Discrete real-time and stochastic-time process algebra for performance analysis of distributed systems

Citation for published version (APA):

Document status and date:
Published: 01/01/2008

Document Version:
Publisher’s PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:
• A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
• The final author version and the galley proof are versions of the publication after peer review.
• The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
• You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the “Taverne” license above, please follow below link for the End User Agreement:
www.tue.nl/taverne

Take down policy
If you believe that this document breaches copyright please contact us at:
openaccess@tue.nl
providing details and we will investigate your claim.

Download date: 04. Oct. 2023
Discrete Real-Time and Stochastic-Time Process Algebra for Performance Analysis of Distributed Systems

J. Markovski* and E.P. de Vink

Formal Methods Group, Eindhoven University of Technology
Den Dolech 2, 5612 AZ Eindhoven, The Netherlands
tel: +31 40 247 5158, fax: +31 40 247 5361
j.markovski@tue.nl, evink@win.tue.nl

Abstract. We present a process algebra with conditionally distributed discrete-time delays and generally-distributed stochastic delays. It features two types of race conditions in terms of conditional random variables. Building on the new theory we analyze extensions of timed process algebra with discrete stochastic time. In the new setting, typical standard notions like time additivity are hard to preserve in the presence of the race condition. We propose context-sensitive interpolation as a restricted form of time additivity to accommodate the extension with stochastic time. The approach enables compositional modeling, a non-trivial expansion law, and explicit manipulation of maximal progress. The approach is illustrated by a specification of the Concurrent Alternating Bit Protocol with unreliable generally-distributed channels in the language $\chi$. We compare performance analysis using discrete timed probabilistic reward graphs and discrete-event simulation.

1 Introduction

Over the past decade stochastic process algebras emerged as compositional modeling formalisms for systems that do not only require functional verification, but performance analysis as well. Many Markovian processes algebras were developed like EMPA, PEPA, IMC, etc. exploiting the memoryless property of the exponential distribution. Before long, the need for general distributions arose, as exponential delays were not sufficient to model, for example, fixed timeouts of the Internet protocols or heavy-tail distributions present in the media streaming services. Prominent stochastic process algebras with general distributions include TIPP, GSMPA, SPADES, IGSMP, NMSPA, and MODEST [1–6].

Despite the greater expressiveness, compositional modeling with general distributions proved to be challenging, as the memoryless property cannot be relied on [7, 8]. Typically, the underlying performance model is a generalized semi-Markov process that exploits clocks to memorize past behavior in order to retain

* This research has been funded by the Dutch BSIK/BRICKS project AFM 3.2
the Markov property of history independence [9]. Similarly, the semantics of stochastic process algebras is given using clocks that represent the stochastic delays at the symbolic level. Such a symbolic representation allows for the manipulation of finite structures, e.g., stochastic automata or extensions of generalized semi-Markov processes. The concrete execution model is subsequently obtained by sampling the clocks, frequently yielding infinite probabilistic timed transition systems.

For the sampling of the clock two execution policies can be adopted: (1) race condition [1, 3, 5, 6], which enables the action transitions guarded by the clocks that expire first, and (2) pre-selection policy [2, 4], which preselects the clocks by a probabilistic choice. To keep track of past behavior, the clock samples have to be updated after each stochastic delay transition. One can do this in two equivalent ways: (1) by keeping track of residual lifetimes [3, 6], i.e., the time left up to expiration, or (2) by keeping track of the spent lifetimes [1, 2, 4, 5], i.e., the time passed since activation. The former manner is more suitable for discrete-event simulation, whereas the latter is acknowledged for its correspondence to real-time semantics [7, 8].

In this paper we consider the race condition with spent-lifetime semantics. However, we do not use clocks to implement the race condition and to determine the winning stochastic delay(s) of the race. Rather, we rely on an equivalent interpretation that uses conditional random variables and makes a probabilistic assumption on the winners followed by conditioning of the distributions of the losers on the time spent for the winning samples [10]. Thus, we no longer speak of clocks as we do not keep track of sample lifetimes, but we only cater for the ages of the conditional distributions [11]. We refer to the samples as stochastic delays, a naming resembling standard timed delays.

The relation between real-time and stochastic time has been studied in various settings: a structural translation from stochastic automata to timed automata with deadlines is given in [12]. This approach found its way into MOD-EST [6] as means to introduce real-time and stochastic time as separate constructs in the same formalism. Also, a translation from IGSMP into pure real-time models called interactive timed automata is reported in [4]. Our previous work studied the interplay between standard timed delays and discrete stochastic delays in [13, 11]. An axiomatization for a process algebra that embeds real-time delays with so-called context-sensitive interpolation into a restricted form of discrete stochastic time is given in [11].

The contribution of the present paper is threefold. Firstly, a sound and ground-complete process theory is provided that accommodates timed delays in a racing context, extending the work of [13, 11]. The theory provides an explicit maximal progress operator and an non-trivial expansion law for the parallel composition. Different from other approaches, we derive stochastic delays as time-delayed processes with explicit information about the winners and the losers that induced the delay. We treat real time as Dirac stochastic time that actually induces a trivial race condition in which the shortest sample is always exhibited by the same set of delays and moreover has a fixed duration.
The theory also provides the possibility of specifying a partial race of stochastic delays, e.g., that one delay has always a shorter, equal, or longer sample than another one. This is required when modeling timed systems which correct behavior depends on the ordering of the durations of the timed delays, like for example, in a time dependent controller. When the timed delays are simply replaced by stochastic delays, the total order of the samples is, in general, lost, unless it is possible to specify which delays are the winners or losers of the imposed race.

