleright \delta{}^{\backprime}\mathbf{0}+$
(B8) $dep_b{}^{\backprime}(t+\frac{1}{s}(l-x))\ Belt(1, t+\frac{1}{s}(l-x), y+(l-x), s, y, a)$
$\quad \triangleleft \sigma = 2 \wedge a = 0 \wedge s \neq 0 \triangleright \delta{}^{\backprime}\mathbf{0}+$
(B9) $\Sigma_{u:\textbf{Time},\gamma:Torque}\ F_b(\gamma){}^{\backprime}u$
$\quad Belt(\sigma, u, (u-t)(\frac{1}{2}a(u-t)+s)+x, a(u-t)+s, (u-t)(\frac{1}{2}a(u-t)+s)+y, b_2\gamma)$
$\quad \triangleleft \sigma \in \{0,1,2\} \triangleright \delta{}^{\backprime}\mathbf{0}$

The relation between $Belt$ and $Belt'_0$ is characterised as follows: $Belt'_0(t_b, s_b, a_b) = Belt(0, t_b, x, s_b, y, a_b)$. Here, the values for the parameters x and y can be chosen arbitrarily. Although this model is easier to compute with, it does not reflect any elegance: intuition is traded for ease of calculation.

3.3.2 Linearisation of process $Cont'^{nd}_0$

The linearisation of process $Cont'^{nd}_0$ is a bit more elaborate. The problem is the finer grained process descriptions: instead of for instance one process $Cont'_0$, there are two processes, viz: $Cont'^{d}_0$ and $Cont'^{nd}_0$ that are to be distinguished. In order to deal with this complication, three alternative sets are introduced: $DState = \{del, nod\}$, $Motion = \{ds, am\}$ and $Accept = \{ac, nac\}$. Here del is taken to represent the *delaying* state of process $Cont'_0$ and nod is the *not delaying* state of process $Cont'_0$. Furthermore, ds is taken to represent the *decelerating or stand still* state of process $Cont'_1$ and am is taken to represent the *accelerating or maximal speed* state of process $Cont'_1$. Finally, ac is taken to represent the state in which process $Cont'_1$ is able to *accept* a new tray and nac is taken to represent the state in which process $Cont'_1$ is not able to accept a new tray. A straightforward translation to an LPE yields the following process:

proc $Cont(\sigma : State, t, d : \mathbf{Time}, s, a : \mathbb{R}, \sigma_d : DState, m : Motion, n : Accept) =$

$$\begin{aligned}&(\mathrm{C1})\ \Sigma_{u:\mathbf{Time}}\ ar_c{}^{\backprime}u\ Cont(1, u, d, a(u-t)+s, a, \sigma_d, ds, nac)\\&\qquad \triangleleft\, \sigma = 0 \wedge 0 \leq a(u-t)+s \leq v_{max} \wedge \sigma_d = nod \triangleright \delta{}^{\backprime}\mathbf{0} +\\&(\mathrm{C2})\ F_c(0){}^{\backprime}(t - \tfrac{s}{a})\ Cont(0, t - \tfrac{s}{a}, d, 0, 0, nod, m, n)\\&\qquad \triangleleft\, \sigma = 0 \wedge a \neq 0 \wedge \sigma_d = nod \triangleright \delta{}^{\backprime}\mathbf{0} +\\&(\mathrm{C3})\ \Sigma_{u:\mathbf{Time}}\ ar_c{}^{\backprime}u\ Cont(1, u, d, a(u-t)+s, a, \sigma_d, am, nac)\\&\qquad \triangleleft\, \sigma = 0 \wedge 0 \leq a(u-t)+s \leq v_{max} \wedge \sigma_d = del \wedge u \leq t+d \triangleright \delta{}^{\backprime}\mathbf{0} +\\&(\mathrm{C4})\ F_c(0){}^{\backprime}(t + \tfrac{1}{a}(v_{max} - s))\ Cont(0, t + \tfrac{1}{a}(v_{max} - s), d - \tfrac{1}{a}(v_{max} - s), v_{max}, 0, del, m, n)\\&\qquad \triangleleft\, \sigma = 0 \wedge a \neq 0 \wedge \tfrac{1}{a}(v_{max} - s) < d \wedge \sigma_d = del \triangleright \delta{}^{\backprime}\mathbf{0} +\\&(\mathrm{C5})\ F_c(-T_{max}){}^{\backprime}(t+d)\ Cont(0, t+d, d, ad+s, -b_2T_{max}, nod, m, n)\\&\qquad \triangleleft\, \sigma = 0 \wedge 0 < ad+s \leq v_{max} \wedge \sigma_d = del \triangleright \delta{}^{\backprime}\mathbf{0} +\\&(\mathrm{C6})\ F_c(0){}^{\backprime}t\ Cont(1, t, d, s, 0, \sigma_d, am, nac)\\&\qquad \triangleleft\, \sigma = 1 \wedge m = ds \wedge n = nac \wedge s = v_{max} \triangleright \delta{}^{\backprime}\mathbf{0} +\\&(\mathrm{C7})\ F_c(T_{max}){}^{\backprime}t\ Cont(1, t, d, s, b_2T_{max}, \sigma_d, am, nac)\\&\qquad \triangleleft\, \sigma = 1 \wedge m = ds \wedge n = nac \wedge s < v_{max} \triangleright \delta{}^{\backprime}\mathbf{0} +\\&(\mathrm{C8})\ \Sigma_{u:\mathbf{Time}}\ mid_c{}^{\backprime}u\ Cont(1, u, d, a(u-t)+s, a, \sigma_d, m, ac)\\&\qquad \triangleleft\, \sigma = 1 \wedge m = am \wedge n = nac \wedge 0 \leq a(u-t)+s \leq v_{max} \triangleright \delta{}^{\backprime}\mathbf{0} +\\&(\mathrm{C9})\ F_c(0){}^{\backprime}(t + \tfrac{1}{a}(v_{max} - s))\ Cont(1, t + \tfrac{1}{a}(v_{max} - s), d, v_{max}, 0, \sigma_d, am, nac)\\&\qquad \triangleleft\, \sigma = 1 \wedge m = am \wedge n = nac \wedge a \neq 0 \triangleright \delta{}^{\backprime}\mathbf{0} +\\&(\mathrm{C10})\ \Sigma_{u:\mathbf{Time}}\ ar_c{}^{\backprime}u\ Cont(2, u, d, a(u-t)+s, a, \sigma_d, m, n)\\&\qquad \triangleleft\, \sigma = 1 \wedge n = ac \wedge 0 \leq a(u-t)+s \leq v_{max} \triangleright \delta{}^{\backprime}\mathbf{0} +\\&(\mathrm{C11})\ \Sigma_{u:\mathbf{Time}}\ dep_c{}^{\backprime}u\ Cont(0, u, k, a(u-t)+s, a, del, m, n)\\&\qquad \triangleleft\, \sigma = 1 \wedge n = ac \wedge 0 \leq a(u-t)+s \leq v_{max} \triangleright \delta{}^{\backprime}\mathbf{0} +\\&(\mathrm{C12})\ F_c(0){}^{\backprime}(t + \tfrac{1}{a}(v_{max} - s))\ Cont(1, t + \tfrac{1}{a}(v_{max} - s), d, v_{max}, 0, \sigma_d, m, ac)\\&\qquad \triangleleft\, \sigma_c = 1 \wedge n = ac \wedge a \neq 0 \triangleright \delta{}^{\backprime}\mathbf{0} +\\&(\mathrm{C13})\ \Sigma_{u:\mathbf{Time}}\ dep_c{}^{\backprime}u\ Cont(1, u, d, a(u-t)+s, a, \sigma_d, am, nac)\\&\qquad \triangleleft\, \sigma_c = 2 \wedge 0 \leq a(u-t)+s \leq v_{max} \triangleright \delta{}^{\backprime}\mathbf{0} +\\&(\mathrm{C14})\ F_c(0){}^{\backprime}(t + \tfrac{1}{a}(v_{max} - s_c))\ Cont(2, t + \tfrac{1}{a}(v_{max} - s), d, v_{max}, \sigma_d, m, n)\\&\qquad \triangleleft\, \sigma_c = 2 \wedge a \neq 0 \triangleright \delta{}^{\backprime}\mathbf{0}\end{aligned}$$

The relation between the process $Cont^{nd}_0$ and the process $Cont$ is defined as follows: $Cont^{nd}_0(t_c, s_c, a_c) = Cont(0, t_c, d, s_c, a_c, nod, m, n)$, where the parameters d, m and n have arbitrary appropriate values.

3.3.3 Redefining process CBS

The processes *Belt* and *Cont*, described in the previous sections are the subject of further investigation. Since process CBS is defined in terms of the processes $Belt'_0$ and $Cont^{nd}_0$, also process CBS needs rewriting. Substitution of the processes *Belt* and *Cont* in the definition of the process

CBS yields the following (equivalent) process CBS':

proc $CBS'(\sigma_b, \sigma_c : State, t_b, t_c, d : \mathbf{Time}, x, s_b, y, s_c, a_b, a_c : \mathbb{R},$
$\sigma_d : DState, m : Motion, n : Accept) =$
$\partial_H(Belt(0, t_b, x, s_b, y, a_b) \parallel Cont(0, t_c, d, s_c, a_b, nod, m, n))$

3.3.4 Calculating the communications of process CBS'

Using the expansion and encapsulation theorem, defined in [12], the communications can easily be calculated. To this end, the communications defined in section 3.1 are used. The condition under which these communications can occur are determined by the conjunct of the conditions under which both actions can occur separately. Moreover, the time at which the communication can occur is also determined by the intersection of the times the individual actions can occur.

B1,C1:
(1) $\Sigma_{u:\mathbf{Time}}\ ar\text{`}u$
$CBS(1, 1, u, u, d, 0, a_b(u - t_b) + s_b, y, a_c(u - t_c) + s_c, a_b, a_c, \sigma_d, ds, nac)$
$\lhd \sigma_b = \sigma_c = 0 \wedge 0 \leq a_c(u - t_c) + s_c \leq v_{max} \wedge \sigma_d = nod \rhd \delta\text{`}\mathbf{0+}$

B1,C3:
(2) $\Sigma_{u:\mathbf{Time}}\ ar\text{`}u$
$CBS(1, 1, u, u, d, 0, a_b(u - t_b) + s_b, y, a_c(u - t_c) + s_c, a_b, a_c, \sigma_d, am, nac)$
$\lhd \sigma_b = \sigma_c = 0 \wedge 0 \leq a_c(u - t_c) + s_c \leq v_{max} \wedge \sigma_d = del \wedge u \leq t_c + d \rhd \delta\text{`}\mathbf{0+}$

B1,C10:
(3) $\Sigma_{u:\mathbf{Time}}\ ar\text{`}u$
$CBS(1, 2, u, u, d, 0, a_b(u - t_b) + s_b, y, a_c(u - t_c) + s_c, a_b, a_c, \sigma_d, m, n)$
$\lhd \sigma_b = 0 \wedge \sigma_c = 1 \wedge 0 \leq a_c(u - t_c) + s_c \leq v_{max} \wedge n = ac \rhd \delta\text{`}\mathbf{0+}$

B2,C8:
(4) $mid\text{`}(t_b + \frac{1}{a_b}(\sqrt{s_b^2 + a_b(l - 2x)} - s_b))$
$CBS(1, 1, t_b + \frac{1}{a_b}(\sqrt{s_b^2 + a_b(l - 2x)} - s_b), t_b + \frac{1}{a_b}(\sqrt{s_b^2 + a_b(l - 2x)} - s_b), d, \frac{1}{2}l,$
$\sqrt{s_b^2 + a_b(l - 2x)}, y, a_c(t_b + \frac{1}{a_b}(\sqrt{s_b^2 + a_b(l - 2x)} - s_b) - t_c) + s_c, a_b, a_c, \sigma_d, m, ac)$
$\lhd \sigma_b = \sigma_c = 1 \wedge a_b \neq 0 \wedge m = am \wedge n = nac \wedge$
$0 \leq a_c(t_b + \frac{1}{a_b}(\sqrt{s_b^2 + a_b(l - 2x)} - s_b) - t_c) + s_c \leq v_{max} \rhd \delta\text{`}\mathbf{0+}$

B3,C8:
(5) $mid\text{`}(t_b + \frac{1}{2s_b}(l - 2x))$
$CBS(1, 1, t_b + \frac{1}{2s_b}(l - 2x), t_b + \frac{1}{2s_b}(l - 2x), d, \frac{1}{2}l, s_b, y, a_c(t_b + \frac{1}{2s_b}(l - 2x) - t_c) + s_c, a_b, a_c, \sigma_d, m, ac)$
$\lhd \sigma_b = \sigma_c = 1 \wedge a_b = 0 \wedge s_b \neq 0 \wedge m = am \wedge n = nac \wedge$
$0 \leq a_c(t_b + \frac{1}{2s_b}(l - 2x) - t_c) + s_c \leq v_{max} \rhd \delta\text{`}\mathbf{0+}$

B4,C11:
(6) $dep\text{`}(t_b + \frac{1}{a_b}(\sqrt{s_b^2 + 2a_b(l - x)} - s_b))$
$CBS(0, 0, t_b + \frac{1}{a_b}(\sqrt{s_b^2 + 2a_b(l - x)} - s_b), t_b + \frac{1}{a_b}(\sqrt{s_b^2 + 2a_b(l - x)} - s_b), k, x,$
$\sqrt{s_b^2 + 2a_b(l - x)}, y, a_c(t_b + \frac{1}{a_b}(\sqrt{s_b^2 + 2a_b(l - x)} - s_b) - t_c) + s_c, a_b, a_c, del, m, n)$
$\lhd \sigma_b = \sigma_c = 1 \wedge a_b \neq 0 \wedge n = ac \wedge 0 \leq a_c(t_b + \frac{1}{a_b}(\sqrt{s_b^2 + 2a_b(l - x)} - s_b) - t_c) + s_c \leq v_{max} \rhd \delta\text{`}\mathbf{0+}$

B4,C13:
(7) $dep\text{`}(t_b + \frac{1}{a_b}(\sqrt{s_b^2 + 2a_b(l - x)} - s_b))$
$CBS(0, 1, t_b + \frac{1}{a_b}(\sqrt{s_b^2 + 2a_b(l - x)} - s_b), t_b + \frac{1}{a_b}(\sqrt{s_b^2 + 2a_b(l - x)} - s_b), d, x,$
$\sqrt{s_b^2 + 2a_b(l - x)}, y, a_c(t_b + \frac{1}{a_b}(\sqrt{s_b^2 + 2a_b(l - x)} - s_b) - t_c) + s_c, a_b, a_c, \sigma_d, am, nac)$

$\triangleleft \sigma_b = 1 \wedge \sigma_c = 2 \wedge a_b \neq 0 \wedge 0 \leq a_c(t_b + \frac{1}{a_b}(\sqrt{s_b^2 + 2a_b(l-x)} - s_b) - t_c) + s_c \leq v_{max} \triangleright \delta‘\mathbf{0}+$

B5,C11:
(8) $dep‘(t_b + \frac{1}{s_b}(l-x))$
$CBS(0,0,t_b + \frac{1}{s_b}(l-x), t_b + \frac{1}{s_b}(l-x), k, x, s_b, y, a_c(t_b + \frac{1}{s_b}(l-x) - t_c) + s_c, a_b, a_c, del, m, n)$
$\triangleleft \sigma_b = \sigma_c = 1 \wedge a_b = 0 \wedge s_b \neq 0 \wedge n = ac \wedge 0 \leq a_c(t_b + \frac{1}{s_b}(l-x) - t_c) + s_c \leq v_{max} \triangleright \delta‘\mathbf{0}+$

B5,C13:
(9) $dep‘(t_b + \frac{1}{s_b}(l-x))$
$CBS(0,1,t_b + \frac{1}{s_b}(l-x), t_b + \frac{1}{s_b}(l-x), d, x, s_b, y, a_c(t_b + \frac{1}{s_b}(l-x) - t_c) + s_c, a_b, a_c, \sigma_d, am, nac)$
$\triangleleft \sigma_b = 1 \wedge \sigma_c = 2 \wedge a_b = 0 \wedge s_b \neq 0 \wedge 0 \leq a_c(t_b + \frac{1}{s_b}(l-x) - t_c) + s_c \leq v_{max} \triangleright \delta‘\mathbf{0}+$

B6,C1:
(10)$\Sigma_{u:\mathbf{Time}} ar‘u$
$CBS(2,1,u,u,d,(u-t_b)(\frac{1}{2}a_b(u-t_b)+s_b)+x, a_b(u-t_b)+s_b, 0, a_c(u-t_c)+s_b, a_b, a_c, \sigma_d, ds, nac)$
$\triangleleft \sigma_b = 1 \wedge \sigma_c = 0 \wedge \sigma_d = nod \wedge 0 \leq a_c(u - t_c) + s_c \leq v_{max} \triangleright \delta‘\mathbf{0}+$

B6,C3:
(11)$\Sigma_{u:\mathbf{Time}} ar‘u$
$CBS(2,1,u,u,d,(u-t_b)(\frac{1}{2}a_b(u-t_b)+s_b)+x, a_b(u-t_b)+s_b, 0, a_c(u-t_c)+s_b, a_b, a_c, \sigma_d, am, nac)$
$\triangleleft \sigma_b = 1 \wedge \sigma_c = 0 \wedge \sigma_d = del \wedge u \leq d + t_c \wedge 0 \leq a_c(u - t_c) + s_c \leq v_{max} \triangleright \delta‘\mathbf{0}+$

B6,C10:
(12)$\Sigma_{u:\mathbf{Time}} ar‘u$
$CBS(2,2,u,u,d,(u-t_b)(\frac{1}{2}a_b(u-t_b)+s_b)+x, a_b(u-t_b)+s_b, 0, a_c(u-t_c)+s_b, a_b, a_c, \sigma_d, m, n)$
$\triangleleft \sigma_b = \sigma_c = 1 \wedge n = ac \wedge 0 \leq a_c(u - t_c) + s_c \leq v_{max} \triangleright \delta‘\mathbf{0}+$

B7,C11:
(13)$dep‘(t_b + \frac{1}{a_b}(\sqrt{s_b^2 + 2a_b(l-x)} - s_b))$
$CBS(1,0,t_b + \frac{1}{a_b}(\sqrt{s_b^2 + 2a_b(l-x)} - s_b), t_b + \frac{1}{a_b}(\sqrt{s_b^2 + 2a_b(l-x)} - s_b), k, y + (l-x),$
$\sqrt{s_b^2 + 2a_b(l-x)}, y, a_c(t_b + \frac{1}{a_b}(\sqrt{s_b^2 + 2a_b(l-x)} - s_b) - t_c) + s_c, a_b, a_c, del, m, n)$
$\triangleleft \sigma_b = 2 \wedge \sigma_c = 1 \wedge a_b \neq 0 \wedge n = ac \wedge$
$0 \leq a_c(t_b + \frac{1}{a_b}(\sqrt{s_b^2 + 2a_b(l-x)} - s_b) - t_c) + s_c \leq v_{max} \triangleright \delta‘\mathbf{0}+$

B7,C13:
(14)$dep‘(t_b + \frac{1}{a_b}(\sqrt{s_b^2 + 2a_b(l-x)} - s_b))$
$CBS(1,1,t_b + \frac{1}{a_b}(\sqrt{s_b^2 + 2a_b(l-x)} - s_b), t_b + \frac{1}{a_b}(\sqrt{s_b^2 + 2a_b(l-x)} - s_b), d, y + (l-x),$
$\sqrt{s_b^2 + 2a_b(l-x)}, y, a_c(t_b + \frac{1}{a_b}(\sqrt{s_b^2 + 2a_b(l-x)} - s_b) - t_c) + s_c, a_b, a_c, \sigma_d, am, nac)$
$\triangleleft \sigma_b = \sigma_c = 2 \wedge a_b \neq 0 \wedge 0 \leq a_c(t_b + \frac{1}{a_b}(\sqrt{s_b^2 + 2a_b(l-x)} - s_b) - t_c) + s_c \leq v_{max} \triangleright \delta‘\mathbf{0}+$

B8,C11:
(15)$dep‘(t_b + \frac{1}{s_b}(l-x))$
$CBS(1,0,t_b + \frac{1}{s_b}(l-x), t_b + \frac{1}{s_b}(l-x), k, y + (l-x), s_b, y, a_c(t_b + \frac{1}{s_b}(l-x) - t_c) + s_c, a_b, a_c, del, m, n)$
$\triangleleft \sigma_b = 2 \wedge \sigma_c = 1 \wedge a_b = 0 \wedge s_b \neq 0 \wedge n = ac \wedge 0 \leq a_c(t_b + \frac{1}{s_b}(l-x) - t_c) + s_c \leq v_{max} \triangleright \delta‘\mathbf{0}+$

B8,C13:
(16)$dep‘(t_b + \frac{1}{s_b}(l-x))$
$CBS(1,1,t_b + \frac{1}{s_b}(l-x), t_b + \frac{1}{s_b}(l-x), d, y + (l-x), s_b, y,$
$a_c(t_b + \frac{1}{s_b}(l-x) - t_c) + s_c, a_b, a_c, \sigma_d, am, nac)$
$\triangleleft \sigma_b = 2 \wedge \sigma_c = 2 \wedge a_b = 0 \wedge s_b \neq 0 \wedge 0 \leq a_c(t_b + \frac{1}{s_b}(l-x) - t_c) + s_c \leq v_{max} \triangleright \delta‘\mathbf{0}+$

B9,C2:

$(17) F(0)\text{‹}(t_c - \frac{s_c}{a_c})$

$CBS(\sigma_b, 0, t_c - \frac{s_c}{a_c}, t_c - \frac{s_c}{a_c}, d, (t_c - \frac{s_c}{a_c} - t_b)(\frac{1}{2}a_b(t_c - \frac{s_c}{a_c} - t_b) + s_b) + x, a_b(t_c - \frac{s_c}{a_c} - t_b) + s_b,$

$(t_c - \frac{s_c}{a_c} - t_b)(\frac{1}{2}a_b(t_c - \frac{s_c}{a_c} - t_b) + s_b) + y, 0, 0, 0, nod, m, n)$

$\lhd \sigma_b \in \{0,1,2\} \land \sigma_c = 0 \land a_c \neq 0 \land \sigma_d = nod \rhd \delta\text{‹}\mathbf{0} +$

B9,C4:

$(18) F(0)\text{‹}(t_c + \frac{1}{a_c}(v_{max} - s_c))$

$CBS(\sigma_b, 0, t_c + \frac{1}{a_c}(v_{max} - s_c), t_c + \frac{1}{a_c}(v_{max} - s_c), d - \frac{1}{a_c}(v_{max} - s_c),$

$(t_c + \frac{1}{a_c}(v_{max} - s_c) - t_b)(\frac{1}{2}a_b(t_c + \frac{1}{a_c}(v_{max} - s_c) - t_b) + s_b) + x, a_b(t_c + \frac{1}{a_c}(v_{max} - s_c) - t_b) + s_b,$

$(t_c + \frac{1}{a_c}(v_{max} - s_c) - t_b)(\frac{1}{2}a_b(t_c + \frac{1}{a_c}(v_{max} - s_c) - t_b) + s_b) + y, v_{max}, 0, 0, del, m, n)$

$\lhd \sigma_b \in \{0,1,2\} \land \sigma_c = 0 \land a_c \neq 0 \land \sigma_d = del \land \frac{1}{a_c}(v_{max} - s_c) < d \rhd \delta\text{‹}\mathbf{0} +$

B9,C5:

$(19) F(-T_{max})\text{‹}(t_c + d)$

$CBS(\sigma_b, 0, t_c + d, t_c + d, d, (t_c + d - t_b)(\frac{1}{2}a_b(t_c + d - t_b) + s_b) + x, a_b(t_c + d - t_b) + s_b,$

$(t_c + d - t_b)(\frac{1}{2}a_b(t_c + d - t_b) + s_b) + y, a_c d + s_c, -b_2 T_{max}, -b_2 T_{max}, nod, m, n)$

$\lhd \sigma_b \in \{0,1,2\} \land \sigma_c = 0 \land \sigma_d = del \land 0 < a_c d + s_c \leq v_{max} \rhd \delta\text{‹}\mathbf{0} +$

B9,C6:

$(20) F(0)\text{‹}t_c$

$CBS(\sigma_b, 1, t_c, t_c, d, (t_c - t_b)(\frac{1}{2}a_b(t_c - t_b) + s_b) + x, a_b(t_c - t_b) + s_b,$

$(t_c - t_b)(\frac{1}{2}a_b(t_c - t_b) + s_b) + y, s_c, 0, 0, \sigma_d, am, nac)$

$\lhd \sigma_b \in \{0,1,2\} \land \sigma_c = 1 \land m = ds \land n = nac \land s_c = v_{max} \rhd \delta\text{‹}\mathbf{0} +$

B9,C7:

$(21) F(T_{max})\text{‹}t_c$

$CBS(\sigma_b, 1, t_c, t_c, d, (t_c - t_b)(\frac{1}{2}a_b(t_c - t_b) + s_b) + x, a_b(t_c - t_b) + s_b, (t_c - t_b)(\frac{1}{2}a_b(t_c - t_b) + s_b) + y,$

$s_c, b_2 T_{max}, b_2 T_{max}, \sigma_d, am, nac)$

$\lhd \sigma_b \in \{0,1,2\} \land \sigma_c = 1 \land m = ds \land n = nac \land s_c < v_{max} \rhd \delta\text{‹}\mathbf{0} +$

B9,C9:

$(22) F(0)\text{‹}(t_c + \frac{1}{a_c}(v_{max} - s_c))$

$CBS(\sigma_b, 1, t_c + \frac{1}{a_c}(v_{max} - s_c), t_c + \frac{1}{a_c}(v_{max} - s_c), d,$

$(t_c + \frac{1}{a_c}(v_{max} - s_c) - t_b)(\frac{1}{2}a_b(t_c + \frac{1}{a_c}(v_{max} - s_c) - t_b) + s_b) + x, a_b(t_c + \frac{1}{a_c}(v_{max} - s_c) - t_b) + s_b,$

$(t_c + \frac{1}{a_c}(v_{max} - s_c) - t_b)(\frac{1}{2}a_b(t_c + \frac{1}{a_c}(v_{max} - s_c) - t_b) + s_b) + y, v_{max}, 0, 0, \sigma_d, am, nac)$

$\lhd \sigma_b \in \{0,1,2\} \land \sigma_c = 1 \land a_c \neq 0 \land m = am \land n = nac \rhd \delta\text{‹}\mathbf{0} +$

B9,C12:

$(23) F(0)\text{‹}(t_c + \frac{1}{a_c}(v_{max} - s_c))$

$CBS(\sigma_b, 1, t_c + \frac{1}{a_c}(v_{max} - s_c), t_c + \frac{1}{a_c}(v_{max} - s_c), d,$

$(t_c + \frac{1}{a_c}(v_{max} - s_c) - t_b)(\frac{1}{2}a_b(t_c + \frac{1}{a_c}(v_{max} - s_c) - t_b) + s_b) + x, a_b(t_c + \frac{1}{a_c}(v_{max} - s_c) - t_b) + s_b,$

$(t_c + \frac{1}{a_c}(v_{max} - s_c) - t_b)(\frac{1}{2}a_b(t_c + \frac{1}{a_c}(v_{max} - s_c) - t_b) + s_b) + y, v_{max}, 0, 0, \sigma_d, m, ac)$

$\lhd \sigma_b \in \{0,1,2\} \land \sigma_c = 1 \land a_c \neq 0 \land n = ac \rhd \delta\text{‹}\mathbf{0} +$

B9,C14:

$(24) F(0)\text{‹}(t_c + \frac{1}{a_c}(v_{max} - s_c))$

$CBS(\sigma_b, 2, t_c + \frac{1}{a_c}(v_{max} - s_c), t_c + \frac{1}{a_c}(v_{max} - s_c), d,$

$(t_c + \frac{1}{a_c}(v_{max} - s_c) - t_b)(\frac{1}{2}a_b(t_c + \frac{1}{a_c}(v_{max} - s_c) - t_b) + s_b) + x, a_b(t_c + \frac{1}{a_c}(v_{max} - s_c) - t_b) + s_b,$

$(t_c + \frac{1}{a_c}(v_{max} - s_c) - t_b)(\frac{1}{2}a_b(t_c + \frac{1}{a_c}(v_{max} - s_c) - t_b) + s_b) + y, v_{max}, 0, 0, \sigma_d, m, ac)$

$\triangleleft \sigma_b \in \{0,1,2\} \wedge \sigma_c = 2 \wedge a_c \neq 0 \triangleright \delta \text{‘} 0 +$

Before continuing with calculating the timed deadlock terms (see section 3.3.5) that follow from the application of the expansion and encapsulation theorem, it turns out that it is more efficient to analyse the above 24 terms that are the result of a communication. Since no actions in *Belt* and *Cont* can occur autonomously, the 24 terms resulting from a communication are the only terms that can cause transitions from one state to another. Hence, all the conditions for verifying invariants of the resulting system are present.

The reason for verifying (and finding) invariants may be obvious: using these invariants, the number of possible timed deadlock terms may be reduced because the conditions under which they occur can be falsified using the invariants, meaning that they will not occur at all. Furthermore, conditions can be simplified using the invariants, which eases calculation (and notation).

Careful analysis of the 24 terms resulting from a communication yields eight (easily verified) invariants

The first invariant is used to state the synchronisation between the controller and the belt with respect to the number of trays present on the belt:

$$I_1 \equiv \sigma_b = \sigma_c \ (= \sigma) \tag{3.6}$$

The second invariant, invariant I_2 expresses that the acceleration experienced by the belt equals the acceleration set by the controller:

$$I_2 \equiv a_b = a_c \ (= a) \tag{3.7}$$

The third invariant states that the controller and the belt are synchronised with respect to the time. If this invariant would not hold, the system would be running "out of sync".

$$I_3 \equiv t_b = t_c \ (= t) \tag{3.8}$$

Invariant I_4 expresses that the speed experienced by the trays on the belt equals the velocity the controller expects the belt to have.

$$I_4 \equiv s_b = s_c \ (= s) \tag{3.9}$$

Invariants $I_5 \ldots I_7$ are used to state that in these states the belt is either accelerating to reach maximal speed (v_{max}), or the belt has already reached this maximal speed and is no longer accelerating (or decelerating).

$$I_5 \equiv \sigma = 0 \wedge \sigma_d = del \Rightarrow a > 0 \vee (a = 0 \wedge s > 0) \tag{3.10}$$

$$I_6 \equiv \sigma = 1 \wedge (m = am \vee n = ac) \Rightarrow a > 0 \vee (a = 0 \wedge s > 0) \tag{3.11}$$

$$I_7 \equiv \sigma = 2 \Rightarrow a > 0 \vee (a = 0 \wedge s > 0) \tag{3.12}$$

Invariant I_8 states that the speed as registered by the conveyor belt system will not exceed the maximal speed, nor will it drop below the 0 velocity, i.e. it will not run backwards. Note that the actual speed of the belt may very well exceed the maximal speed if a deadlock occurs, however, since this speed will not be registered by the system, this means that invariant I_8 will still hold.

$$I_8 \ \equiv 0 \leq s \leq v_{max} \tag{3.13}$$

The last invariant, invariant I_9 expresses that when there are two trays on the belt, their distance is at least $\frac{1}{2}l$. Again, this holds as long as no deadlock occurs.

$$I_9 \ \equiv \sigma = 2 \Rightarrow \frac{1}{2}l \leq x - y \tag{3.14}$$

With respect to the proofs of the invariants: the proof of invariants I_1, I_2, I_3, I_4, I_8 and I_9 are trivial. The proof of I_5, I_6 and I_7 has to be done simultaneously and is trivial as well.

Additional advantage is the reduction of the number of terms resulting from communication. The terms $3, 7, 9, 10, 11, 13$ and 15 are all equal to δ‘$\mathbf{0}$ since their conditions are all falsified by invariant I_1. The total number of communicating terms thus equals 17.

3.3.5 Calculating the timed deadlock terms of process CBS'

This section will continue with a further application of the expansion and encapsulation theorem, now strengthened using invariants $I_1 ... I_9$. The following notational convention is used: Bx,Cy denotes the two independent terms of processes *Belt* and *Cont*, and the ($\subseteq n$) denotes the communication terms which "absorb" the timed deadlock terms (see also section 4.1).