Afterwards, we take exactly opposite approach by treating real time from the stochastic viewpoint and we reveal the other side of the same coin. Here, we treat timed and stochastic delays as ‘atomic’, rather than series of unit timed delays. This puts the timed delays on the same level with the stochastic ones as passage of time is studied in terms of discrete events, where the actual duration/sample of the delay plays a background role. The race condition remains the central notion in both settings. We investigate what needs to be in place to generalize timed delays to stochastic ones. Therefore, we analyze stochastic bisimulation as well as the fit of real-time features, like time determinism and time additivity, in a stochastic-time setting. This brings us to the notion of context-sensitive interpolation, which can be viewed as an interpretation of the race condition in the timed setting. We benefit from our findings in the development of a stochastic process algebra that retains many features of the timed process theories, but permits a restricted form of time additivity only.

We illustrate the theories by revisiting the $G/G/1/\infty$ queue from [13], treating it more elegantly, and by specifying a variant of the Concurrent Alternating Bit Protocol, CABP for short, that has fixed time-outs (represented by timed delays) and faulty generally distributed channels (represented by stochastic delays), stressing the interplay of real time and stochastic time.

Our third contribution concerns automated performance analysis. It is well known that only a small, restricted classes of models of general distributions are analytically solvable. Preliminary research on model checking of stochastic automata is reported in [14] and a proposal for model checking probabilistic timed systems is given in [15]. However, at the moment, the performance analysts turn to simulation when it comes to analyzing models with generally distributed delays. For the purpose of analyzing the specification of the CABP we depend on the toolset of the $\chi$-language [16]. At the start, $\chi$ was used to model discrete-event systems only, not supported by an explicit semantics. However, recently, it has been turned into a formal specification language set up as a process algebra with data [17]. In addition, in [18] a proposal was given to extend $\chi$ with a probabilistic choice to enable long-run performance analysis of probabilistic timed specifications using discrete-time probabilistic reward graphs (DTPRGs for short). We augment this prototype extension of the $\chi$-toolset to cater for transient analysis too. The case study illustrates the new approach with channel distributions that are deterministic.

Acknowledgments Many thanks to Jos Baeten, Nikola Trčka, Bas Luttik, and Walter van Niftrik for fruitful discussions on the topic.
2 Race Condition

In this section we provide the mathematical background and we postulate the central concepts of race condition, timed and stochastic delays.

2.1 Preliminaries

We use discrete random variables to represent durations of stochastic delays. The set of discrete distribution functions $F$ such that $F(n) = 0$ for $n \leq 0$ is denoted by $\mathcal{F}$; the set of the corresponding random variables by $\mathcal{V}$. We use $X$, $Y$, and $Z$ to range over $\mathcal{V}$ and $F_X$, $F_Y$, and $F_Z$ for their respective distribution functions. Also, $W$, $L$, $V$, and $D$ range over $2^\mathcal{V}$. By assumption, the support set $\text{supp}(X) = \{n > 0 \mid P(X = n) > 0\}$ of a random variable $X$ is finite or countably infinite. The domain $A$ of a function $f : A \rightarrow B$ is denoted by $\text{dom}(f)$. In case $f$ is bijective, we write $f : A \rightarrow B$. The identity bijection on the set $A$ is denoted by $\text{id}_A$.

We write $p \subseteq A$ for a predicate $p : A \rightarrow \{\top, \bot\}$, where $\top$ and $\bot$ denote the truth values true and false, respectively. Composition of two relations $r_1 \subseteq A \times B$ and $r_2 \subseteq B \times C$ is given by $r_2 \circ r_1 \subseteq A \times C$ where $(x, z) \in r_2 \circ r_1$ if there exists a $y \in B$ such that $(x, y) \in r_1$ and $(y, z) \in r_2$. We restrict and rename functions on disjoint parts of the domain by $g\{f_1/D_1\} \ldots \{f_n/D_n\}(x) = f_i(x)$ if $x \in D_i$, and $g(x)$ if $x \in D \setminus (\bigcup_{i=1}^n D_i)$, for functions $g, f_1, \ldots, f_n : A \rightarrow B$ and disjoint subsets $D_1, \ldots, D_n \subseteq A$. By $\mathcal{P}(A)$ we denote the set of standard probabilistic spaces $(A, \mathcal{P})$ over the set $A$ with probability measure $\mathcal{P}$.

2.2 Racing Stochastic Delays

A stochastic delay is a timed delay of a duration that is guided by a random variable. We use the random variable as the name of the stochastic delay. We observe simultaneous passage of time for a number of stochastic delays until one or some of them expire. This phenomenon is referred to as the race condition and the underlying process as the race. For multiple racing stochastic delays, different stochastic delays may be observed simultaneously as being the shortest. The ones that have the shortest duration are called winners, the others are referred to as losers. We illustrate the concepts by an example.

Example 1 (Race condition). Let $X$ and $Y$ be random variables with $P(X = 1) = P(X = 2) = P(X = 3) = \frac{1}{4}$ and $P(Y = 1) = \frac{1}{2}, P(Y = 2) = \frac{1}{4}, P(Y = 3) = \frac{1}{4}$. Now, let us assume that two delays $X$ and $Y$ are guided by the variables with the same name. The probability that $X$ wins the race is the probability $P(X < Y) = \frac{7}{12}$. Then, then winning delay is distributed as $W_X = \langle X \mid X < Y \rangle$ with $P(W_X = 1) = P(X = 1 \mid X < Y) = \frac{1}{2}$, $P(W_X = 2) = \frac{3}{4}$, and $P(W_X = 3) = \frac{1}{4}$. Similarly, the probability that $Y$ wins the race is the probability $P(Y < X) = \frac{5}{12}$. Then, then winning delay is distributed as $W_Y = \langle Y \mid Y < X \rangle$ with $P(W_Y = 1) = \frac{1}{4}$. Both, $X$ and $Y$ win the race together with probability $P(X = Y) = \frac{3}{12}$ and a winning delay distributed as $W_{XY} = \langle X \mid X = Y \rangle$ (or, the equivalent, $\langle Y \mid X = Y \rangle$) with $P(W_{XY} = 2) = \frac{2}{3}$ and $P(W_{XY} = 3) = \frac{1}{3}$.
An outcome of a race is completely determined by the winners and the losers. So, we can explicitly represent the outcome of the race by a pair of sets of stochastic delays $[W_i]$, where $W$ is the set of winners and $L$ is the set of losers. We have occasion to write $[W]$ instead of $[W_0]$ and omit the set brackets when clear from the context. Thus, $[X]$ represents a stochastic delay with name $X$, guided by the random variable $X$.