Note that for the fact that for $\sigma = 0$ there are two terms, viz. the terms B1 and B9, which can do an action at the same time under the same conditions. This means that these actions yield the same timed deadlock terms after an application of the encapsulation theorem, which then can be joined using the idempotency of alternative choice. Instead of first calculating these timed deadlock terms, they are immediately joined. The timed deadlock terms for $\sigma = 0$ now are:

B1,B9,C1:
$(\delta 1a)\Sigma_{u,w:\mathbf{Time}}\delta$‘$u$ ($\subseteq 1$)
$\quad \lhd \sigma = 0 \wedge u \leq w \wedge 0 \leq a(w-t) + s_c \leq v_{max} \wedge \sigma_d = nod \rhd \delta$‘$\mathbf{0}+$

$(\delta 1b)\Sigma_{u,w:\mathbf{Time}}\delta$‘$w$ ($\subseteq 1$)
$\quad \lhd \sigma = 0 \wedge w \leq u \wedge 0 \leq a(w-t) + s_c \leq v_{max} \wedge \sigma_d = nod \rhd \delta$‘$\mathbf{0}+$

B1,B9,C2:
$(\delta 2a)\Sigma_{u:\mathbf{Time}}\delta$‘$(t - \frac{s_c}{a})$ ($\subseteq 17$)
$\quad \lhd \sigma = 0 \wedge t - \frac{s_c}{a} \leq u \wedge a \neq 0 \wedge \sigma_d = nod \rhd \delta$‘$\mathbf{0}+$

$(\delta 2b)\Sigma_{u:\mathbf{Time}}\delta$‘$u$ ($\subseteq 17$)
$\quad \lhd \sigma = 0 \wedge u \leq t - \frac{s_c}{a} \wedge a \neq 0 \wedge \sigma_d = nod \rhd \delta$‘$\mathbf{0}+$

B1,B9,C3:
$(\delta 3a)\Sigma_{u,w:\mathbf{Time}}\delta$‘$u$ ($\subseteq 2$)
$\quad \lhd \sigma = 0 \wedge u \leq w \wedge 0 \leq a(w-t) + s_c \leq v_{max} \wedge w \leq t + d \wedge \sigma_d = del \rhd \delta$‘$\mathbf{0}+$

$(\delta 3b)\Sigma_{u,w:\mathbf{Time}}\delta$‘$w$ ($\subseteq 2$)
$\quad \lhd \sigma = 0 \wedge w \leq u \wedge 0 \leq a(w-t) + s_c \leq v_{max} \wedge w \leq t + d \wedge \sigma_d = del \rhd \delta$‘$\mathbf{0}+$

B1,B9,C4:
$(\delta 4a)\Sigma_{u:\mathbf{Time}}\delta$‘$u$ ($\subseteq 18$)

$$\lhd \sigma = 0 \land u \leq t + \tfrac{1}{a}(v_{max} - s_c) \land a \neq 0 \land \tfrac{1}{a}(v_{max} - s_c) < d \land \sigma_d = del \rhd \delta\text{‘}\mathbf{0}+$$

$(\delta 4b)\Sigma_{u:\mathbf{Time}}\delta\text{‘}(t + \tfrac{1}{a}(v_{max} - s_c))$ $(\subseteq 18)$
$$\lhd \sigma = 0 \land t + \tfrac{1}{a}(v_{max} - s_c) \leq u \land a \neq 0 \land \tfrac{1}{a}(v_{max} - s_c) < d \land \sigma_d = del \rhd \delta\text{‘}\mathbf{0}+$$

B1,B9,C5:
$(\delta 5a)\Sigma_{u:\mathbf{Time}}\delta\text{‘}u$ $(\subseteq 19)$
$$\lhd \sigma = 0 \land u \leq t + d \land 0 < ad + s_c \leq v_{max} \land \sigma_d = del \rhd \delta\text{‘}\mathbf{0}+$$

$(\delta 5b)\Sigma_{u:\mathbf{Time}}\delta\text{‘}(t + d)$ $(\subseteq 19)$
$$\lhd \sigma = 0 \land t + d \leq u \land 0 < ad + s_c \leq v_{max} \land \sigma_d = del \rhd \delta\text{‘}\mathbf{0}+$$

For $\sigma = 1$, a similar observation as for $\sigma = 0$ is made. This means that instead of calculating the timed deadlock terms for the terms B6 and B9, and for C10 and C11 independently, the calculation is done in one go, resulting in the following timed deadlock terms:

B2,C6:
$(\delta 6a)\delta\text{‘}(t + \tfrac{1}{a}(\sqrt{s^2 + a(l - 2x)} - s))$ $(\subseteq 20)$
$$\lhd \sigma = 1 \land t + \tfrac{1}{a}(\sqrt{s^2 + a(l - 2x)} - s) \leq t \land m = ds \land n = nac \land s = v_{max} \land a \neq 0 \rhd \delta\text{‘}\mathbf{0}+$$

$(\delta 6b)\delta\text{‘}t$ $(\subseteq 20)$
$$\lhd \sigma = 1 \land t \leq t + \tfrac{1}{a}(\sqrt{s^2 + a(l - 2x)} - s) \land m = ds \land n = nac \land s = v_{max} \land a \neq 0 \rhd \delta\text{‘}\mathbf{0}+$$

B2,C7:
$(\delta 7a)\delta\text{‘}(t + \tfrac{1}{a}(\sqrt{s^2 + a(l - 2x)} - s))$ $(\subseteq 21)$
$$\lhd \sigma = 1 \land t + \tfrac{1}{a}(\sqrt{s^2 + a(l - 2x)} - s) \leq t \land m = ds \land n = nac \land s < v_{max} \land a \neq 0 \rhd \delta\text{‘}\mathbf{0}+$$

$(\delta 7b)\delta\text{‘}t$ $(\subseteq 21)$
$$\lhd \sigma = 1 \land t \leq t + \tfrac{1}{a}(\sqrt{s^2 + a(l - 2x)} - s) \land m = ds \land n = nac \land s < v_{max} \land a \neq 0 \rhd \delta\text{‘}\mathbf{0}+$$

B2,C8:
$(\delta 8a)\Sigma_{w:\mathbf{Time}}\delta\text{‘}(t + \tfrac{1}{a}(\sqrt{s^2 + a(l - 2x)} - s))$ $(\subseteq 22)$
$$\lhd \sigma = 1 \land t + \tfrac{1}{a}(\sqrt{s^2 + a(l - 2x)} - s) \leq u \land m = am \land n = nac \land$$
$$0 \leq a(w - t) + s \leq v_{max} \land a \neq 0 \rhd \delta\text{‘}\mathbf{0}+$$

$(\delta 8b)\Sigma_{w:\mathbf{Time}}\delta\text{‘}w$ $(\subseteq 22)$
$$\lhd \sigma = 1 \land w \leq t + \tfrac{1}{a}(\sqrt{s^2 + a(l - 2x)} - s) \land m = am \land n = nac \land$$
$$0 \leq a(w - t) + s \leq v_{max} \land a \neq 0 \rhd \delta\text{‘}\mathbf{0}+$$

B2,C9:
$(\delta 9a)\delta\text{‘}(t + \tfrac{1}{a}(\sqrt{s^2 + a(l - 2x)} - s))$ $(\subseteq 22)$
$$\lhd \sigma = 1 \land \sqrt{s^2 + a(l - 2x)} - s \leq v_{max} - s \land m = am \land n = nac \land a \neq 0 \rhd \delta\text{‘}\mathbf{0}+$$

$(\delta 9b)\delta\text{‘}(t + \tfrac{1}{a}(v_{max} - s))$ $(\subseteq 22)$
$$\lhd \sigma = 1 \land v_{max} - s \leq \sqrt{s^2 + a(l - 2x)} - s \land m = am \land n = nac \land a \neq 0 \rhd \delta\text{‘}\mathbf{0}+$$

B2,C10,C11:
$(\delta 10a)\Sigma_{w:\mathbf{Time}}\delta\text{‘}(t + \tfrac{1}{a}(\sqrt{s^2 + a(l - 2x)} - s))$ $(\subseteq 12 + 22)$
$$\lhd \sigma = 1 \land t + \tfrac{1}{a}(\sqrt{s^2 + a(l - 2x)} - s) \leq w \land n = ac \land 0 \leq a(w - t) + s \leq v_{max} \land a \neq 0 \rhd \delta\text{‘}\mathbf{0}+$$

$(\delta 10b)\Sigma_{w:\mathbf{Time}}\delta\text{‘}w$ $(\subseteq 12)$
$$\lhd \sigma = 1 \land w \leq t + \tfrac{1}{a}(\sqrt{s^2 + a(l - 2x)} - s) \land n = ac \land 0 \leq a(w - t) + s \leq v_{max} \land a \neq 0 \rhd \delta\text{‘}\mathbf{0}+$$

B2,C12:

$(\delta 11a)\delta\text{‘}(t+\frac{1}{a}(\sqrt{s^2+a(l-2x)}-s))$ $(\subseteq 6)$

$\quad \lhd \sigma = 1 \wedge \sqrt{s+a(l-2x)} - s \leq v_{max} - s \wedge n = ac \wedge a \neq 0 \rhd \delta\text{‘}\mathbf{0}+$

$(\delta 11b)\delta\text{‘}(t+\frac{1}{a}(v_{max}-s))$ $(\subseteq 23)$

$\quad \lhd \sigma = 1 \wedge v_{max} - s \leq \sqrt{s^2+a(l-2x)} - s \wedge n = ac \wedge a \neq 0 \rhd \delta\text{‘}\mathbf{0}+$

B3,C6:

$(\delta 12a)\delta\text{‘}(t+\frac{1}{2s}(l-2x))$ $(\subseteq 20)$

$\quad \lhd \sigma = 1 \wedge \frac{1}{2s}(l-2x) \leq 0 \wedge a = 0 \wedge s \neq 0 \wedge m = ds \wedge n = nac \wedge s = v_{max} \rhd \delta\text{‘}\mathbf{0}+$

$(\delta 12b)\delta\text{‘}t$ $(\subseteq 20)$

$\quad \lhd \sigma = 1 \wedge 0 \leq \frac{1}{2s}(l-2x) \wedge a = 0 \wedge s \neq 0 \wedge m = ds \wedge n = nac \wedge s = v_{max} \rhd \delta\text{‘}\mathbf{0}+$

B3,C7:

$(\delta 13a)\delta\text{‘}(t+\frac{1}{2s}(l-2x))$ $(\subseteq 21)$

$\quad \lhd \sigma = 1 \wedge \frac{1}{2s}(l-2x) \leq 0 \wedge a = 0 \wedge s \neq 0 \wedge m = ds \wedge n = nac \wedge s < v_{max} \rhd \delta\text{‘}\mathbf{0}+$

$(\delta 13b)\delta\text{‘}t$ $(\subseteq 21)$

$\quad \lhd \sigma = 1 \wedge 0 \leq \frac{1}{2s}(l-2x) \wedge a = 0 \wedge s \neq 0 \wedge m = ds \wedge n = nac \wedge s < v_{max} \rhd \delta\text{‘}\mathbf{0}+$

B3,C8:

$(\delta 14a)\Sigma_{w:\mathbf{Time}}\delta\text{‘}(t+\frac{1}{2s}(l-2x))$ $(\subseteq 22)$

$\quad \lhd \sigma = 1 \wedge t+\frac{1}{2s}(l-2x) \leq w \wedge a = 0 \wedge s \neq 0 \wedge m = am \wedge n = nac \wedge 0 \leq a(w-t)+s \leq v_{max} \rhd \delta\text{‘}\mathbf{0}+$

$(\delta 14b)\Sigma_{w:\mathbf{Time}}\delta\text{‘}w$ $(\subseteq 22)$

$\quad \lhd \sigma = 1 \wedge w \leq t+\frac{1}{2s}(l-2x) \wedge a = 0 \wedge s \neq 0 \wedge m = am \wedge n = nac \wedge 0 \leq a(w-t)+s \leq v_{max} \rhd \delta\text{‘}\mathbf{0}+$

B3,C10,C11:

$(\delta 15a)\Sigma_{w:\mathbf{Time}}\delta\text{‘}(t+\frac{1}{2s}(l-2x))$ $(\subseteq 8)$

$\quad \lhd \sigma = 1 \wedge t+\frac{1}{2s}(l-2x) \leq w \wedge a = 0 \wedge s \neq 0 \wedge n = ac \wedge 0 \leq a(w-t)+s \leq v_{max} \rhd \delta\text{‘}\mathbf{0}+$

$(\delta 15b)\Sigma_{w:\mathbf{Time}}\delta\text{‘}w$ $(\subseteq 8)$

$\quad \lhd \sigma = 1 \wedge w \leq t+\frac{1}{2s}(l-2x) \wedge a = 0 \wedge s \neq 0 \wedge n = ac \wedge 0 \leq a(w-t)+s \leq v_{max} \rhd \delta\text{‘}\mathbf{0}+$

B3,C12:

$(\delta 16a)\delta\text{‘}(t+\frac{1}{2s}(l-2x))$ $(\subseteq 23)$

$\quad \lhd \sigma = 1 \wedge \frac{1}{2s}(l-2x) \leq \frac{1}{a}(v_{max}-s) \wedge n = ac \wedge a \neq 0 \rhd \delta\text{‘}\mathbf{0}+$

$(\delta 16b)\delta\text{‘}(t+\frac{1}{a}(v_{max}-s))$ $(\subseteq 23)$

$\quad \lhd \sigma = 1 \wedge \frac{1}{a}(v_{max}-s) \leq \frac{1}{2s}(l-2x) \wedge n = ac \wedge a \neq 0 \rhd \delta\text{‘}\mathbf{0}+$

B4,C6:

$(\delta 17a)\delta\text{‘}(t+\frac{1}{a}(\sqrt{s^2+2a(l-x)}-s))$ $(\subseteq 20)$

$\quad \lhd \sigma = 1 \wedge t+\frac{1}{a}(\sqrt{s^2+2a(l-x)}-s) \leq t \wedge m = ds \wedge n = nac \wedge s = v_{max} \wedge a \neq 0 \rhd \delta\text{‘}\mathbf{0}+$

$(\delta 17b)\delta\text{‘}t$ $(\subseteq 20)$

$\quad \lhd \sigma = 1 \wedge t \leq t+\frac{1}{a}(\sqrt{s^2+2a(l-x)}-s) \wedge m = ds \wedge n = nac \wedge s = v_{max} \wedge a \neq 0 \rhd \delta\text{‘}\mathbf{0}+$

B4,C7:

$(\delta 18a)\delta\text{‘}(t+\frac{1}{a}(\sqrt{s^2+2a(l-x)}-s))$ $(\subseteq 21)$

$\quad \lhd \sigma = 1 \wedge t+\frac{1}{a}(\sqrt{s^2+2a(l-x)}-s) \leq t \wedge m = ds \wedge n = nac \wedge s < v_{max} \wedge a \neq 0 \rhd \delta\text{‘}\mathbf{0}+$

$(\delta 18b)\delta\text{‘}t\ (\subseteq 21)$
$\quad\lhd \sigma = 1 \land t \leq t + \frac{1}{a}(\sqrt{s^2 + 2a(l-x)} - s) \land m = ds \land n = nac \land s < v_{max} \land a \neq 0 \rhd \delta\text{‘}\mathbf{0}+$

B4,C8:
$(\delta 19a)\Sigma_{w:\mathbf{Time}}\delta\text{‘}(t + \frac{1}{a}(\sqrt{s^2 + 2a(l-x)} - s))\ (\subseteq 22)$
$\quad\lhd \sigma = 1 \land t + \frac{1}{a}(\sqrt{s^2 + 2a(l-x)} - s) \leq w \land m = am \land n = nac \land$
$\quad 0 \leq a(w-t) + s \leq v_{max} \land a \neq 0 \rhd \delta\text{‘}\mathbf{0}+$

$(\delta 19b)\Sigma_{w:\mathbf{Time}}\delta\text{‘}w\ (\subseteq 22)$
$\quad\lhd \sigma = 1 \land w \leq t + \frac{1}{a}(\sqrt{s^2 + 2a(l-x)} - s) \land m = am \land n = nac \land$
$\quad 0 \leq a(w-t) + s \leq v_{max} \land a \neq 0 \rhd \delta\text{‘}\mathbf{0}+$

B4,C9:
$(\delta 20a)\delta\text{‘}(t + \frac{1}{a}(\sqrt{s^2 + 2a(l-x)} - s))\ (\subseteq 22)$
$\quad\lhd \sigma = 1 \land \sqrt{s^2 + 2a(l-x)} - s \leq v_{max} - s \land m = am \land n = nac \land a \neq 0 \rhd \delta\text{‘}\mathbf{0}+$

$(\delta 20b)\delta\text{‘}(t + \frac{1}{a}(v_{max} - s))\ (\subseteq 22)$
$\quad\lhd \sigma = 1 \land v_{max} - s \leq \sqrt{s^2 + 2a(l-x)} - s \land m = am \land n = nac \land a \neq 0 \rhd \delta\text{‘}\mathbf{0}+$

B4,C10,C11:
$(\delta 21a)\Sigma_{w:\mathbf{Time}}\delta\text{‘}(t + \frac{1}{a}(\sqrt{s^2 + 2a(l-x)} - s))\ (\subseteq 23)$
$\quad\lhd \sigma = 1 \land t + \frac{1}{a}(\sqrt{s^2 + 2a(l-x)} - s) \leq w \land n = ac \land 0 \leq a(w-t) + s \leq v_{max} \land a \neq 0 \rhd \delta\text{‘}\mathbf{0}+$

$(\delta 21b)\Sigma_{w:\mathbf{Time}}\delta\text{‘}w\ (\subseteq 23)$
$\quad\lhd \sigma = 1 \land w \leq t + \frac{1}{a}(\sqrt{s^2 + 2a(l-x)} - s) \land n = ac \land 0 \leq a(w-t) + s \leq v_{max} \land a \neq 0 \rhd \delta\text{‘}\mathbf{0}+$

B4,C12:
$(\delta 22a)\delta\text{‘}(t + \frac{1}{a}(\sqrt{s^2 + 2a(l-x)} - s))\ (\subseteq 20)$
$\quad\lhd \sigma = 1 \land \sqrt{s^2 + 2a(l-x)} - s \leq v_{max} - s \land n = ac \land a \neq 0 \rhd \delta\text{‘}\mathbf{0}+$

$(\delta 22b)\delta\text{‘}(t + \frac{1}{a}(v_{max} - s))\ (\subseteq 20)$
$\quad\lhd \sigma = 1 \land v_{max} - s \leq \sqrt{s^2 + 2a(l-x)} - s \land n = ac \land a \neq 0 \rhd \delta\text{‘}\mathbf{0}+$

B5,C6:
$(\delta 23a)\delta\text{‘}(t + \frac{1}{s}(l-x))\ (\subseteq 21)$
$\quad\lhd \sigma = 1 \land \frac{1}{s}(l-x) \leq 0 \land a = 0 \land s \neq 0 \land m = ds \land n = nac \land s = v_{max} \rhd \delta\text{‘}\mathbf{0}+$

$(\delta 23b)\delta\text{‘}t\ (\subseteq 21)$
$\quad\lhd \sigma = 1 \land 0 \leq \frac{1}{s}(l-x) \land a = 0 \land s \neq 0 \land m = ds \land n = nac \land s = v_{max} \rhd \delta\text{‘}\mathbf{0}+$

B5,C7:
$(\delta 24a)\delta\text{‘}(t + \frac{1}{s}(l-x))\ (\subseteq 8)$
$\quad\lhd \sigma = 1 \land \frac{1}{s}(l-x) \leq 0 \land a = 0 \land s \neq 0 \land m = ds \land n = nac \land s < v_{max} \rhd \delta\text{‘}\mathbf{0}+$

$(\delta 24b)\delta\text{‘}t\ (\subseteq 8)$
$\quad\lhd \sigma = 1 \land 0 \leq \frac{1}{s}(l-x) \land a = 0 \land s \neq 0 \land m = ds \land n = nac \land s < v_{max} \rhd \delta\text{‘}\mathbf{0}+$

B5,C9:
$(\delta 25a)\delta\text{‘}(t + \frac{1}{s}(l-x))\ (\subseteq 23)$
$\quad\lhd \sigma = 1 \land \frac{1}{s}(l-x) \leq \frac{1}{a}(v_{max} - s) \land a = 0 \land s \neq 0 \land m = am \land n = nac \land a \neq 0 \rhd \delta\text{‘}\mathbf{0}+$

$(\delta 25b)\delta\text{‘}(t + \frac{1}{a}(v_{max} - s))\ (\subseteq 23)$

$\lhd \sigma = 1 \wedge \frac{1}{a}(v_{max} - s) \leq \frac{1}{s}(l - x) \wedge a = 0 \wedge s \neq 0 \wedge m = am \wedge n = nac \wedge a \neq 0 \rhd \delta\text{‘}\mathbf{0}+$

B5,C10,C11:

$(\delta 26a)\Sigma_{w:\mathbf{Time}}\delta\text{‘}(t + \frac{1}{s}(l - x))$ $(\subseteq 23)$

$\lhd \sigma = 1 \wedge t + \frac{1}{s}(l - x) \leq w \wedge a = 0 \wedge s \neq 0 \wedge n = ac \wedge 0 \leq a(w - t) + s \leq v_{max} \rhd \delta\text{‘}\mathbf{0}+$

$(\delta 26b)\Sigma_{w:\mathbf{Time}}\delta\text{‘}w$ $(\subseteq 23)$

$\lhd \sigma = 1 \wedge w \leq t + \frac{1}{s}(l - x) \wedge a = 0 \wedge s \neq 0 \wedge n = ac \wedge 0 \leq a(w - t) + s \leq v_{max} \rhd \delta\text{‘}\mathbf{0}+$

B5,C12:

$(\delta 27a)\delta\text{‘}(t + \frac{1}{s}(l - x))$ $(\subseteq 23)$

$\lhd \sigma = 1 \wedge \frac{1}{s}(l - x) \leq \frac{1}{a}(v_{max} - s) \wedge n = ac \wedge a \neq 0 \rhd \delta\text{‘}\mathbf{0}+$

$(\delta 27b)\delta\text{‘}(t + \frac{1}{a}(v_{max} - s))$ $(\subseteq 23)$

$\lhd \sigma = 1 \wedge \frac{1}{a}(v_{max} - s) \leq \frac{1}{s}(l - x) \wedge n = ac \wedge a \neq 0 \rhd \delta\text{‘}\mathbf{0}+$

B6,B9,C6:

$(\delta 28a)\Sigma_{u:\mathbf{Time}}\delta\text{‘}u$ $(\subseteq 20)$

$\lhd \sigma = 1 \wedge u \leq t \wedge m = ds \wedge n = nac \wedge s = v_{max} \rhd \delta\text{‘}\mathbf{0}+$

$(\delta 28b)\Sigma_{u:\mathbf{Time}}\delta\text{‘}t$ $(\subseteq 20)$

$\lhd \sigma = 1 \wedge t \leq u \wedge m = ds \wedge n = nac \wedge s = v_{max} \rhd \delta\text{‘}\mathbf{0}+$

B6,B9,C7:

$(\delta 29a)\Sigma_{u:\mathbf{Time}}\delta\text{‘}u$ $(\subseteq 21)$

$\lhd \sigma = 1 \wedge u \leq t \wedge m = ds \wedge n = nac \wedge s < v_{max} \rhd \delta\text{‘}\mathbf{0}+$

$(\delta 29b)\Sigma_{u:\mathbf{Time}}\delta\text{‘}t$ $(\subseteq 21)$

$\lhd \sigma = 1 \wedge t \leq u \wedge m = ds \wedge n = nac \wedge s < v_{max} \rhd \delta\text{‘}\mathbf{0}+$

B6,B9,C8:

$(\delta 30a)\Sigma_{u,w:\mathbf{Time}}\delta\text{‘}u$ $(\subseteq 22)$

$\lhd \sigma = 1 \wedge u \leq w \wedge m = am \wedge n = nac \wedge 0 \leq a(w - t) + s \leq v_{max} \rhd \delta\text{‘}\mathbf{0}+$

$(\delta 30b)\Sigma_{u,w:\mathbf{Time}}\delta\text{‘}w$ $(\subseteq 22)$

$\lhd \sigma = 1 \wedge w \leq u \wedge m = am \wedge n = nac \wedge 0 \leq a(w - t) + s \leq v_{max} \rhd \delta\text{‘}\mathbf{0}+$

B6,B9,C9:

$(\delta 31a)\Sigma_{u:\mathbf{Time}}\delta\text{‘}u$ $(\subseteq 22)$

$\lhd \sigma = 1 \wedge u \leq t + \frac{1}{a}(v_{max} - s) \wedge m = am \wedge n = nac \wedge a \neq 0 \rhd \delta\text{‘}\mathbf{0}+$

$(\delta 31b)\Sigma_{u:\mathbf{Time}}\delta\text{‘}(t + \frac{1}{a}(v_{max} - s))$ $(\subseteq 22)$

$\lhd \sigma = 1 \wedge t + \frac{1}{a}(v_{max} - s) \leq u \wedge m = am \wedge n = nac \wedge a \neq 0 \rhd \delta\text{‘}\mathbf{0}+$

B6,B9,C10,C11:

$(\delta 32a)\Sigma_{u,w:\mathbf{Time}}\delta\text{‘}u$ $(\subseteq 12)$

$\lhd \sigma = 1 \wedge u \leq w \wedge n = ac \wedge 0 \leq a(w - t) + s \leq v_{max} \rhd \delta\text{‘}\mathbf{0}+$

$(\delta 32b)\Sigma_{u,w:\mathbf{Time}}\delta\text{‘}w$ $(\subseteq 12)$

$\lhd \sigma = 1 \wedge w \leq u \wedge n = ac \wedge 0 \leq a(w - t) + s \leq v_{max} \rhd \delta\text{‘}\mathbf{0}+$

B6,B9,C12:

$(\delta 33a)\Sigma_{u:\mathbf{Time}}\delta\text{‘}u$ $(\subseteq 23)$

$\triangleleft \sigma = 1 \wedge u \leq t + \frac{1}{a}(v_{max} - s) \wedge n = ac \wedge a \neq 0 \triangleright \delta\text{‘}\mathbf{0}+$

$(\delta 33b)\Sigma_{u:\mathbf{Time}}\delta\text{‘}(t + \frac{1}{a}(v_{max} - s))\ (\subseteq 23)$
$\quad \triangleleft \sigma = 1 \wedge t + \frac{1}{a}(v_{max} - s) \leq u \wedge n = ac \wedge a \neq 0 \triangleright \delta\text{‘}\mathbf{0}+$

For $\sigma = 2$ no further efficiency in the calculation can be introduced, except for the elimination of the timed deadlock terms resulting from the combination of B8 and C14, since the condition under which this timed deadlock terms can take place is falsified under the assumption of invariant I_2. The timed deadlock terms that are not eliminated by the invariants are:

B7,C13:
$(\delta 34a)\Sigma_{w:\mathbf{Time}}\delta\text{‘}(t + \frac{1}{a}(\sqrt{s^2 + 2a(l - x)} - s)\ (\subseteq 24)$
$\quad \triangleleft \sigma = 2 \wedge t + \frac{1}{a}(\sqrt{s^2 + 2a(l - x)} - s \leq w \wedge a \neq 0 \wedge 0 \leq a(w - t) + s \leq v_{max} \triangleright \delta\text{‘}\mathbf{0}+$

$(\delta 34b)\Sigma_{w:\mathbf{Time}}\delta\text{‘}w\ (\subseteq 24)$
$\quad \triangleleft \sigma = 2 \wedge w \leq t + \frac{1}{a}(\sqrt{s^2 + 2a(l - x)} - s \wedge a \neq 0 \wedge 0 \leq a(w - t) + s \leq v_{max} \triangleright \delta\text{‘}\mathbf{0}+$

B7,C14:
$(\delta 35a)\delta\text{‘}(t + \frac{1}{a}(\sqrt{s^2 + 2a(l - x)} - s)\ (\subseteq 24)$
$\quad \triangleleft \sigma = 2 \wedge \sqrt{s^2 + 2a(l - x)} - s \leq v_{max} - s \wedge a \neq 0 \triangleright \delta\text{‘}\mathbf{0}+$

$(\delta 35b)\delta\text{‘}(t + \frac{1}{a}(v_{max} - s))\ (\subseteq 24)$
$\quad \triangleleft \sigma = 2 \wedge v_{max} - s \leq \sqrt{s^2 + 2a(l - x)} - s \wedge a \neq 0 \triangleright \delta\text{‘}\mathbf{0}+$

B8,C13:
$(\delta 36a)\Sigma_{w:\mathbf{Time}}\delta\text{‘}(t + \frac{1}{s}(l - x))\ (\subseteq 16)$
$\quad \triangleleft \sigma = 2 \wedge t + \frac{1}{s}(l - x) \leq w \wedge 0 \leq a(w - t) + s \leq v_{max} \wedge a = 0 \wedge s \neq 0 \triangleright \delta\text{‘}\mathbf{0}+$

$(\delta 36b)\Sigma_{w:\mathbf{Time}}\delta\text{‘}w\ (\subseteq 16)$
$\quad \triangleleft \sigma = 2 \wedge w \leq t + \frac{1}{s}(l - x) \wedge 0 \leq a(w - t) + s \leq v_{max} \wedge a = 0 \wedge s \neq 0 \triangleright \delta\text{‘}\mathbf{0}+$

B9,C13:
$(\delta 37a)\Sigma_{u,w:\mathbf{Time}}\delta\text{‘}u\ (\subseteq 24)$
$\quad \triangleleft \sigma = 2 \wedge u \leq w \wedge 0 \leq a(w - t) + s \leq v_{max} \triangleright \delta\text{‘}\mathbf{0}+$

$(\delta 37b)\Sigma_{u,w:\mathbf{Time}}\delta\text{‘}w\ (\subseteq 24)$
$\quad \triangleleft \sigma = 2 \wedge w \leq u \wedge 0 \leq a(w - t) + s \leq v_{max} \triangleright \delta\text{‘}\mathbf{0}+$

B9,C14:
$(\delta 38a)\Sigma_{u:\mathbf{Time}}\delta\text{‘}u\ (\subseteq 24)$
$\quad \triangleleft \sigma = 2 \wedge u \leq t + \frac{1}{a}(v_{max} - s) \wedge a \neq 0 \triangleright \delta\text{‘}\mathbf{0}+$

$(\delta 38b)\Sigma_{u:\mathbf{Time}}\delta\text{‘}(t + \frac{1}{a}(v_{max} - s))\ (\subseteq 24)$
$\quad \triangleleft \sigma = 2 \wedge t + \frac{1}{a}(v_{max} - s) \leq u \wedge a \neq 0 \triangleright \delta\text{‘}\mathbf{0}$

The total expansion, under the assumption of invariants $I_1 ... I_9$, yields 38 pairs of timed deadlock terms. The invariant I_1 furthermore reduces the numbers of terms containing communications by seven. The total number of terms for the expansion of process CBS' thus equals 93. The number of timed deadlock terms is rather alarming, they make up 82 percent of the total number of terms. A straightforward application of the encapsulation theorem, i.e. without using the invariants would even yield a 91-percent rate, which is rather unacceptable. Clearly, this calls for more investigation into the encapsulation and expansion theorem.

4 Analysis of the Conveyor Belt System

The previous sections focused mainly on the construction and transformation of our conveyor belt system. Now using the descriptions, devised in the previous sections we are now in a position to perform analysis on the conveyor belt system. Basically, there are two criteria that can be analysed, viz:

- proving absence of deadlock of the model,
- analysing the performance of the conveyor belt system.

The first item is addressed in section 4.1. The performance is subsequently discussed in section 4.2.

4.1 Proving absence of deadlock

In order to prove absence of deadlock, it suffices to be able to show that every timed deadlock term, calculated in section 3.3.5, can be absorbed by at least one term resulting from a communication. The proofs thereof, however, are tedious but require little calculation, and are therefore here omitted. Suffices to state that all deadlocks can be removed under the assumption of invariants $I_1 \ldots I_9$. It must be noted that these invariants are also necessary for the proofs, which may seem rather surprising. A few representative proofs can be found in appendix A. The communication terms absorbing a timed deadlock term are annotated in section 3.3.5, next to the timed deadlock term in question.
The process that describes the Conveyor Belt System thus can be written as follows (after delinearisation):

$$\textbf{proc } CBS'^{nd}_0(t : \textbf{Time}, s, a : \mathbb{R}) =$$

$$\Sigma_{u:\textbf{Time}}\ ar\text{‘}u\ CBS'^{nac,ds}_1(u, 0, a(u-t)+s, a) \triangleleft 0 \leq a(u-t)+s \leq v_{max} \triangleright \delta\text{‘}\mathbf{0} +$$

$$F(0)\text{‘}(t - \tfrac{s}{a})\ CBS'^{nd}_0(t - \tfrac{s}{a}, 0, 0) \triangleleft a \neq 0 \triangleright \delta\text{‘}\mathbf{0}$$

$$\textbf{proc } CBS'^{d}_0(t, d : \textbf{Time}, s, a : \mathbb{R}) =$$

$$\Sigma_{u:\textbf{Time}}\ ar\text{‘}u\ CBS'_1 nac, am(u, 0, a(u-t)+s, a) \triangleleft 0 \leq a(u-t)+s \leq v_{max} \wedge u \leq t+d \triangleright \delta\text{‘}\mathbf{0} +$$

$$F(-T_{max})\text{‘}(t+d)\ CBS'^{nd}_0(t+d, ad+s, -b_2T_{max}) \triangleleft 0 < ad+s \leq v_{max} \triangleright \delta\text{‘}\mathbf{0} +$$

$$F(0)\text{‘}(t + \tfrac{1}{a}(v_{max}-s))\ CBS'^{d}_0(t + \tfrac{1}{a}(v_{max}-s), d - \tfrac{1}{a}(v_{max}-s), v_{max}, 0)$$
$$\triangleleft a \neq 0 \wedge \tfrac{1}{a}(v_{max}-s) < d \triangleright \delta\text{‘}\mathbf{0}$$

$$\textbf{proc } CBS'^{nac,ds}_1(t : \textbf{Time}, x, s, a : \mathbb{R}) =$$

$$F(0)\text{‘}t\ CBS'^{nac,am}_1(t, x, s, 0) \triangleleft s = v_{max} \triangleright \delta\text{‘}\mathbf{0} +$$

$$F(T_{max})\text{‘}t\ CBS'^{nac,am}_1(t, x, s, b_2T_{max}) \triangleleft s < v_{max} \triangleright \delta\text{‘}\mathbf{0}$$

$$\textbf{proc } CBS'^{nac,am}_1(t : \textbf{Time}, x, s, a : \mathbb{R}) =$$

$$mid\text{‘}(t + \tfrac{1}{2s}(l-2x))\ CBS'^{ac}_1(t + \tfrac{1}{2s}(l-2x), \tfrac{1}{2}l, s, a) \triangleleft a = 0 \wedge 0 < s \leq v_{max} \triangleright \delta\text{‘}\mathbf{0} +$$

$$mid\text{‘}(t + \tfrac{1}{a}(\sqrt{s^2 + a(l-2x)} - s))\ CBS'^{ac}_1(t + \tfrac{1}{a}(\sqrt{s^2 + a(l-2x)} - s), \tfrac{1}{2}l, \sqrt{s^2 + a(l-2x)}, a)$$
$$\triangleleft a \neq 0 \wedge 0 \leq \sqrt{s^2 + a(l-2x)} \leq v_{max} \triangleright \delta\text{‘}\mathbf{0} +$$

$$F(0)\text{‘}(t + \tfrac{1}{a}(v_{max} - s))\ CBS'^{nac,am}_{1}(t + \tfrac{1}{a}(v_{max} - s), \tfrac{1}{2a}(v_{max}^2 - s^2) + x, v_{max}, 0) \lhd a \neq 0 \rhd \delta\text{‘}\mathbf{0}$$

proc $CBS'^{ac}_{1}(t : \mathbf{Time}, x, s, a : \mathbb{R}) =$

$$\Sigma_{u:\mathbf{Time}}\ ar\text{‘}u\ CBS'_2(u, (u-t)(\tfrac{1}{2}a(u-t)+s)+x, a(u-t)+s, 0, a) \lhd 0 \leq a(u-t)+s \leq v_{max} \rhd \delta\text{‘}\mathbf{0} +$$

$$dep\text{‘}(t + \tfrac{1}{s}(l - x))\ CBS'^{d}_{0}(t + \tfrac{1}{s}(l - x), k, s, a) \lhd a = 0 \wedge 0 < s \leq v_{max} \rhd \delta\text{‘}\mathbf{0} +$$

$$dep\text{‘}(t + \tfrac{1}{a}(\sqrt{s^2 + 2a(l - x)} - s))\ CBS'^{d}_{0}(t + \tfrac{1}{a}(\sqrt{s^2 + 2a(l - x)} - s), k, \sqrt{s^2 + 2a(l - x)}, a)$$
$$\lhd a \neq 0 \wedge 0 \leq \sqrt{s^2 + 2a(l - x)} \leq v_{max} \rhd \delta\text{‘}\mathbf{0} +$$

$$F(0)\text{‘}(t + \tfrac{1}{a}(v_{max} - s))\ CBS'^{nac,am}_{1}(t + \tfrac{1}{a}(v_{max} - s), \tfrac{1}{2a}(v_{max}^2 - s^2) + x, v_{max}, 0) \lhd a \neq 0 \rhd \delta\text{‘}\mathbf{0}$$

proc $CBS'_2(t : \mathbf{Time}, x, s, y, a : \mathbb{R}) =$

$$dep\text{‘}(t + \tfrac{1}{s}(l - x))\ CBS'^{nac,am}_{1}(t + \tfrac{1}{s}(l - x), y + (l - x), s, a) \lhd a = 0 \wedge 0 < s \leq v_{max} \rhd \delta\text{‘}\mathbf{0} +$$

$$dep\text{‘}(t + \tfrac{1}{a}(\sqrt{s^2 + 2a(l - x)} - s))$$
$$CBS'^{nac,am}_{1}(t + \tfrac{1}{a}(\sqrt{s^2 + 2a(l - x)} - s), y + (l - x), \sqrt{s^2 + 2a(l - x)}, a)$$
$$\lhd a \neq 0 \wedge 0 \leq \sqrt{s^2 + 2a(l - x)} \leq v_{max} \rhd \delta\text{‘}\mathbf{0} +$$

$$F(0)\text{‘}(t + \tfrac{1}{a}(v_{max} - s))\ CBS'_2(t + \tfrac{1}{a}(v_{max} - s), \tfrac{1}{2a}(v_{max}^2 - s^2) + x, v_{max}, \tfrac{1}{2a}(v_{max}^2 - s^2) + y, 0)$$
$$\lhd a \neq 0 \rhd \delta\text{‘}\mathbf{0}$$