Outcomes of races may be involved in other races, so we refer to an outcome $[W_i]$ as a (conditional) stochastic delay induced by the disjoint sets of winners $W$ and losers $L$. The probability of the outcome $[W_i]$ is

$$P(X_1 = X_2 \text{ for } X_1, X_2 \in W \text{ and } X_3 < Y \text{ for } X_3 \in W, Y \in L)$$

and the stochastic delay is guided by the conditional random variable

$$\langle X \mid X_1 = X_2 \text{ for } X_1, X_2 \in W \text{ and } X_3 < Y \text{ for } X_3 \in W, Y \in L \rangle$$

for any $X \in W$. Two stochastic delays $[W_i]$ and $[W_j]$ can race each other and they can form a joint outcome if it is possible to consistently combine the winners and the losers such that the resulting outcome has disjoint winners and losers. Here, by consistently we mean that in the joint outcome no winner can come from the original sets of losers $L_1$ or $L_2$.

We take a closer look at the relation between the winners and the losers of the racing delays $[W_i]$ and $[W_j]$. There are three possible combinations that give the relation between the winners and the losers: (1) $L_1 \cap W_2 \neq \emptyset$, which means that the race must be won by $W_1$ and lost by $L_1 \cup W_2 \cup L_2$, (2) $W_1 \cap W_2 \neq \emptyset$, which means that the race must be won by $W_1 \cup W_2$ and lost by $L_1 \cup L_2$, and (3) $W_1 \cap L_2 \neq \emptyset$, which means that the race must be won $W_2$ and lost by $W_1 \cup L_1 \cup L_2$. Obviously, these ‘restrictions’ are disjoint and cannot be applied together as if more than one holds, then they lead to ill-defined outcomes. For example, if both (1) and (2) hold at the same time, then $L_1$ and $W_2$ must observe the same sample and also $W_1$ and $W_2$ must observe the same sample. Then $W_1$ and $L_1$ must observe the same sample, which is a contradiction.

To summarize, there are four possible joint outcomes of a race between $[W_i]$ and $[W_j]$: if (1) holds then the outcome is given by $[W_1 \cup W_2, \cup L_1, L_2]$, if (2) holds the outcome is given by $[W_1 \cup W_2]$, if (3) holds then the outcome is given by $[W_1 \cup W_1]$, and if none of the restrictions (1)–(3) hold, then all three (disjoint) outcomes are possible: $[W_1 \cup W_2, \cup L_1, L_2]$, $[W_1 \cup W_2]$, and $[W_1 \cup W_1]$. If at least two restrictions apply, then the outcomes cannot be combined as they represent disjoint events. In this case we say the race between the delays $[W_i]$ and $[W_j]$ with $W_1 \cup L_1 = W_2 \cup L_2$, is resolved.

To say that a race is resolved, we need that all the races stemming from the same race are resolved. We define a predicate $\text{rr}([W_i], [W_j])$ that checks whether two delays $[W_i]$ and $[W_j]$ are in a resolved race. It is satisfied if $W_1 \cup L_1 = W_2 \cup L_2$.
and at least two of the above three restrictions hold, i.e.,
\[
\text{rr}([W_1], [W_2]) \text{ if } W_1 \cup L_1 = W_2 \cup L_2 \text{ and } \\
((L_1 \cap W_2 \neq \emptyset \text{ and } W_1 \cap W_2 \neq \emptyset) \text{ or } \\
(L_1 \cap W_2 \neq \emptyset \text{ and } W_1 \cap L_2 \neq \emptyset) \text{ or } \\
(W_1 \cap W_2 \neq \emptyset \text{ and } W_1 \cap L_2 \neq \emptyset)).
\]
We proceed by introducing processes that are prefixed by stochastic delays.

2.3 Stochastic Delay Prefix
By \([X]\), \(p\) we denote a process term \(p\) prefixed by a stochastic delay \([X]\). This prefixed term denotes a process that behaves as \(p\) when \([X]\) expires. To express a race, we use the alternative composition \(- + \). So, \([X].p_1 + [Y].p_2\) represents two processes that are prefixed by the stochastic delays \(X\) and \(Y\) that are racing each other. As discussed above, there are three possible outcomes of this race in terms of the participating stochastic delays: (1) \([Y]\), (2) \([X, Y]\), and (3) \([X]\), i.e., the first stochastic delay expires before the second, they both expire together, or the second stochastic delay expires before the first. The passage of time of the stochastic delay \([Y]\) is guided by the conditional random variable \(X | X < Y\). In this case, the stochastic delay \(X\) expires, whereas \(Y\) becomes dependent on the amount of time that has passed for \(X\). Intuitively, this is represented by the term \([X].(p_1 + [Y].p_2)\), where both \(Y\)'s refer to the same stochastic delay, i.e., the second occurrence of \(Y\) is bound by the first one. Similarly, we have \([X].([X].p_1 + p_2)\), when the winner is \(Y\). In the case when both delays win, they expire together. By the notion of time determinism, which states that passage of time by itself cannot make a choice, the resulting term intuitively is \([X, Y].(p_1 + p_2)\).

The race is resolved when every possible outcome of the race is enumerated, i.e., no more outcomes are possible. Thus, we can also write \([Y].(p_1 + [Y].p_2) + [X, Y].(p_1 + p_2) + [X].([X].p_1 + p_2)\) instead of \([X].p_1 + [Y].p_2\) as both expressions give the same final outcomes of a race. The advantage of the first term is that it explicitly states all possible outcomes of the race and that these events are disjoint. Thus, we can clearly separate the stochastic behavior of the term depending on the resolved outcomes of the race. If an additional racing delay \(Z\) is added to the race, this also leads to the same outcomes, i.e., \(([X] + [Y]) + [Z]\) and \(([Y] + [X, Y] + [X]) + [Z]\) will yield the same racing behaviour. As an example, the outcome of \([Y] + [Z]\) is given by \([X, Y] + [Y, Z] + [X, Z]\). When considering complete races, i.e., race which have all possible outcomes, such an alternative composition is associative (cf. [11]). However, when considering incomplete races, e.g., the race induced by the term \([Y].p_1 + [X].p_2\), the alternative composition is no longer associative as discussed below in Section 4.3.

Next, we motivate the need and introduce an additional type of a race condition.

2.4 Dependent and Independent Race Condition
We give a motivation and illustrate the notions of a dependent and an independent race condition by a simple example. Consider the term \([X],p \| [X],p\),
where \( \parallel \) denotes the parallel composition. The semantics of the race condition in the parallel composition is the same as for the alternative composition. We can interpret the race between the two processes above in two ways: (1) from the standard viewpoint of Markovian/race condition semantics, the process is a composition of two independent components that are competing for the same resource and (2) from real-time perspective this composition synchronizes the two components that exhibit the same sample as they have the same name. The former interpretation is according to the independent (standard) race condition and it enables compositional modeling. It states that stochastic delays with the same name have the same distribution, but do not necessarily exhibit the same sample. This is the standard notion of a race condition and we refer to it as independent for the sake of consistency. The latter interpretation is according to the dependent race condition that forces racing delays with the same name to always exhibit the same duration and it supports the existence of expansion laws and it enables resolution of races. We provide for a better intuition by interpreting a simple race in both ways.

**Example 2.** The term \([X, p_1 + [X, p_2])\) should be equivalent to the term \([X, (p_1 + p_2)]\) if \(X\) is treated as a dependent stochastic delay. Both stochastic delays have a winner guided by \(X\), which exhibits the same sample in both terms and, therefore, the winners of both delays must exhibit passage of time together. On the other hand, if \(X\) is treated as an independent stochastic delay, then the same term is equivalent to \([Y, (p_1 + p_2)]\) for a random variable \(U\) satisfying \(F_U = F_X\). In the standard independent race condition interpretation, the two occurrences of \(X\) can exhibit different samples that are guided by the same distribution. Therefore, they actually represent separate stochastic delays and the second occurrence of \(X\) is renamed to a new stochastic delay \(U\) with the same distribution.

We introduce a dependence scope operator \(|p|_D\) for \(D \subseteq \mathcal{V}\) to specify dependent and independent delays that give rise to dependent and independent races, respectively. The racing delays in the races induced by the term \(p\) that are in \(D\) are treated as dependent. The names of dependent delays are important as they identify stochastic delays that exhibit the same sample. On the contrary, the names of the independent delays play no role except for identifying stochastic delays with the same distribution. In the previous example, \(|[Y, p_2]|_X\) would denote that \(X\) is a dependent stochastic delay, but \(Y\) is an independent one. Intuitively, this term is equivalent to \(|[X, p_2]|_X\), for every \(Z\) such that \(F_Z = F_Y\), but it is not equivalent to \(|[Y, p_2]|_U\), for any \(U \neq X\) even if \(F_U = F_X\). Multiple scopes intersect and, e.g., \(|[Y, p_2]|_Y|_X\) denotes a process prefixed by the independent delay \(|[Y, p_2]|_X\) because \(\{X\} \cap \{Y\} = \emptyset\).

The dependence scope plays an important role in giving operational semantics to the terms. As a reminder, the stochastic delay prefix \([Y, p]\) denotes an outcome of a race between the stochastic delays in \(W \cup L\), where the winners are given by \(W\) and the losers are given by \(L\). Moreover, it denotes that there was passage of time for the losing delays in \(L\) that continue to persist in \(p\). This
means that the losers do not have their original distribution in the resulting process \( p \) and that their distributions must be ‘aged’ by the duration of the sample exhibited by the winners \( W \). Therefore, the names of the losing delays must be protected in \( p \), i.e., the become dependent. This is achieved by writing \( |p|_L \) as a remaining term after the expiration of the delay given by \([W]_L\). Thus, \([W]_L \cdot p\) is actually equivalent to \([W]_L \cdot |p|_L\) as only the names in \( L \) must be preserved in \( p \). This also means that the stochastic delays that are not in \( L \) become independent. To support the meaning of process terms as discussed above, the stochastic delays that are not encompassed by any dependence scope are considered as dependent. Thus, \([W]_L \cdot p\) is actually equivalent to \([X, Y]_L \cdot p\). We illustrate the above discussion by an example. The first occurrences of \( X \) and \( Y \) in the term \([X, Y]_L \cdot p\), denote dependent stochastic delays \([X]_L \) and \([Y]_L \). However, the second occurrence of \( X \) in the subterm \([X, Y]_L \cdot p\), which by the discussion above is equivalent to \([X, Y]_L \cdot p\), denotes an independent stochastic delay, whereas the second occurrence of \( Y \) in the same subterm refers to the losing dependent delay \([Y]_L \) from the first race.

2.5 Timed Delays in a Racing Context

Before introducing timed delays in the process theory, we give a simple example of an expiration of a stochastic delay over a period of time.

Example 3. Suppose that \( X \) is a random variable such that \( P(X = 1) = \frac{1}{2}, \ P(X = 2) = \frac{1}{3}, \) and \( P(X = 3) = \frac{1}{6} \). We observe what happens to the stochastic delay \([X]_L \) after 1 unit of time. Then, either the stochastic delay expires with probability \( \frac{1}{2} \) or it is aged by 1 time unit. In the latter case it allows a passage of time as the random variable \( X' = \langle X \mid X > 1 \rangle \), where \( P(X' = 1) = \frac{2}{3} \) and \( P(X' = 2) = \frac{1}{3} \). Now, we observe what happens to the delay \([X']_L \) after one unit of time. The delay \([X']_L \) can expire with probability that \([X]_L \) did not expire in the first time unit multiplied by the probability that \( X' = 1 \), i.e., \( P(X > 1) \cdot P(X' = 1) = \frac{1}{2} \cdot \frac{2}{3} = \frac{1}{3} \) (\( = P(X = 2) \)). However, it can also delay more than one time unit and become aged by 1. Then, it allows passage of time according to \( X'' = \langle X' \mid X' > 1 \rangle \) (\( = \langle X \mid X > 2 \rangle \)), with \( P(X'' = 1) = \frac{1}{2} \). Obviously, \([X'']_L \) must expire in one time unit with probability that both \([X] \) and \([X']_L \) did not expire in one time unit, i.e., \( P(X > 1) \cdot P(X' > 1) \cdot P(X'' = 1) = \frac{1}{2} \cdot \frac{1}{3} \cdot \frac{1}{3} = \frac{1}{6} \) (\( = P(X = 3) \)).