In section 3.3.4 it is stated that invariant I_8 expresses the fact that during normal control the controller takes care of keeping the velocity of the belt between 0 and v_{max}. This is valid unless a deadlock occurs, but since in this section it is proved that the system is deadlock free, it also means that the second safety requirement is met, and can actually be proved to hold.

Since no deadlock occurs, also the safety control objective that the trays should not collide is proven in the same manner as above, since invariant I_9 remains valid.

4.2 Performance analysis of the conveyor belt system

This section mainly deals with the performance analysis of the conveyor belt system. The items that fall into this category are the following:

- the average throughput time of trays,
- the start-up time of the system,
- energy efficiency of the conveyor belt system.

The throughput efficiency and the start-up time of the system are not too hard to define. However, there are some problems in defining the energy efficiency of the system. This somehow has to reflect a combination of the amount of time the belt is running idle, i.e. the belt is not decelerating while there are no trays on the belt, and the time the belt needs to accelerate in order to deliver a tray to the environment. This notion of performance is very hard to define formally. Since the performance of the system is greatly dependent on the delivery of the trays, i.e. on the environment, it is necessary to choose a specific environment in order to do the performance analysis on the system. Here, only two different environments are modelled, i.e. one with a continuous delivery of trays and one with a delivery of trays on the basis of a constant *inter-arrival* time T. The inter-arrival time is in this case defined by the time between the arrival of two successive trays.

4.2.1 A continuous *asap* delivery of trays

The environment that is most likely to give an upper bound for the performance of the conveyor belt system is the system in which the conveyor belt system is actually the bottle neck. This means the up-stream environment is always able to deliver trays to the belt. A model for this kind of environment is a buffer that is in principle infinite, in which case it is always possible to feed a tray to the conveyor belt system. Using meta-knowledge of the conveyor belt system, in this case the knowledge that a new tray can only arrive when the previous tray already reached the mid point of the belt, a specification in timed μCRL for this kind of environment is as follows:

proc $Env_u(t : \mathbf{Time}) =$

$$\Sigma_{u:\mathbf{Time}} ar_u ‘ t . mid_u ‘ u \ Env_u(u)$$

The communication between the upstream environment and the conveyor belt system is defined as follows:

comm

$$\begin{array}{lcl} ar_e & = & ar_u \mid ar \\ mid_e & = & mid_u \mid mid \end{array}$$

The actual behaviour of the system, under the assumption that the environment is as specified as above is as follows:

proc $Continuous_{asap}(t_u, t_c, d : \mathbf{Time}, \sigma : State, x, s, y, a : \mathbb{R},$
$\qquad \sigma_d : DState, m : Motion, n : Accept) =$
$\quad \partial_{H'}(Env_u(t_u) || CBS'(\sigma, t_c, d, x, s, y, a, \sigma_d, m, n))$

Where the set H' is defined as follows: $H' = \{ar, ar_u, mid_u, mid\}$.
Initially, the environment behaves as follows:

Init $= Continuous_{asap}(0, 0, d, 0, 0, 0, 0, 0, nod, m, n)$

In this system, the throughput of the belt is from one point in time and onwards constant. One interesting factor that can be calculated is the duration of the start-up noise, i.e. the time the system needs to show a stable behaviour. In this case, this equals the time the system needs to accelerate to v_{max}.

- Assuming s = 0 when the first tray arrives, the start-up noise time can be calculated to be $\frac{v_{max}}{b_2 T_{max}}$.

This means, that after this time, the belt is moving with the constant speed of v_{max}. The throughput, i.e. the time between the arrival of a tray and its subsequent departure, can easily be calculated:

- throughput: $t_{min} = \frac{l}{v_{max}}$.

This throughput time is also a lower bound for the throughput time, since no environment utilises the belt in a more efficient manner than this environment. This means that the time t_{min} can be used for comparison purposes.

Since this system does not depend in any way on the variable k (the delay) to determine the efficiency of the belt, the analysis of the system in this context does not yield any interesting results. Although this system is optimal with respect to the start-up noise time and the throughput time, it is also in general not very realistic, since in practice there is hardly ever an infinite buffer containing the trays.

4.2.2 A continuous, *constant inter-arrival time* delivery of trays

A more realistic environment is the environment in which trays arrive at a constant inter-arrival time T, where the inter-arrival time is defined as the time between two successive arrivals of trays. The up-stream environment can in this case be described by a buffer which outputs trays at a constant inter-arrival time of T time-units. In order to avoid introducing a deadlock in the composition of the environment and the conveyor belt system, a lower bound for T is the time it takes for the belt to transport a tray to the mid point when running at zero speed. Process Env'_u represents a timed μCRL model of the up-stream environment as defined in this section.

$$\textbf{proc} \quad Env'_u(t : \textbf{Time}) =$$

$$ar_e{}^{\varsigma}t.Env'_u(t+T) \lhd \sqrt{\tfrac{1}{b_2 T_{max}} 2l} \leq T \rhd \delta{}^{\varsigma}\mathbf{0}$$

The communications between the up-stream environment and the conveyor belt system can be defined as follows:

$$\textbf{comm}$$

$$ar_e \quad = \quad ar_u \mid ar$$

The behaviour of the composition of the up-stream environment and the conveyor belt system is defined as follows:

$$\begin{aligned} &\textbf{proc} \quad Continuous_{constant}(t_u, t_c, d : \textbf{Time}, \sigma : State, x, s, y, a : \mathbb{R}, \\ &\qquad\qquad\qquad\qquad \sigma_d : DState, m : Motion, n : Accept) = \\ &\quad \partial_{H''}(Env'_u(t_u) \mid\mid CBS'(\sigma, t_c, d, x, s, y, a, \sigma_d, m, n)) \end{aligned}$$

Where the encapsulation set H'' is defined as follows: $H'' \;=\; \{ar_u, ar\}$.
Initially, the system behaves as follows:

$$\textbf{Init} = Continuous_{constant}(0, 0, d, 0, 0, 0, 0, 0, nod, m, n)$$

An operational analysis of this model yields the following questions: dependent on the values of a, l and T, can v_{max} ever be reached, and what is the throughput efficiency compared to the most efficient throughput time t_{min}. To solve this problem two cases can be distinguished. On the one hand, there is the case when the belt eventually reaches the maximal speed, and on the other hand, there is the case where the belt never reaches the maximal speed. These two cases are subsequently investigated.

Case 1 Suppose the belt can eventually be accelerated to speed v_{max}. Abstracting from the start-up noise time, figure 4 shows the behaviour of the system when it has finally become stable. For the moment, it is assumed that the system can finally become stable, but whether this can actually be the case, is still an open question.
Suppose T is greater than the time that is necessary for one tray to arrive on the belt and be transported to the end of the belt, i.e. it is always the case there is at most one tray on the belt. If this assumption is violated, the system is comparable with the continuous asap delivery of trays, and only the start-up noise time can be interesting.

If we take the throughput to be at least p percent of the minimal throughput time (so $p = \frac{100 t_{min}}{t}$, where t is the actual throughput time), then it is easy to calculate the conditions under which this percentage is achieved by using the following equations:

$$s = v_{max} - a t_0 \tag{4.15}$$

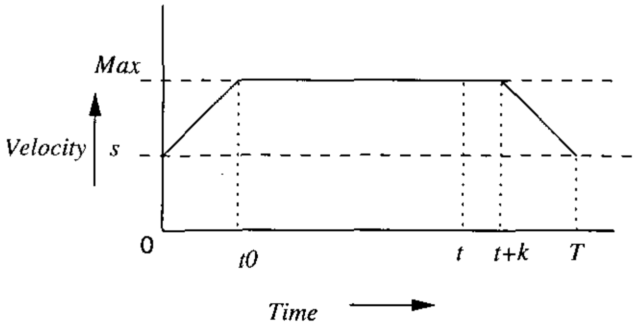

Figure 4: *A velocity-time diagram after stabilising*

Equation 4.15 actually expresses that the maximal speed is obtained after t_0 time units starting from a speed s.

$$s = v_{max} - a(T - (t + k)) \tag{4.16}$$

Equation 4.16 expresses that after delivery of a tray to the environment, a new tray arrives when the belt has the same speed it had when the previous tray arrived. Note that altering the delay k also means altering the conditions on the starting speed s.

Using the fact that $t = \frac{100t_{min}}{p}$ and $t_{min} = \frac{l}{v_{max}}$, the following equation can be derived:

$$s = v_{max} + \frac{100al}{pv_{max}} + a(k - T) \tag{4.17}$$

This means that the initial speed is dependent only of the length of the belt, the maximal speed, the inter-arrival time and the variable k which represents a delay time. Using equation 4.17, it is possible to calculate how great the delay should be in order to obtain a throughput efficiency of p-percent:

$$k = T - \frac{100al}{pv_{max}} - \frac{v_{max} - s}{a} \tag{4.18}$$

Until now we have assumed that the start-up noise time is limited and that a stable system can be achieved. This however, does not have to be the case. Finding out whether the system will stabilise is left as a future research exercise.

Case 2 Now, assume the belt cannot be accelerated to v_{max}, due to the too small delay k and a too great inter-arrival time T. This brings about the problem that the speed for the tray that arrives next is highly dependent on the speed the belt had when the previous tray was delivered. Again two cases can be distinguished. If the inter-arrival time T, in combination with the delay time k, causes the belt to be decelerated to zero speed *before* a new tray arrives, the system will behave stable immediately: the throughput of trays will be constant, starting from the first tray, and the end situation will always be that the belt has come to a standstill. The only possible way to achieve a $p\%$ throughput efficiency is by choosing a specific acceleration, which can be calculated as follows:

$\quad \frac{at^2}{2} \wedge t = \frac{100t_{min}}{p}$
$\equiv$ { rewrite t_{min}}
$\quad a = \frac{2l}{t^2} \wedge t = \frac{100l}{pv_{max}}$
$\Rightarrow$ { substitute t in first equation }

$a = 2\frac{(pv_{max})^2}{100^2 l}$

□

These equations use the variables a and l as they are used in the model described in section 4.1, in which a represents the maximal acceleration (bT_{max}) and l represents the length of the belt. The choice for a can be realistic, depending on the choice for both l and v_{max} and of course p.

The second case is when the inter-arrival time T, in combination with the delay time k, causes the belt to be decelerated to a velocity, greater than zero before a new tray arrives. The question now is whether the p% efficiency *can* be achieved. This situation is sketched in figure 5.

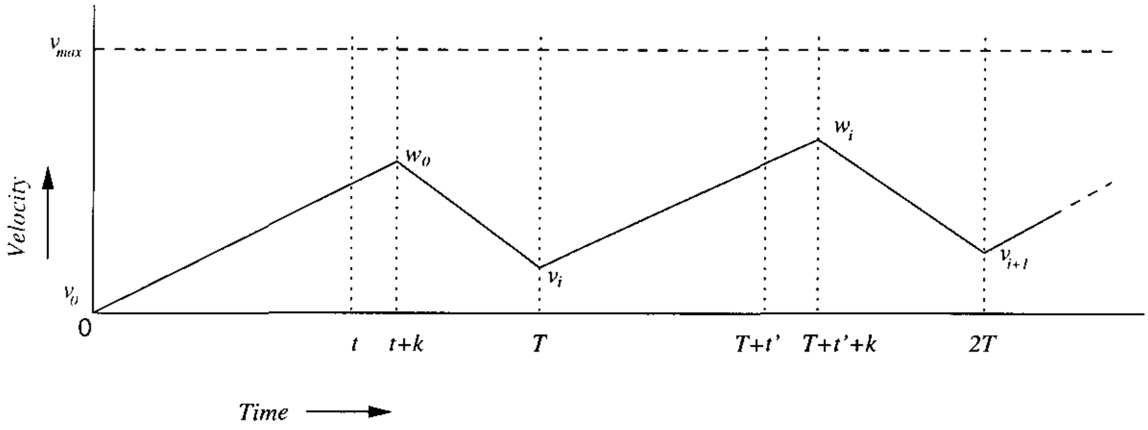

Figure 5: *A velocity-time diagram for Case 2*

Careful analysis and standard mathematics will in this case be the answer to this question. Under the assumption that, given a T and a k, v_{max} is *never* reached, the following equations are known:

$$v_0 = 0 \tag{4.19}$$

This expresses that the belt is initially at a stand-still. Note that this is an assumption, however, this assumption is very straightforward.

$$w_0 = \sqrt{2al} \tag{4.20}$$

The peak velocity after delivering the first tray is represented by w_0. In this equation the variables as used in the model described in section 4.1 are used again, in which l represents the length of the belt and a represents the maximal acceleration again.

$$\frac{w_i - v_i}{t_i + k} = a \tag{4.21}$$

Using our knowledge of the model, we know that the belt is accelerated during $t + k$ time units from speed v_i to speed w_i, with an acceleration of a. This is in essence exactly what is expressed in equation 4.21. The variable t_i is in this case used to express the time that is needed for a tray to arrive at the belt and subsequently be delivered to the environment.

$$at^2 + 2v_i t_i = 2l \tag{4.22}$$

Equation 4.22 expresses the fact that after t_i time units, the tray has travelled a distance of exactly l, when the speed of the belt was v_i when the tray arrived.

$$\frac{w_i - v_{i+1}}{T - (t_i + k)} = a \tag{4.23}$$

Equation 4.23 expresses the fact that the belt, after the peak velocity is reached, is decelerated at the same pace as it is accelerated, until a new tray arrives.

Using basic mathematics, the following equation can be derived from the equations 4.21 and 4.22:

$$t_i = \frac{-v_i + \sqrt{v_i^2 + 2al}}{a} \vee t_i = \frac{-v_i - \sqrt{v_i^2 + 2al}}{a} \tag{4.24}$$

Since t_i is a value that represents a time, it may be clear that only the first solution is appropriate: the second solution is negative unless $a < 0$, but this cannot be the case (see for instance invariant I_7). Note that this equation is similar to the equations found in the model in section 4.1.