Although being a simple exercise in probability, Example 3 illustrates how to handle an expiration of a stochastic delay per unit of time. First, we formalize the notion of an aging of a distribution, which gives the right shift of a distribution over passage of time.

Definition 4. A distribution function \( F \) can be aged by \( m \in \mathbb{N} \) if \( F(m) < 1 \). The resulting distribution \( F|m \) is given by

\[
(F|m)(n) = \frac{F(n + m) - F(m)}{1 - F(m)} ,
\]
If the condition of Definition 4 is fulfilled, then $F|_m$ is again a probability distribution function. Because we work with probability distributions satisfying $F(0) = 0$, we have that $F|0 = F$. Moreover, iterative application of the aging function is the same as aging the function once by the accumulative time duration as illustrated by Example 3 [19], i.e.,

$$(\ldots(F|m_1)\ldots)|_{m_k} = F\left(\sum_{i=1}^{k} m_i\right).$$

As a direct consequence, to compute a total age of a distribution of a stochastic delay it suffices only to compute the sum of the duration of the samples of every race that it lost.

Now, let us denote by $\sigma_0^X$ the event that the delay $[X]$ expires after one time unit has passed, i.e., in race condition terminology the stochastic delay $[X]$ wins a race with a sample of one unit timed delay and there are no losers. Let us assume that the age of $X$ is $m$ and let us denote by $X|m = (X | X > m)$ the conditional random variable with distribution $F_X|m$. Then, the probability of the event $\sigma_0^X$ is $P((X|m) = 1)$, i.e., the probability that $[X]$ expired after $m + 1$ unit of time. By $\sigma_k^X$, we denote the event that the delay $[X]$ does not expire in one time unit, i.e., the stochastic delay $[X]$ loses the race to a unit timed delay and there are no additional winners. Again, by assuming that $X$ has an age $m$, the probability of this event is $P((X|m) > 1)$, and after the expiration of the timed delay, the age of $X$ becomes $m + 1$. Thus, at each point in time we have two possibilities: either the delay expires, or the delay does not expire and it is aged by one time unit. Then, the process $[X],p$ can be specified as as the solution of the recursive equation

$$A = \sigma_0^X,p + \sigma_0^X,A,$$

for the recursive variable $A$.

In a generalized context, by the same reasoning we specify a stochastic delay $[W]L,p$ as the solution of the recursive equation for $B$:

$$B = \sigma_W^L,p + \sigma_W^L,B.$$

We will refer to $\sigma_W^L$ as a unit timed delay prefix in a racing context of the race induced by the winner $W$ and the losers $L$, or simply timed delay prefix for short. The probability of this event is denoted by

$$RC_1(W, L) = P(W = 1, L > 1),$$

where the racing delays in $W \cup L$ can have their own ages as in the discussion for a race with a single delay $[X]$ above.

We emphasize that timed delays are not stochastic delays that impose a race condition and give joint outcomes by resolving them, but they allow passage of a unit of time in an presupposed racing context. In our setting we build a process theory for timed delays in a racing context and retrieve stochastic delays via guarded recursive specifications as indicated above. The standard unit timed
delay prefix is embedded in the theory as \( \sigma^\emptyset \), i.e., a timed delay in an empty racing context. By convention we put \( RC_1(\emptyset, \emptyset) = 1 \). We omit the empty sets in the notation when clear from the context and we also write \( \sigma^n \) for \( n \geq 1 \) subsequent timed delays prefixes \( \sigma \).

Timed delays can also be in a context of resolved races. If \( rr([W_1], [L_1]) \) holds, then \( \sigma^{W_1}_L \) and \( \sigma^{L_2}_W \) are in the context of the resolved race between \([W_1]_L\) and \([L_2]_W\).

However, this does not cover the case when there are no winners in the racing context, i.e., no stochastic delays expire after one unit of time. For that purpose we overload the resolved race predicate \( rr(\_\_\_\_) \) to \( rr(\sigma^{W_1}_L, \sigma^{L_2}_W) \) as follows:

\[
rr(\sigma^{W_1}_L, \sigma^{L_2}_W) \text{ if } W_1 \cup L_1 = W_2 \cup L_2 \text{ and }
\begin{align*}
&((L_1 \cap W_2 \neq \emptyset \text{ and } W_1 \cap W_2 \neq \emptyset) \text{ or } \newline
&(L_1 \cap W_2 \neq \emptyset \text{ and } W_1 \cap L_2 \neq \emptyset) \text{ or } \newline
&(W_1 \cap W_2 \neq \emptyset \text{ and } W_1 \cap L_2 \neq \emptyset) \text{ or } \newline
&(W_1 = \emptyset \text{ and } W_2 \cap L_1 \neq \emptyset) \text{ or } \newline
&(W_2 = \emptyset \text{ and } W_1 \cap L_2 \neq \emptyset))
\end{align*}
\]

As a reminder, the predicate \( rr(\_\_\_\_) \) defined the context in which the race between the stochastic delays \([W_1]_L\) and \([L_2]_W\) is resolved. The extra conditions deal with the overloaded situation for the timed delays \([W_1]_L\) and \([L_2]_W\) where in the context of one timed delay no racing delay has yet expired, whereas in the context of the other the winners have expired, creating a disjoint event.