Now the equation 4.24 can be put to use for finding recurrence relations between the two peak-velocity v_i and w_i. Substitution of equation 4.24 in equations 4.21 and 4.22 yields the following two relations:

$$w_i = \sqrt{v_i^2 + 2al} + ak \tag{4.25}$$

$$v_{i+1} = 2\sqrt{v_i^2 + 2al} - aT - v_i + 2ak \tag{4.26}$$

Interesting to see is that the peak velocity v_{i+i} can be expressed as a dependency only on the previous peak velocity v_i, and not as one would expect on the peak velocity w_i. This means that now the behaviour of the system after a while can be studied, by calculating what the values of the peak velocities v_i and w_i tend to become when i tends to go to ∞. Under the assumption that this limit actually exists, the following calculation can be made:

$\lim_{i\to\infty} v_i$
$= \{$ substitute $i + 1$ for $i\}$
$\lim_{i\to\infty} v_{i+1}$
$= \{$ definition of $v_{i+1}\}$
$\lim_{i\to\infty}(2\sqrt{v_i^2 + 2al} - aT - v_i + 2ak)$

□

Based on the recurrence in this calculation, the following equality can be derived:

$$(\lim_{i\to\infty} v_i) + \tfrac{1}{2}aT - ak = \lim_{i\to\infty}(\sqrt{v_i^2 + 2al})$$

This equality gives rise to the following equality:

$$((\lim_{i\to\infty} v_i) + (\tfrac{1}{2}aT - ak))^2 = (\lim_{i\to\infty} v_i)^2 + 2al$$

□

Now using standard mathematics, the following equation can be derived:

$$V = \lim_{i\to\infty} v_i = \frac{2al - (aT - 2ak)^2}{4(aT - 2ak)} \tag{4.27}$$

A similar exercise can be done for the peak value w_i for i tending to ∞, using the now known value for $\lim_{i\to\infty} v_i$. This yields the following value:

$$W = \lim_{i\to\infty} w_i = \sqrt{V^2 + 2al} + ak \tag{4.28}$$

Using the equations 4.27 and 4.28, the eventual throughput time can be calculated, using equation 4.21 to express this time in the values V and W as follows:

$$t_\infty = \frac{W - V - ak}{a} \tag{4.29}$$

In terms of throughput efficiency (p), this can be expressed as follows:

$$p = \frac{100al}{v_{max}(W - V - ak)}\% \tag{4.30}$$

In order to visualise the throughput efficiency for a given a, l and v_{max}, the throughput efficiency is plotted as a function which takes k and T as its arguments. Figure 6 shows the throughput efficiency for $a = 0.1\ m/s^2$, $l = 5\ m$ and a maximal speed of $v_{max} = 0.75\ m/s$.

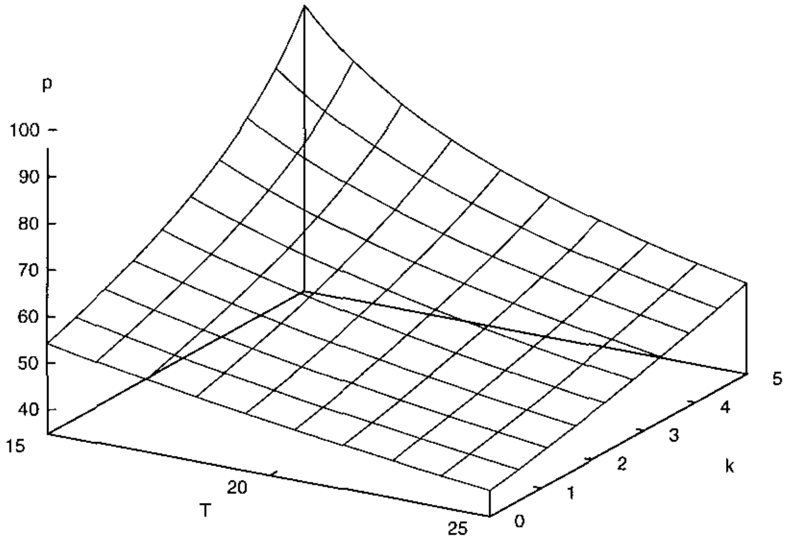

Figure 6: *The throughput efficiency for a constant maximal speed*

If the maximal speed of the belt is variable, e.g. dependent on the inter-arrival time T, the results are clearly different.

A reason for having a dependency on the inter-arrival time, could be that when the belt has a lot of trays that need transportation, this needs to be done as fast as possible. When the inter-arrival time is very short, the maximal speed needs to be higher as to obtain a higher throughput efficiency, as opposed to when the inter-arrival time is very long, in which case it can be more efficient to lower the maximal speed. Using the same constants for a and l, the maximal speed is now defined as follows: $v_{max}(T) = \frac{1}{T}3l\ m/s$. The effects this would have on the throughput efficiency are plotted in figure 7.

4.3 Summary

The analysis in previous sections clearly showed that the greater k, the higher the throughput efficiency p is. However, the factor k/T is a measurement for the time the belt runs idle. So in order to obtain a high throughput efficiency and a low idle-time percentage, certain choices have to be made. Figures like figure 6 and 7 help in making choices about the system. Even though the system itself may look very elementary, its behaviour most of the time is rather complex and changes with every parameter that is adjusted. Analysis of such systems are very difficult to make and are certainly assisted by the use of a formal model of the system.

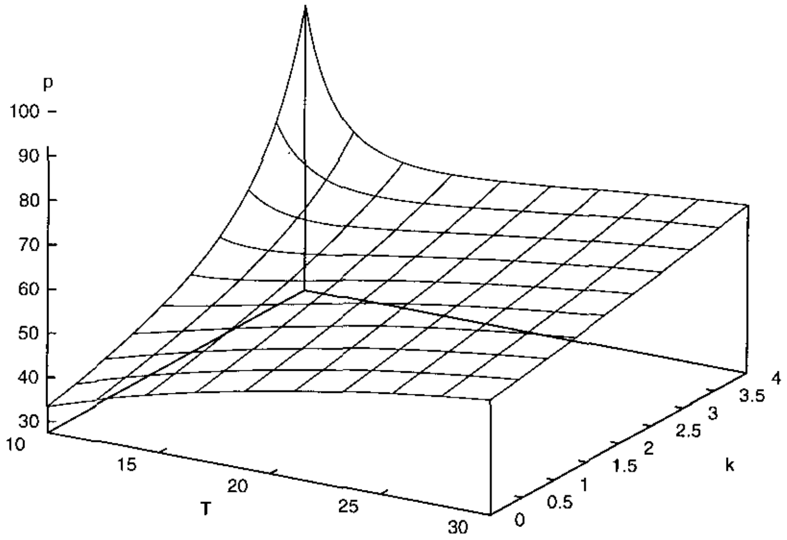

Figure 7: *The throughput efficiency for a variable maximal speed*

5 Related work

In the computing science literature, a few competing theories for hybrid systems have been proposed. In order to understand the value of timed μCRL, it is desirable to make a comparison with these theories. To this end, the next section is devoted to a brief introduction to one of the more popular theories, the theory of hybrid automata [14]. Subsequently, this theory is compared to timed μCRL.

5.1 Hybrid Automata

The first accounts of the theory of Hybrid Automata date back to 1993. In essence, the theory of hybrid automata is a generalisation of the theory of timed automata (see for instance [2]). The definition of a hybrid automaton discussed in here is as defined in [14]:

Definition 5.1 [Hybrid Automata] *A Hybrid Automata H consists of the following components.*

Variables *A finite set $X = \{x_1, \ldots, x_n\}$ of real-numbered variables. The number n is called the dimension of H. The set $\dot{X} = \{\dot{x_1}, \ldots, \dot{x_n}\}$, represents the first derivatives during continuous change and the set $X' = \{x'_1, \ldots, x'_n\}$ of primed variables are used to represent values at the conclusion of a discrete change.*

Control graph *A finite directed multigraph (V, E). The vertices in V are called control modes. The edges in E are called control switches.*

Initial, invariant and flow conditions *Three vertex labelling functions init, inv and flow that assign to each control mode $v \in V$ three predicates. Each initial condition init(v) is a predicate whose free variables are from X. Each invariant condition inv(v) is a predicate whose free vari-*

ables are from X. Each flow condition flow(v) is a predicate whose free variables are from $X \cup \dot{X}$.

Jump conditions *An edge labelling function jump that assigns to each control switch $e \in E$ a predicate. Each jump condition jump(e) is a predicate whose free variables are from $X \cup X'$.*

Events *A finite set Σ of events, and an edge labelling function event:$E \to \Sigma$ that assigns to each control switch an event.* □

The semantics for this model is given in terms of a *Labelled Transition System.* A labelled transition system is defined as follows:

Definition 5.2 [Labelled transition system] *A labelled transition system S consists of the following components.*

State space *A (possibly infinite) set Q of states, and a subset $Q^0 \subseteq Q$ of initial states.*

Transition relations *A (possibly infinite) set A of labels, and for each label $a \in A$, a binary relation $\xrightarrow{a}$ on the state space Q. Each triple $q \xrightarrow{a} q'$ is called a transition.*

A subset $R \subseteq Q$ of the state space is called a region. Given a region R and a label $a \in A$, we write $post_a(R) = \{q' | \exists q \in R : q \xrightarrow{a} q'\}$ for the region of a-successors of R, and we write $pre_a(R) = \{q | \exists q' \in R : q \xrightarrow{a} q'\}$ for the region of a-predecessors of R. □

Now, for a given hybrid automaton, two labelled transition systems are defined. Both transition systems represent discrete jumps by transitions. The timed transition system abstracts continuous flows by transitions, whereas the time-abstract transition system also abstracts from the duration of flows.

Definition 5.3 [Timed transition system] *The timed transition system S_H^t of the hybrid automaton H is the labelled transition system with the components Q, Q^0, A and $\xrightarrow{a}$ for each $a \in A$, defined as follows:*

- *Define $Q, Q^0 \subseteq V \times \mathbb{R}^n$ such that $(v, \mathbf{x}) \in Q$ iff the closed predicate inv(v)[$X := \mathbf{x}$] is true, and $(v, \mathbf{x}) \in Q^0$ iff both init(v)[$X := \mathbf{x}$] and inv(v)[$X := \mathbf{x}$] are true. The set Q is called the state space of H, and the subsets of Q are called H-regions.*
- $A = \Sigma \cup \mathbb{R}_{\geq 0}$.
- *For each event $\sigma \in \Sigma$, define $(v, \mathbf{x}) \xrightarrow{\sigma} (v', \mathbf{x}')$ iff there is a control switch $e \in E$ such that (1) the source of e is v and the target of e is v', (2) the closed predicate jump(e)[$X, X' := \mathbf{x}, \mathbf{x}'$] is true, and (3) event(e) $= \sigma$.*
- *For each nonnegative real $\delta \in \mathbb{R}_{\geq 0}$, define $(v, \mathbf{x}) \xrightarrow{\delta} (v', \mathbf{x}')$ iff $v = v'$ and there is a differentiable function $f : [0, \delta] \to \mathbb{R}^n$, with the first derivative $\dot{f} : (0, \delta) \to \mathbb{R}^n$, such that (1) $f(0) = \mathbf{x}$ and $f(\delta) = \mathbf{x}'$ and (2) for all reals $\epsilon \in (0, \delta)$, both inv(v)[$X := f(\epsilon)$] and flow(v)[$X, \dot{X} := f(\epsilon), \dot{f}(\epsilon)$] are true. The function f is called a witness for the transition $(v, \mathbf{x}) \xrightarrow{\delta} (x', \mathbf{x}')$.*

□

The time-abstract transition system is defined in a similar manner, but now the duration of a transition is abstracted from.

Definition 5.4 [Time-abstract transition system] *The time-abstract transition system S_H^a of the hybrid automaton H is the labelled transition system with the components Q, Q^0, B and $\xrightarrow{b}$ for each $b \in A$, defined as follows:*

- *Q and Q^0 are defined as for S^t_H,*
- *$B = \Sigma \cup \{\tau\}$, for some event $\tau \notin \Sigma$,*
- *For each event $\sigma \in \Sigma$, define $\xrightarrow{\sigma}$ as for S^t_H,*
- *Define $(v, \mathrm{x}) \xrightarrow{\tau} (v', \mathrm{x}')$ iff there is a nonnegative real $\delta \in \mathbb{R}_{\geq 0}$ such that for S^t_H, $(v, \mathrm{x}) \xrightarrow{\delta} (v, \mathrm{x}')$.*

□

5.2 Hybrid Automata versus timed μCRL

Thus far, several examples have shown that the theory of timed μCRL is fully capable of specifying hybrid systems and allows for performing subsequent analysis and verification on these models. The transitions between various models (e.g. the simplification of the original conveyor belt model) is quite natural and intuitive and involves only basic mathematics. There are differences, however, between the hybrid automata and models in timed μCRL.

5.2.1 Difference in operational semantics

The obvious difference between the theory of hybrid automata and the theory of timed μCRL is that the theory of hybrid automata explicitly describes the behaviour of the continuous variables in time, by allowing flow transitions. In fact, these flow transitions are primitive in the theory of hybrid automata. In timed μCRL, the primitive transitions are time steps. Moreover, in timed μCRL a timed transition does not fix values for conditions, but a time transition is only made if it is allowed by the conditions. To emphasise this difference, the following toy example is devised:

[**Example**] Consider a water tank which outputs water at a rate of $\dot{x} = 1$ in case the outlet is obstructed, and $\dot{x} = 2$ in case the outlet is not obstructed by a small object. If the tank is empty, an *empty* event is signalled and the tank is filled at a constant rate of $\dot{y} = 3$. Once the tank is full again, a *full* event is signalled and the tank again behaves as above. A timed μCRL description of this system is as follows:

proc $Drain(t : \mathbf{Time}, x_s : \mathbb{R}) =$

$$\Sigma_{u:\mathbf{Time}, x:\mathbb{R}\to\mathbb{R}}\ \mathit{empty}{}^{\backprime}u\ \mathit{Accumulate}(u, 0) \lhd \forall t' : 1 \leq \dot{x}(t') \leq 2 \wedge x(t) = x_s \wedge x(u) = 0 \rhd \delta{}^{\backprime}\mathbf{0}$$

proc $Accumulate(t : \mathbf{Time}, y_s : \mathbb{R}) =$

$$\Sigma_{u:\mathbf{Time}, y:\mathbb{R}\to\mathbb{R}}\ \mathit{full}{}^{\backprime}u\ \mathit{Drain}(u, -10) \lhd \forall t' : \dot{y}(t') = 3 \wedge y(u) = 10 \wedge y(t) = y_s \rhd \delta{}^{\backprime}\mathbf{0}$$

Figure 8: *A timed μCRL description for the watertank*

A hybrid automaton for this system is described in figure 9. Clearly, the continuous component is described differently in this system. However, the hybrid automaton of figure 9 is not the only description for this system. Replacing the flow condition $\dot{x} \in \{1, 2\}$ by $\dot{x} \in [1, 2]$ would specify the same behaviour.

The fact that, in timed μCRL the continuous system needs to be described using an interval instead of a finite set of behaviours, arises from the fact that in timed μCRL an idle transition (i.e. a transition that allows passage of time) does not affect the continuous behaviour directly. This may cause some problems in describing certain continuous processes in timed μCRL. However, it can also be argued that the theory of hybrid automata is not discriminative enough, since it does not distinguish the continuous behaviours $\dot{x} \in \{a, b\}$ and $\dot{x} \in [a, b]$.

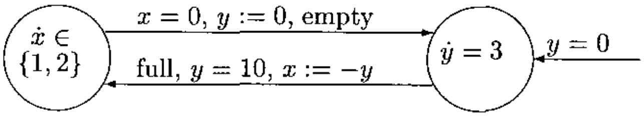

Figure 9: *A hybrid automaton for the watertank*

5.2.2 Difference in expressiveness

Apart from the difference in the operational semantics, it can be argued that the theory of timed μCRL is more expressive, in that it allows the description of systems that are not restricted to a behaviour described by DAEs with a constant first derivative (see f.i. the thermometer example in [11]). Formally, this cannot be expressed in hybrid automata. Although in timed μCRL, these expressions are not formalised, it is conceivable that they will be in the near future. A link could be by incorporating some computer algebra into the tools, which, in cooperation with the Sum Elimination Theorem, can reduce models the way it has been done by hand in this paper.

5.2.3 Tool support

The strong point for hybrid automata is the fact that tool support exists (see e.g. [13]). Although in the untimed setting, tool support also exists for μCRL, the timed extensions are not yet supported by tools. One can think of tool support that allows automated linearisation for timed processes, automating the expansion and encapsulation theorem and possibly also the automating of the proving (or falsifying) of absence of deadlock for systems. Obviously, more work needs to be done in this area.

5.2.4 Conceptual differences

The theory of hybrid automata differs somewhat in the notion of the parallel constructions. In timed μCRL, a notion of encapsulation is defined, which is not present in hybrid automata. Moreover, in timed μCRL, deadlocks and timed deadlocks can be denoted explicitly, whereas in hybrid automata these transitions are not present. This means that, in order to check for deadlocks, a reachability analysis needs to be performed to find the deadlocking locations and the transitions that lead to these locations. On the one hand, the deadlock transitions that arise from the application of the encapsulation operator are something to worry about, yet on the other hand, it is also a means of finding the deadlock transitions without first having to perform a reachability analysis. If some way of controlling the number of deadlock transitions that arise from the application of the encapsulation operator can be found, this may very well result in a preference for the theory of timed μCRL for the analysis of systems, since reachability analysis is a hard problem.

5.3 The theory of Hybrid I/O Automata

Besides the already existing hybrid automata, another notion of automata has been devised. In [17], a model is proposed that is based on timed automata and phase transition systems. One of the differences between this theory and the theory of hybrid automata is that not only communication via shared actions is possible, but also communication via shared variables. Furthermore, it is possible to express a hybrid automaton in the sense of [14] in the theory of hybrid I/O automata and the theory of hybrid I/O automata is regarded as being more expressive, since it allows for non-constant first derivative functions for the continuous behaviours. For a detailed account on the hybrid I/O automata see [18].

5.4 Concluding Remarks

The theory of timed μCRL lends itself perfectly well for the description of real time systems and the control of hybrid systems. The specification of these systems is relatively easy, and already quite a few results and theorems are available to perform transformations between various descriptions of systems. Although various issues are still to be solved, it is likely that the theory of timed μCRL will become a useful addition to already existing theories in the future. Various topics will remain future work, e.g. controlling deadlocks resulting from the expansion theorem, formalising the notion of continuous variables and differential algebraic equations, guidelines for performing analysis on timed μCRL descriptions, and developing tool assistance for validation and verification of systems. It should also be investigated whether the theory really scales up, or is only capable of describing problems in the theoretical domain instead of the industrial domain.

References

[1] J.-R. Abrial. The steam-boiler control specification problem. In J.-R. Abrial, E. Börger, and H. Langmaack, editors, *Formal Methods for Industrial Applications: Specifying and Programming the Steam Boiler Control*, number 1165 in LNCS. Springer Verlag, 1996.

[2] R. Alur and D.L. Dill. A theory of timed automata. In *Theoretical Computer Science*, volume 126, pages 183–235, 1994.

[3] J. C. M. Baeten and W. P . Weijland. *Process Algebra.* Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 1990.

[4] V. Bos and J.J.T. Kleijn. Structured operational semantics of χ. Computer Science Reports 01, Eindhoven University of Technology, 1999.

[5] Michael S. Branicky, Vivek S. Borkar, and Sanjoy K. Mitter. A unified framework for hybrid control: Model and optimal control theory. In *IEEE Transactions on Automatic Control*, volume 43, no. 1, pages 31–45, 1998.

[6] Charles M. Close and Dean K. Frederick. *Modeling and Analysis of Dynamic Systems.* Houghton Mifflin Company, international student edition edition, 1993.

[7] D. A. van Beek, J. E. Rooda, and S. H. F. Gordijn. Hybrid modelling in discrete-event control system design. In *Proceedings of the 1996 IEEE/IMACS Conference on Computational Engineering in Systems Applications*, 1996.

[8] Jan Friso Groote, Michel Reniers, Jos van Wamel, and Mark van der Zwaag. A Theoretical Basis for μCRL with Time. Unfinished revision of [12]., 1999.

[9] J.F. Groote. The syntax and semantics of timed μCRL. Technical Report SEN-R9709, CWI, June 1997.

[10] J.F. Groote and A. Ponse. The syntax and semantics of μ CRL. In A. Ponse, C. Verhoef, and S.F.M. van Vlijmen, editors, *Algebra of Communicating Processes*, Workshop in Computing, pages 26–62. Springer-Verlag, 1994.

[11] J.F. Groote and J.J. van Wamel. Analysis of three hybrid systems in timed μCRL. Technical Report SEN-R9815, CWI, September 1998.

[12] J.F. Groote and J.J. van Wamel. Basic theorems for parallel processes in timed μCRL. Software Engineering SEN-R9808, CWI, June 1998.

[13] T.A. Henzinger, P.-H. Ho, and H. Wong-Toi. HyTech: A model checker for hybrid systems. *Software Tools for Technology Transfer*, (1):110–122, 1997.

[14] Thomas A. Henzinger. The theory of hybrid automata. In *the Proceedings of the 11th Annual IEEE Symposium on Logic in Computer Science (LICS 96)*, pages 278–292, 1996.

[15] ISO/IEC. *A Formal Description Technique Based on the Temporal Ordering of Observational Behaviour.* ISO, 1988.

[16] John Lygeros, Datta N. Godbole, and Shankar Sastry. A game theoretic approach to hybrid system design. In *Hybrid Systems III*, number 1066 in LNCS, pages 1–12. Springer Verlag, 1996.

[17] Nancy Lynch, Roberto Segala, Frits Vaandrager, and H.B. Weinberg. Hybrid I/O automata. In *Hybrid Systems III*, number 1066 in LNCS, pages 496–510. Springer Verlag, 1996.

[18] Nancy Lynch, Roberto Segala, Frits Vaandrager, and H.B. Weinberg. Hybrid i/o automata. Technical report, University of Nijmegen, May 1999.

[19] R. Milner. *Communication and Concurrency.* Prentice Hall International, 1989.

[20] J. H. van Schuppen. Control for a class of hybrid systems. Technical Report PNA-R9716, CWI, November 1997.

A Selected proofs for deadlock removal

This section contains selected proofs of the removal of timed deadlock terms that were the result of an application of the encapsulation theorem. In essence, only the axioms of [9] and some lemmas of [12] are used. The only additional lemma that is used is the following:

Lemma Weakening:

$$A \rightarrow B \Rightarrow \delta\text{‘}u \triangleleft A \triangleright \delta\text{‘}\mathbf{0} \subseteq \delta\text{‘}u \triangleleft B \triangleright \delta\text{‘}\mathbf{0}$$

□

The validity of this lemma is easily proved from the definition of $\subseteq$.

$\delta 1a \subseteq 1$.

Proof:

$$\begin{aligned}
&\Sigma_{u,w:\mathbf{Time}}\delta\text{‘}u \triangleleft \sigma = 0 \wedge u \leq w \wedge 0 \leq a(w-t)+s \leq v_{max} \wedge \sigma_d = nod \triangleright \delta\text{‘}\mathbf{0}\\
=&\ \{\text{ axiom SUM3 }\}\\
&\Sigma_{w:\mathbf{Time}}\Sigma_{u:\mathbf{Time}}(\delta\text{‘}w + \delta\text{‘}u \triangleleft u \leq w \triangleright \delta\text{‘}\mathbf{0}) \triangleleft \sigma = 0 \wedge 0 \leq a(w-t)+s \leq v_{max} \wedge \sigma_d = nod \triangleright \delta\text{‘}\mathbf{0}\\
=&\ \{\text{ axiom ATA2 }\}\\
&\Sigma_{w:\mathbf{Time}}\Sigma_{u:\mathbf{Time}}\delta\text{‘}w \triangleleft \sigma = 0 \wedge 0 \leq a(w-t)+s \leq v_{max} \wedge \sigma_d = nod \triangleright \delta\text{‘}\mathbf{0}\\
\subseteq&\ \{\text{ definition } \subseteq\text{, axiom ATA2 }\}\\
&\Sigma_{w:\mathbf{Time}}ar\text{‘}w \triangleleft \sigma = 0 \wedge 0 \leq a(w-t)+s \leq v_{max} \wedge \sigma_d = nod \triangleright \delta\text{‘}\mathbf{0}
\end{aligned}$$

□

$\delta 1b \subseteq 1$.

Proof:

$$\begin{aligned}
&\Sigma_{u,w:\mathbf{Time}}\delta\text{‘}w \triangleleft \sigma = 0 \wedge w \leq u \wedge 0 \leq a(w-t)+s \leq v_{max} \wedge \sigma_d = nod \triangleright \delta\text{‘}\mathbf{0}\\
\subseteq&\ \{\text{ weakening law }\}\\
&\Sigma_{u,w:\mathbf{Time}}\delta\text{‘}w \triangleleft \sigma = 0 \wedge 0 \leq a(w-t)+s \leq v_{max} \wedge \sigma_d = nod \triangleright \delta\text{‘}\mathbf{0}\\
\subseteq&\ \{\text{ axiom ATA2 }\}\\
&\Sigma_{w:\mathbf{Time}}(ar\text{‘}w) \triangleleft \sigma = 0 \wedge 0 \leq a(w-t)+s \leq v_{max} \wedge \sigma_d = nod \triangleright \delta\text{‘}\mathbf{0}
\end{aligned}$$

□

$\delta 2a \subseteq 17$.

Proof:

$$\begin{aligned}
&\Sigma_{u:\mathbf{Time}}\delta\text{‘}(t-\tfrac{s}{a}) \triangleleft \sigma = 0 \wedge t-\tfrac{s}{a} \leq u \wedge a \neq 0 \wedge \sigma_d = nod \triangleright \delta\text{‘}\mathbf{0}\\
\subseteq&\ \{\text{ weakening law }\}\\
&\delta\text{‘}(t-\tfrac{s}{a}) \triangleleft \sigma = 0 \wedge a \neq 0 \wedge \sigma_d = nod \triangleright \delta\text{‘}\mathbf{0}\\
\subseteq&\ \{\text{ axiom ATA2 }\}\\
&F(0)\text{‘}(t-\tfrac{s}{a}) \triangleleft \sigma = 0 \wedge a \neq 0 \wedge \sigma_d = nod \triangleright \delta\text{‘}\mathbf{0}
\end{aligned}$$

□

$\delta 2b \subseteq 17$.

Proof:

$$\begin{aligned}
&\Sigma_{u:\mathbf{Time}}\delta\text{‘}u \triangleleft \sigma = 0 \wedge u \leq t-\tfrac{s}{a} \wedge a \neq 0 \wedge \sigma_d = nod \triangleright \delta\text{‘}\mathbf{0}\\
=&\ \{\text{ Lemma A.3.3 }\}\\
&(\Sigma_{u:\mathbf{Time}}\delta\text{‘}u \triangleleft u \leq t-\tfrac{s}{a} \triangleright \delta\text{‘}\mathbf{0}) \triangleleft \sigma = 0 \wedge a \neq 0 \wedge \sigma_d = nod \triangleright \delta\text{‘}\mathbf{0}\\
\subseteq&\ \{\text{ axiom ATA2 }\}\\
&\Sigma_{u:\mathbf{Time}}F(0)\text{‘}(t-\tfrac{s}{a}) \triangleleft \sigma = 0 \wedge a \neq 0 \wedge \sigma_d = nod \triangleright \delta\text{‘}\mathbf{0}\\
=&\ \{\text{ axiom SUM1 }\}
\end{aligned}$$

$F(0)\text{‹}(t - \frac{s}{a}) \lhd \sigma = 0 \wedge a \neq 0 \wedge \sigma_d = nod \rhd \delta\text{‹}\mathbf{0}$

□

The next two proofs typically state the need for invariants to prove absence of deadlock. Using the invariant I_6, we are able to conclude that $a > 0$, so a division by a is possible.

$\delta 8a \subseteq 22$
Proof:

$\Sigma_{w:\mathbf{Time}}\delta\text{‹}(t + \frac{1}{a}(\sqrt{s^2 + a(l-2x)} - s)) \lhd \sigma = 1 \wedge t + \frac{1}{a}(\sqrt{s^2 + a(l-2x)} - s) \leq w \wedge$
$\quad m = am \wedge n = nac \wedge 0 \leq a(w-t) + s \leq v_{max} \wedge a \neq 0 \rhd \delta\text{‹}\mathbf{0}$
$\subseteq$ { rewrite condition under invariance of I_6, weakening }
$\Sigma_{w:\mathbf{Time}}\delta\text{‹}(t + \frac{1}{a}(\sqrt{s^2 + a(l-2x)} - s)) \lhd \sigma = 1 \wedge t + \frac{1}{a}(\sqrt{s^2 + a(l-2x)} - s) \leq w \wedge$
$\quad m = am \wedge n = nac \wedge w \leq t + \frac{1}{a}(v_{max} - s) \wedge a \neq 0 \rhd \delta\text{‹}\mathbf{0}$
$\subseteq$ { weakening, transitivity, axiom SUM1 }
$\delta\text{‹}(t + \frac{1}{a}(\sqrt{s^2 + a(l-2x)} - s)) \lhd \sigma = 1 \wedge t + \frac{1}{a}(\sqrt{s^2 + a(l-2x)} - s)) \leq t + \frac{1}{a}(v_{max} - s) \wedge a \neq 0 \rhd \delta\text{‹}\mathbf{0}$
$\subseteq$ { axiom ATA2 }
$F(0)\text{‹}(t + \frac{1}{a}(v_{max} - s) \lhd \sigma = 1 \wedge a \neq 0 \rhd \delta\text{‹}\mathbf{0}$

□

$\delta 8b \subseteq 22$
Proof:

$\Sigma_{w:\mathbf{Time}}\delta\text{‹}w \lhd \sigma = 1 \wedge w \leq t + \frac{1}{a}(\sqrt{s^2 + a(l-2x)} - s) \wedge$
$\quad m = am \wedge n = nac \wedge 0 \leq a(w-t) + s \leq v_{max} \wedge a \neq 0 \rhd \delta\text{‹}\mathbf{0}$
$\subseteq$ { rewrite condition under invariance of I_6, weakening }
$\Sigma_{w:\mathbf{Time}}\delta\text{‹}w \lhd \sigma = 1 \wedge w \leq t + \frac{1}{a}(\sqrt{s^2 + a(l-2x)} - s) \wedge$
$\quad m = am \wedge n = nac \wedge w \leq t + \frac{1}{a}(v_{max} - s) \wedge a \neq 0 \rhd \delta\text{‹}\mathbf{0}$
$\subseteq$ { weakening, axiom SUM1, ATA2 }
$mid\text{‹}(t + \frac{1}{a}(v_{max} - s)) \lhd \sigma = 1 \wedge a \neq 0 \rhd \delta\text{‹}\mathbf{0}$

□

B The theory timed μCRL

This Appendix is devoted to a concise description of the theory of timed μCRL, based on a yet to appear paper [8], which is a revision of [12].
The theory of timed μCRL is a conservative extension of the theory of μCRL, in which time was not yet introduced. First the axiom system pCRL$_t$ for *pico* CRL with time is presented.

B.1 Axioms for pCRL$_t$

Let A be a set of atomic actions. These actions can be parameterised with data. Without loss of generality we assume that all actions have exactly one parameter. This is denoted by the set $\mathcal{AT}$. If the special symbol δ is added, modelling inaction or *deadlock*, the set is written $\mathcal{AT}_\delta$. Table 1 refers to the conventions used in this appendix.
The two elementary operators that are used to construct processes are $(+)$ the alternative composition $(\cdot)$ and the sequential composition. For two process terms p and q, the alternative composition $p + q$ behaves like p or q. The sequential composition $p \cdot q$ performs the actions of p until p terminates, and then continues with the actions in q. Binding conventions are that $\cdot$ binds stronger than $+$. In table 2, the axioms A1-A5 describe the elementary properties of the sequential and alternative composition. Axioms A6$^-$ and A7 are rules for the inaction. Furthermore, one additional operator is added, viz. an *if then else* construction, denoted $\cdots \lhd \cdots \rhd \cdots$, for which the

variable	range
x, y, z	process variables
p, q, r	process terms
X, Y	functions from data to processes
D, E	data sorts
d, e	data variables of arbitrary sort
b, c, α, β	variables of sort **Bool**
t, u, v	variables of sort **Time**
a, b, c	actions or elements from $\mathcal{AT}$ or $\mathcal{AT}_\delta$

Table 1: *Conventions*

axioms C1 and C2 are given. In general, for $n > 0$ finite sums $p_1 + \cdots + p_n$ are abbreviated by $\Sigma_{i \in I} p_i$ where $I = \{1, \cdots, n\}$. In pCRL$_t$, a summation construct of the form $\Sigma_{d:D} p$ is a binder of variable d of data sort D in process term p, in which D may be infinite. By convention, we write $\Sigma_{i \in \emptyset} p = \delta\text{‘}\mathbf{0}$.

A1	$x + y = y + x$	SUM1	$\Sigma_{d:D} x = x$
A2	$x + (y + z) = (x + y) + z$	SUM3	$\Sigma X = Xd + \Sigma X$
A3	$x + x = x$	SUM4	$\Sigma_{d:D}(Xd + Yd) = \Sigma X + \Sigma Y$
A4	$(x + y) \cdot z = x \cdot z + y \cdot z$	SUM5	$(\Sigma X) \cdot x = \Sigma_{d:D}(X \cdot dx)$
A5	$(x \cdot y) \cdot z = x \cdot (y \cdot z)$	SUM11	$(\forall d \in D\ Xd = Yd) \rightarrow \Sigma X = \Sigma Y$
A6$^-$	$a + \delta = a$		
A7	$\delta \cdot x = \delta$	C1	$x \lhd \mathbf{t} \rhd y = x$
		C2	$x \lhd \mathbf{f} \rhd y = y$

Table 2: *Core axioms of pCRL$_t$, where $a \in \mathcal{AT}_\delta$*

In the axioms SUM1, SUM3, SUM4, SUM5 and SUM11, a distinction is made between *sum operators* Σ and *sum constructs* $\Sigma_{d:D} p$. And X in ΣX may be instantiated with functions from some data sort to the sort of processes, such as $\lambda d : D.p$, where variable d in p may *not* become bound by Σ. An expression $\Sigma_{d:D} x$, where some term p is substituted for x, it may *not* contain a free variable d. Data terms are considered modulo *α-conversion.*

The time line in pCRL$_t$ can be any structure, as long as it adheres to the axioms in table 3.

Time1	*if* $t \leq u \wedge u \leq v = \mathbf{t}$ *then* $t \leq v = \mathbf{t}$
Time3	$t \leq u \vee u \leq t = \mathbf{t}$
Time4	*if* $t \leq u \wedge u \leq t = \mathbf{t}$ *then* $t = u$
Time5	$eq(t, u) = t \leq u \wedge u \leq t$
Time6	$min(t, u) = if(t \leq u, t, u)$
Time7	$if(\mathbf{t}, t, u) = t$
Time8	$if(\mathbf{f}, t, u) = u$

Table 3: *Axioms for time*

The way to express that actions must occur at certain points in time is expressed using the *"at"* operator, denoted by the special symbol ‘. A process $p\text{‘}t$ behaves like process p with the restriction that its first action must take place at time t. Table 4 lists the axioms for $p\mathrm{CRL}_t$.