As stochastic delays can form inconsistent races, timed delays can also have inconsistent racing contexts. However, unlike the stochastic delays, the context of the timed delay is static, i.e., the racing condition is not resolved, but only endorsed. We illustrate the situation by an example.

**Example 5.** The process \( \sigma^X.p_1 + \sigma^X.p_2 \) can only deadlock. The process \( \sigma^X.p_1 \) performs a unit of time after which \([X]\) expires. The process \( \sigma^X.p_2 \) performs a unit of time after which \([Y]\) expires in a context of a race in which \([Y]\) won over \([X]\). Thus, the process allows \([X]\) to expire in one timed unit, but it also allows for \([Y]\) to expire in one time unit. However, \([Y]\) should delay less than \([X]\) as implied by the racing context of \( \sigma^X \), which leads to an inconsistency as there is no information about \([Y]\) in context of the first timed delay.

Example 5 also illustrates the main difference between stochastic delays and timed delays in a racing context as \([X].p_1 + [X].p_2\) is equivalent to \([X].([X].p_1 + p_2)\), after the resolution of the race between \([X]\) and \([X]\). This type of dynamics is enabled for the timed delays by using the unfolding of the guarded recursive specifications that models the stochastic delays (see Section 5.2 below).

### 2.6 Design Choices

We model processes using probabilistic timed automata that have probabilistic timed transitions systems as an underlying model. Processes have outgoing timed delay transitions and immediate action transitions that do not allow any passage of time. The choice between several action transitions is nondeterministic and, in
general, depends on the environment as in standard process algebras. The choice between timed delays is probabilistic as it is induced by the racing context of the delays. We favor time-determinism, i.e., the principle that passage of time alone cannot make a choice [20, 21]. The probabilistic choices only resolve the race condition, but do not resolve the choice in the alternative composition. Also, we adopt the weak choice between immediate actions and passage of time, i.e., we impose a nondeterministic choice on the immediate action transitions and the passage of time in the vein of ACP-styled timed process algebras [20, 21]. To support maximal progress, i.e., to prefer immediate action to passage of time, we include a maximal progress operator in the theory together with encapsulation of actions, thereby disabling unwanted action transitions. We also opt for guarded recursion introduced by means of guarded recursive specifications. We derive delayable actions as solutions of guarded recursive equations that can perform an immediate action at any point in time. Stochastic delays are also introduced in the theory using guarded recursive specifications as briefly discussed above. We believe this approach to be systematic as it builds on well-established notions. Moreover, it helps to identify the set of primitive operators that can be combined to bring the other more complex features into the theory.

In the next section, we introduce the signature of the theory and we give semantics to the process terms using a type of probabilistic timed automata we refer to as racing timed transition schemes.

3 Process Theory TCP

In this section we begin with the introduction to TCP

\[ TCP_{\text{drst}}(\mathcal{A}, \mathcal{V}, \mathcal{R}, \gamma) \]

the theory of communicating processes with discrete real and stochastic time, where \( \mathcal{A} \) denotes the set of actions, \( \mathcal{V} \) denotes the set of random variables, \( \mathcal{R} \) denotes the set of recursive variables, and \( \gamma \) is the commutative and associative action synchronization function. First, we analyze the nonrecursive part of the theory denoted by TCP

\[ TCP_{\text{rec}}(\mathcal{A}, \mathcal{V}, \gamma) \]

We introduce guarded recursion later in Section 4.10 by means of guarded recursive specifications. We give operational semantics to process terms using racing timed transition schemes. We give a strong bisimulation relation and show that it is congruence for the given operators. Afterwards, we use it to define a term model for the theory.

3.1 Racing Timed Transition Schemes

In essence, racing timed transition schemes are probabilistic timed automata in which the probabilistic choice is implicitly (symbolically) stated by the racing context of the timed delays. The states determine the timed transitions, whereas we use an additional construct, called an environment, to keep track of the ages of the racing delays. It is denoted by a function \( \alpha \) that holds the age of the distribution function of each racing stochastic delay. We put \( \alpha : \mathcal{V} \rightarrow \mathbb{N} \) and we write \( \mathcal{E} \) for the set of all such environments. We recall that age 0 actually means that the stochastic delay has no age, i.e., it did not lose any race until that point. The independent racing delays are identified in each state by the function \( I(\_\_\_ ) \).
Definition 6. A racing timed transition scheme \((S \times \mathcal{E}, A, V, \rightarrow, \mapsto, \downarrow, I)\) is a tuple, where \(u = \langle s, \alpha \rangle \in S \times \mathcal{E}\) represents a state \(s\) in an environment \(\alpha\), \(A\) is a set of actions, \(V\) is a set of random variables giving the stochastic delay names, and

- \(\rightarrow \subseteq (S \times \mathcal{E}) \times A \times (S \times \mathcal{E})\) is the action transition relation;
- \(\mapsto \subseteq (S \times \mathcal{E}) \times 2^V \times 2^V \times (S \times \mathcal{E})\) is a timed delay transition relation. For every timed delay transition \(u \xrightarrow{W_{1}} u_1 \neq u \xrightarrow{W_{2}} u_2\) the predicate \(\text{rr}(\sigma_{W_{1}}, \sigma_{W_{2}})\) is satisfied.
- \(\downarrow \subseteq S \times \mathcal{E}\) is the immediate termination predicate;
- \(I: S \rightarrow 2^V\) is the independent racing delays function. It satisfies that \(I(s) \subseteq \bigcup_{(s, \alpha) \xrightarrow{W_{1}} (s', \alpha')} (W \cup L)\), for every \(\alpha \in \mathcal{E}\).