ATA1	$x = \Sigma_{t:\mathbf{Time}}\ x\text{‘}t$
ATA2	$a\text{‘}t\ x = a\text{‘}t(t \gg x)$
ATA6	$x + \delta\text{‘}\mathbf{0} = x$
ATB1	$a\text{‘}t\text{‘}u = (a\text{‘}t \lhd u \leq t \rhd \delta\text{‘}t) \lhd t \leq u \rhd \delta\text{‘}u$
ATB2	$(x + y)\text{‘}t = x\text{‘}t + y\text{‘}t$
ATB3	$(x \cdot y)\text{‘}t = x\text{‘}ty$
ATB4	$(\Sigma_{d:D}Xd)\text{‘}t = \Sigma_{d:D}Xd\text{‘}t$
$\gg 1$	$t \gg x = \Sigma_{u:\mathbf{Time}}x\text{‘}u \lhd t \leq u \rhd \delta\text{‘}t$

Table 4: *Time related axioms of pCRL$_t$, where $a \in \mathcal{AT}_\delta$*

The process $\delta\text{‘}t$ models the process that deadlocks at moment t. These processes are called *time deadlocks*. Using axiom ATA1, we can write $\delta = \Sigma_{t:\mathbf{Time}}\delta\text{‘}t$. For arbitrary time t, the following holds: $\delta\text{‘}t\ +\ \delta = \delta$. Axiom AT6 expresses that $\delta\text{‘}\mathbf{0}$ serves as a neutral element for the alternative composition. We require that $\mathbf{0}$ is the minimal element for **Time**.

B.2 Axioms for $\mu\mathrm{CRL}_t$

This section is devoted to the extension of $p\mathrm{CRL}_t$ with parallelism, called the axiom system $\mu\mathrm{CRL}_t$. For communication we have a binary commutative and associative function γ, which is only defined on *action names*. In order for a communication to occur between action terms $a(d), b(e) \in \mathcal{AT}$, $\gamma(a, b)$ should be defined, and the data parameters d and e of the action terms should match according to axiom CF in table 6.

The three basic operators for concurrency are the *merge* $\|$, the *left merge* $\lfloor\!\lfloor$ and the *communication merge* $|$. The process $p\|q$ symbolises the parallel execution of p and q. It 'starts' with an action of either p or q, or with the communication between p and q. We write $p \lfloor\!\lfloor\, q$ for a process which behaves like $p\|q$, but the first action must come from p.

For the axiomatisation of the left merge $\lfloor\!\lfloor$, the auxiliary *before* operator is defined; $p \ll q$ should be interpreted as the process that behaves like p, provided that p can do a step before or at the moment t_0 after which q gets definitely disabled. Otherwise $p \ll q$ becomes a time deadlock at time t_0. The axioms related to the parallelism are listed in tables 5 and 6.

ATB7	$(x\|y)\text{‘}t = x\text{‘}t\|y$
ATB8	$(x\|y)\text{‘}t = x\|y\text{‘}t$
ATB9	$\partial_H(x)\text{‘}t = \partial_H(x\text{‘}t)$
$\ll 1$	$x \ll a\text{‘}t = \Sigma_{u:\mathbf{Time}}\ x\text{‘}u \lhd u \leq t \rhd x\text{‘}t$
$\ll 2$	$x \ll (y + z) = x \ll y + x \ll z$
$\ll 3$	$x \ll y \cdot z = x \ll y$
$\ll 4$	$x \ll \Sigma X = \Sigma_{d:D}x \ll Xd$

Table 5: *Time related axioms of μCRL$_t$, where $a \in \mathcal{AT}_\delta$ and $H \subseteq A$*

SUM6	$(\Sigma X) \mathbin{\|\!\lfloor} x = \Sigma_{d:D}(Xd \mathbin{\|\!\lfloor} x)$		
SUM7	$(\Sigma X)\|x = \Sigma_{d:D}(Xd\|x)$		
SUM7'	$x\|(\Sigma X) = \Sigma_{d:D}(x\|Xd)$		
SUM8	$\partial_H(\Sigma X) = \Sigma_{d:D}\partial_H(Xd)$		
CM1	$x\|\|y = x \mathbin{\|\!\lfloor} y + y \mathbin{\|\!\lfloor} x + x\|y$	CD1	$\delta\|a = \delta$
CM2	$a\text{‘}t \mathbin{\|\!\lfloor} x = (a\text{‘}t \ll x) \cdot x$	CD2	$a\|\delta = \delta$
CM3	$a\text{‘}t \cdot x \mathbin{\|\!\lfloor} y = (a\text{‘}t \ll y) \cdot (t \gg x\|\|y)$		
CM4	$(x + y) \mathbin{\|\!\lfloor} z = x \mathbin{\|\!\lfloor} z + y \mathbin{\|\!\lfloor} z$	DD	$\partial_H(\delta) = \delta$
CM5	$a \cdot x\|b = (a\|b) \cdot x$		
CM6	$a\|b \cdot x = (a\|b) \cdot x$	D1	$\partial_H(c(d)) = c(d)$ if $c \notin H$
CM7	$a \cdot x\|b \cdot y = (a\|b) \cdot (x\|\|y)$	D2	$\partial_H(c(d))$ if $c \in H$
CM8	$(x + y)\|z = x\|z + y\|z$	D3	$\partial_H(x + y) = \partial_H(x) + \partial_H(y)$
CM9	$x\|(y + z) = x\|y + x\|z$	D4	$\partial_H(x \cdot y) = \partial_H(x) \cdot \partial_H(y)$
CF	$c(d)\|c'(e) = \begin{cases} \gamma(c,c')(d) \lhd eq(d,e) \rhd \delta \text{ if } \gamma(c,c') \text{ defined} \\ \delta \text{ otherwise} \end{cases}$		

Table 6: *Axioms for parallelism of μCRL_t, where $a, b \in \mathcal{AT}_\delta$, $c, c' \in A$ and $H \subseteq A$*

B.3 Basic forms

A basic syntactic format for $\Sigma(p\mathrm{CRL}_t)$-terms is provided.

Definition B.1 *A basic form over $\Sigma(pCRL_t)$-terms is a process-closed term of the form*

$$\begin{aligned} r \quad = \quad & \Sigma_{i\in I}\Sigma_{d_1^i:D_1^i}\cdots\Sigma_{d_{m_i}^i:D_{m_i}^i}\Sigma_{u:\mathbf{Time}}\; a_i\text{‘}u \cdot r_i \lhd \alpha_i \rhd \delta\text{‘}\mathbf{0}+ \\ & \Sigma_{j\in J}\Sigma_{e_1^j:E_1^j}\cdots\Sigma_{e_{n_j}^j:E_{n_j}^j}\Sigma_{v:\mathbf{Time}}\; b_j\text{‘}v \lhd \beta_j \rhd \delta\text{‘}\mathbf{0} \end{aligned}$$

where $a_i \in \mathcal{A}$ and $b_j \in \mathcal{AT}_\delta$, and the r_i are also basic forms. □

In the sequel, we will often write $\overline{\Sigma}_{d_1,\cdots,d_m}x$ for $\Sigma_{d_1:D_1}\cdots\Sigma_{d_m:D_m}$ and $\overline{d}_m$ for $d_1,\cdots,d_m$. By convention $\overline{\Sigma}_{\overline{d}_0}x = x$, and it can be proved that the order of the d_k in $\overline{\Sigma}_{\overline{d}_0}x = x$ may be permutated arbitrarily. We take care that no confusion can arise w.r.t. the sorts of the d_k. For example, if we treat $\Sigma_{i\in I}$ and $\Sigma_{j\in J}$ as formal summations we may abbreviate r in the above definition to:

$$\overline{\Sigma}_{i,\overline{d}_{m_i}^i,u}\; a_i\text{‘}ur_i \lhd \alpha_i \rhd \delta\text{‘}\mathbf{0} + \overline{\Sigma}_{j,\overline{e}_{n_j}^j,v}\; b_j\text{‘}v \lhd \beta_j \rhd \delta\text{‘}\mathbf{0}.$$

A more general format for representing basic forms is defined as follows:

Lemma B.1 Representation *Basic form r given in Definition B.1 can be represented by*

$$\overline{\Sigma}_{i,\overline{d}_m,u}\; a_i\text{‘}ur_i \lhd \alpha_i \rhd \delta\text{‘}\mathbf{0} + \overline{\Sigma}_{j,\overline{e}_n,v}\; b_j\text{‘}v \lhd \beta_j \rhd \delta\text{‘}\mathbf{0}.$$

where the sequence $d_1,\cdots,d_m$ contains all data variables from $\cup_{i\in I}\{d_1^i,\cdots,d_{m_i}^i\}$, and $e_1,\cdots,e_n$ contains all data variables from $\cup_{j\in J}\{e_1^j,\cdots,e_{n_j}^j\}$. □

Theorem B.1 Basic Forms *If q is a process-closed term over $\Sigma(pCRL_t)$ then there is a basic form p such that $\mu CRL_t \vdash\; p = q$.* □

Computing Science Reports

Department of Mathematics and Computing Science
Eindhoven University of Technology

In this series appeared:

96/01	M. Voorhoeve and T. Basten	Process Algebra with Autonomous Actions, p. 12.
96/02	P. de Bra and A. Aerts	Multi-User Publishing in the Web: DreSS, A Document Repository Service Station, p. 12
96/03	W.M.P. van der Aalst	Parallel Computation of Reachable Dead States in a Free-choice Petri Net, p. 26.
96/04	S. Mauw	Example specifications in phi-SDL.
96/05	T. Basten and W.M.P. v.d. Aalst	A Process-Algebraic Approach to Life-Cycle Inheritance Inheritance = Encapsulation + Abstraction, p. 15.
96/06	W.M.P. van der Aalst and T. Basten	Life-Cycle Inheritance A Petri-Net-Based Approach, p. 18.
96/07	M. Voorhoeve	Structural Petri Net Equivalence, p. 16.
96/08	A.T.M. Aerts, P.M.E. De Bra, J.T. de Munk	OODB Support for WWW Applications: Disclosing the internal structure of Hyperdocuments, p. 14.
96/09	F. Dignum, H. Weigand, E. Verharen	A Formal Specification of Deadlines using Dynamic Deontic Logic, p. 18.
96/10	R. Bloo, H. Geuvers	Explicit Substitution: on the Edge of Strong Normalisation, p. 13.
96/11	T. Laan	AUTOMATH and Pure Type Systems, p. 30.
96/12	F. Kamareddine and T. Laan	A Correspondence between Nuprl and the Ramified Theory of Types, p. 12.
96/13	T. Borghuis	Priorean Tense Logics in Modal Pure Type Systems, p. 61
96/14	S.H.J. Bos and M.A. Reniers	The I^2C-bus in Discrete-Time Process Algebra, p. 25.
96/15	M.A. Reniers and J.J. Vereijken	Completeness in Discrete-Time Process Algebra, p. 139.
96/17	E. Boiten and P. Hoogendijk	Nested collections and polytypism, p. 11.
96/18	P.D.V. van der Stok	Real-Time Distributed Concurrency Control Algorithms with mixed time constraints, p. 71.
96/19	M.A. Reniers	Static Semantics of Message Sequence Charts, p. 71
96/20	L. Feijs	Algebraic Specification and Simulation of Lazy Functional Programs in a concurrent Environment, p. 27.
96/21	L. Bijlsma and R. Nederpelt	Predicate calculus: concepts and misconceptions, p. 26.
96/22	M.C.A. van de Graaf and G.J. Houben	Designing Effective Workflow Management Processes, p. 22.
96/23	W.M.P. van der Aalst	Structural Characterizations of sound workflow nets, p. 22.
96/24	M. Voorhoeve and W. van der Aalst	Conservative Adaption of Workflow, p.22
96/25	M. Vaccari and R.C. Backhouse	Deriving a systolic regular language recognizer, p. 28
97/01	B. Knaack and R. Gerth	A Discretisation Method for Asynchronous Timed Systems.
97/02	J. Hooman and O. v. Roosmalen	A Programming-Language Extension for Distributed Real-Time Systems, p. 50.
97/03	J. Blanco and A. v. Deursen	Basic Conditional Process Algebra, p. 20.
97/04	J.C.M. Baeten and J.A. Bergstra	Discrete Time Process Algebra: Absolute Time, Relative Time and Parametric Time, p. 26.
97/05	J.C.M. Baeten and J.J. Vereijken	Discrete-Time Process Algebra with Empty Process, p. 51.
97/06	M. Franssen	Tools for the Construction of Correct Programs: an Overview, p. 33.

97/07	J.C.M. Baeten and J.A. Bergstra	Bounded Stacks, Bags and Queues, p. 15.
97/08	P. Hoogendijk and R.C. Backhouse	When do datatypes commute? p. 35.
97/09	Proceedings of the Second International Workshop on Communication Modeling, Veldhoven, The Netherlands, 9-10 June, 1997.	Communication Modeling- The Language/Action Perspective, p. 147.
97/10	P.C.N. v. Gorp, E.J. Luit, D.K. Hammer E.H.L. Aarts	Distributed real-time systems: a survey of applications and a general design model, p. 31.
97/11	A. Engels, S. Mauw and M.A. Reniers	A Hierarchy of Communication Models for Message Sequence Charts, p. 30.
97/12	D. Hauschildt, E. Verbeek and W. van der Aalst	WOFLAN: A Petri-net-based Workflow Analyzer, p. 30.
97/13	W.M.P. van der Aalst	Exploring the Process Dimension of Workflow Management, p. 56.
97/14	J.F. Groote, F. Monin and J. Springintveld	A computer checked algebraic verification of a distributed summation algorithm, p. 28
97/15	M. Franssen	λP-: A Pure Type System for First Order Loginc with Automated Theorem Proving, p.35.
97/16	W.M.P. van der Aalst	On the verification of Inter-organizational workflows, p. 23
97/17	M. Vaccari and R.C. Backhouse	Calculating a Round-Robin Scheduler, p. 23.
97/18	Werkgemeenschap Informatiewetenschap redactie: P.M.E. De Bra	Informatiewetenschap 1997 Wetenschappelijke bijdragen aan de Vijfde Interdisciplinaire Conferentie Informatiewetenschap, p. 60.
98/01	W. Van der Aalst	Formalization and Verification of Event-driven Process Chains, p. 26.
98/02	M. Voorhoeve	State / Event Net Equivalence, p. 25
98/03	J.C.M. Baeten and J.A. Bergstra	Deadlock Behaviour in Split and ST Bisimulation Semantics, p. 15.
98/04	R.C. Backhouse	Pair Algebras and Galois Connections, p. 14
98/05	D. Dams	Flat Fragments of CTL and CTL*: Separating the Expressive and Distinguishing Powers. P. 22.
98/06	G. v.d. Bergen, A. Kaldewaij V.J. Dielissen	Maintenance of the Union of Intervals on a Line Revisited, p. 10.
98/07	Proceedings of the workhop on Workflow Management: Net-based Concepts, Models, Techniques and Tools (WFM'98) June 22, 1998 Lisbon, Portugal	edited by W. v.d. Aalst, p. 209
98/08	Informal proceedings of the Workshop on User Interfaces for Theorem Provers. Eindhoven University of Technology ,13-15 July 1998	edited by R.C. Backhouse, p. 180
98/09	K.M. van Hee and H.A. Reijers	An analytical method for assessing business processes, p. 29.
98/10	T. Basten and J. Hooman	Process Algebra in PVS
98/11	J. Zwanenburg	The Proof-assistemt Yarrow, p. 15
98/12	Ninth ACM Conference on Hypertext and Hypermedia Hypertext '98 Pittsburgh, USA, June 20-24, 1998 Proceedings of the second workshop on Adaptive Hypertext and Hypermedia.	Edited by P. Brusilovsky and P. De Bra, p. 95.
98/13	J.F. Groote, F. Monin and J. v.d. Pol	Checking verifications of protocols and distributed systems by computer. Extended version of a tutorial at CONCUR'98, p. 27.
98/14	T. Verhoeff (artikel volgt)	
99/01	V. Bos and J.J.T. Kleijn	Structured Operational Semantics of χ , p. 27
99/02	H.M.W. Verbeek, T. Basten and W.M.P. van der Aalst	Diagnosing Workflow Processes using Woflan, p. 44

99/03	R.C. Backhouse and P. Hoogendijk	Final Dialgebras: From Categories to Allegories, p. 26
99/04	S. Andova	Process Algebra with Interleaving Probabilistic Parallel Composition, p. 81
99/05	M. Franssen, R.C. Veltkamp and W. Wesselink	Efficient Evaluation of Triangular B-splines, p. 13
99/06	T. Basten and W. v.d. Aalst	Inheritance of Workflows: An Approach to tackling problems related to change, p. 66
99/07	P. Brusilovsky and P. De Bra	Second Workshop on Adaptive Systems and User Modeling on the World Wide Web, p. 119.
99/08	D. Bosnacki, S. Mauw, and T. Willemse	Proceedings of the first international syposium on Visual Formal Methods - VFM'99
99/09	J. v.d. Pol, J. Hooman and E. de Jong	Requirements Specification and Analysis of Command and Control Systems