Definition 6 requires that the predicate \(\text{rr}(\sigma_{W_{1}}, \sigma_{W_{2}})\) holds for every two different timed delay transitions \(u \xrightarrow{W_{1}} u_1 \neq u \xrightarrow{W_{2}} u_2\) originating from the same state \(u\). This implies that \(W_{1} \cup L_{1} = W_{2} \cup L_{2}\). Thus, for every state \(s\) there exists a set of racing delays \(R(s)\) satisfying \(R(s) = W \cup L\) for every \((s, \alpha) \xrightarrow{W} (s', \alpha')\). Then, for the independent racing delays it holds that \(I(s) \subseteq R(s)\) and the set of dependent racing delays is given by \(D(s) = R(s) \setminus I(s)\).

3.2 Probabilistic Timed Transition Systems

A probabilistic timed transition system represents an instantiation of a transition scheme with respect to a given assignment \(d: V \rightarrow \mathcal{F}\) of the probability distributions. The race condition is used to derive the underlying probability spaces that define the probabilistic behavior of each timed delay transition. In order to compute the correct distributions of the racing delays we will use the environment and the aging function. More precisely, the distribution of a racing delay \([X]\) in an environment \(\alpha\) is given by \(F_X = d(X)|_{\alpha(X)}\).

Definition 7. A probabilistic timed transition system \((S, A, d, \rightarrow, \mapsto, \downarrow)\) is a tuple, where \(S\) is the set of states, \(A\) is a set of labels, \(d: V \rightarrow \mathcal{F}\) assigns the distributions to the random variables, and

- \(\rightarrow \subseteq S \times A \times \mathcal{E}\) is the action transition relation;
- \(\mapsto: S \rightarrow \mathcal{P}(\mathbb{N} \times S)\) is the probabilistic timed transition function;
- \(\downarrow \subseteq S\) is the immediate termination predicate.

Each racing timed transition scheme coupled with an assignment of probability distributions to the stochastic delays induces a probabilistic timed transition system. The action transitions and the termination predicate are adopted from the racing timed transition scheme. The probability measure of the (unit) timed delay is induced by its racing context. The formal definition is as follows.
Definition 8. Let \( R = (S \times \mathcal{E}, A, V, \rightarrow, \rightarrow, \downarrow, I) \) be a racing timed transition scheme and \( d: V \rightarrow F \) a distribution assignment function. Then, \( (R, d) \) induces the probabilistic timed transition system \( P = (S \times \mathcal{E}, A, d, \rightarrow, \rightarrow, \downarrow) \), where the action transition and termination options \( \rightarrow \) and \( \downarrow \) of \( P \) are given by \( \rightarrow \) and \( \downarrow \) of \( R \), respectively, and \( \rightarrow(u) = ((1, S \times \mathcal{E}), P) \) for \( u = (s, \alpha) \) is the probability space induced by the race condition. The probability measure \( P \) is given by

\[
P(1, u') = \frac{\text{RC}_1(W', L')}{{\sum}_{u'' \rightarrow u} \text{RC}_1(W, L)} \quad \text{if} \quad R(s) = W' \cup L' \neq \emptyset,
\]

or \( P(1, u') = 1 \) otherwise, for \( u \rightarrow u' \) and the distribution functions of \( X \in R(u) \) are given by \( F_X = d(X)|\alpha(X) \).

We remind the reader that \( W' \cup L' = W \cup L \) for every timed delay transition \( u \rightarrow u' \) of \( u \). The probability measure is normalized because the race need not be complete, i.e., \( \sum_{u'' \rightarrow u} \text{RC}_1(W, L) \leq 1 \). Only if the race is complete, i.e., all possible outcomes are stated by the timed delay transitions, the sum above equals one.

3.3 Bisimulation Relation

We define a strong bisimulation relation on racing timed transition schemes. It requires timed delays to be in the same racing context modulo names of independent delays. This ensures that the related racing timed transition schemes have the same probabilistic behavior, i.e., they induce the same probabilistic timed transition systems when coupled with corresponding distribution assigning functions. As usual, bisimilar terms are required to have the same termination options and action transitions [20, 21].

Definition 9. Let \( R \subseteq (S \times \mathcal{E})^2 \times (V \leftrightarrow V) \) be a symmetric relation. Then \( R \) is a racing timed bisimulation if for all \( ((s_1, \alpha_1), (s_2, \alpha_2), r) \in R \) it holds that \( r: R(s_1) \leftrightarrow R(s_2) \) is a bijection with \( r(I(s_1)) = I(s_2) \), and \( F_X = F_{r(X)} \) and \( \alpha_1(X) = \alpha_2(r(X)) \) for \( X \in \text{dom}(r) \), and:

1. if \( u_1 \downarrow \) then \( u_2 \downarrow \);
2. if \( u_1 \xrightarrow{a} u'_1 \) for some \( u'_1 \in S \times \mathcal{H} \), then \( u_2 \xrightarrow{a} u'_2 \) for some \( u'_2 \in S \times \mathcal{H} \) such that \( (u'_1, u'_2, r') \in R \) for some \( r' \in V \leftrightarrow V \); and
3. if \( u_1 \xrightarrow{w_{i_1}} u'_1 \) for some \( u'_1 = (s'_1, \alpha'_1) \in S \times \mathcal{E} \), then \( u_2 \xrightarrow{w_{i_2}} u'_2 \) for some \( u'_2 \in S \times \mathcal{E} \) where \( r(W_1) = W_2 \), \( r(L_1) = L_2 \), and \( (u'_1, u'_2, r') \in R \) for some \( r' \in V \leftrightarrow V \) satisfying \( r'(X) = r(X) \) for \( X \in L_1 \cap D(s'_1) \).

We say that two states \( u_1 \) and \( u_2 \) are racing timed bisimilar, notation \( u_1 \equiv_r u_2 \), if there exists a bisimulation relation \( R \) such that \( (u_1, u_2, r) \in R \) for some \( r \in V \leftrightarrow V \).
The relationship between racing contexts of timed delays of bisimilar states is established using the bijection $r$. It is a bijection as the same number of racing delays must be present in both states. It also must respect the independent delays stated by $r(I(s_1)) = I(s_2)$. The independent delays can have different names, but they must have the same distribution and the same age, meaning that they will exhibit the same probabilistic behavior. Conditions 1 and 2 state that bisimilar states have the same termination options and action transitions. Timed delays are performed by winners and losers related by $r$. Condition 3 requires that the losers are backward compatible, i.e., they retain their name as it is bound by the first race that they lost.

As a prerequisite to being a congruence in $\text{TCP}^{\text{dest}}$, bisimilarity should be an equivalence relation as stated in the following theorem.

**Theorem 10.** Bisimilarity is an equivalence relation.

**Proof.** It should be clear that $\equiv_i$ is a reflexive relation, i.e., $u \equiv_i u$, by putting $R = \{(u, u, \text{id}_{R(u)}) \mid u \in S \times \mathcal{E}\}$.

For symmetry, assume that $u \equiv_i v$. Then there exists a bisimulation $R$ such that $(u, v, r) \in R$, for some bijection $r$ satisfying the conditions of Definition 9. Put $R' = \{(v, u, r^{-1}) \mid (u, v, r) \in R\}$. Clearly $r^{-1}$ satisfies the conditions of Definition 9 and $R'$ is a stochastic bisimulation.

For transitivity, assume that $u_1 \equiv_i u_2 \equiv_i u_3$, i.e., there exist two bisimulation relations $R_1$ and $R_2$ such that $(u_1, u_2, r_1) \in R_1$ and $(u_2, u_3, r_2) \in R_2$. Define $R_3$ as the composition $R_3 = R_2 \circ R_1$, where $r_3 = r_2 \circ r_1$ is again a bijection satisfying the conditions of Definition 9. It is not difficult to see that $R_3$ is a bisimulation, which completes the proof. $\Box$

Next, we introduce the process theory and we give semantics to the process terms using racing timed transitions schemes.

### 3.4 Signature

We informally introduce the operators before giving a formal definition of the language. The deadlocked process that does not have any outgoing transitions is denoted by $\delta$; successful termination by $\epsilon$. Undelayable action prefixing is a unary operator scheme $\underline{a}$, for every $a \in \mathcal{A}$. Similarly, timed delay prefixes are of the form $\sigma_W^L$, for $W, L \subseteq V$ disjoint. The dependence scope operator scheme is given by $|.-|_D$, for a dependence binding set $D \subseteq V$. The encapsulation operator scheme $\partial_H(\cdot)$ for $H \subseteq \mathcal{A}$ suppresses the actions in $H$. The maximal time progress operator scheme $\theta_H(\cdot)$ for $H \subseteq \mathcal{A}$ gives priority to the undelayable actions in $H \subseteq \mathcal{A}$ over passage of time. The alternative composition is given by $\cdot + \cdot$, at the same time representing a nondeterministic choice between action transitions and termination, a weak nondeterministic choice between action and timed delay transitions, and probabilistically resolving the racing context for the timed delay transitions. The parallel composition is given by $\cdot \| \cdot$. It allows passage of time only if both components do so.
Definition 11. The signature of $\text{TCP}^{\text{drst}}$ is given by

$$P := \delta \mid \epsilon \mid a.P \mid \sigma^W_P \mid |P|_D \mid \partial_H(P) \mid \theta_H(P) \mid P + P \mid P \parallel P,$$

where $a \in A$, $W, L, D \subseteq V$ with $W \cap L = \emptyset$, and $H \subseteq A$. The set of closed terms that do not contain term variables is denoted by $\mathcal{C}(\text{TCP}^{\text{drst}})$ and it is ranged over by $p$ and $q$.

Next we take a closer look at the races induced by the timed delay prefixes.

3.5 Auxiliary Operations

The general idea of having both dependent and independent delays available is the following: For specification one can use multiple instances of a component comprising independent delays. As the delays are independent, there is no need to worry about the actual samples. However, outgoing timed delay transitions from the states of the racing timed transition schemes have racing delays with unique names (as there the races are resolved). So, process terms may exhibit naming conflicts. For example, the term $p = |\sigma^X.q|_\emptyset \parallel |\sigma^Y.q|_\emptyset$ expresses a race between two components guided by independent delays with the same name. However, the timed delay transitions of $\langle p, \alpha \rangle$ comprise two racing delays with unique different names, but equal distributions. For $p$ to have proper semantics, the conflicting independent delay names have to be detected and renamed, e.g., to $|\sigma^Y.q|_\emptyset \parallel |\sigma^X.q|_\emptyset$ where $F_X = F_Y$. To detect the conflicting racing delay names, we use auxiliary operations $D(p)$ and $I(p)$ to extract the dependent racing delays and the independent racing delay names of the term $p$, respectively. We say independent delay names instead of independent delays since there might not be one-to-one correspondence between the two in the process terms, e.g., in $p$ from above. Having the dependent racing delays and the independent racing delay names, the set of racing delay names is given by $R(p) = D(p) \cup I(p)$.

One more type of naming conflicts arises when a loser and some new independent delay, which became enabled due to an expiration of a winner, have the same name. For example, such situation is given by the term $\sigma^X.\sigma^Y.\delta + \sigma^Y.\delta$. If the winner of the race between $[X]$ and $[Y]$ is $[X]$, then the resulting term is $|\sigma^Y.\delta|_\emptyset + |\sigma^Y.\delta|_\emptyset$. It has two racing delays with the name $Y$ that do not represent the same racing delay, because the one on the right has age of at least 1, whereas the one on the left is independent (as $[X]$ has no losers it does not induce any dependence) and it has no age at all. To detect this type of naming conflicts, the set of newly enabled independent delay names $N(p)$ of a term $p$ is extracted.

We will use $\alpha$-conversion to enable dynamic renaming that resolves local naming conflicts in the vein of [22]. Intuitively, $\alpha$-conversion enables renaming of independent delay names without distorting the structure of the term, conforming to the bisimulation relation. Its definition requires renaming of racing delay names, including the ones that are in the dependence set $D$ of the dependence scope operator $|.|_D$. We refer to these delay names as the dependence binding delay names and we denote them by $B(p)$. 

15
The definitions of the auxiliary operations are given in Table 1. The dependent racing delays D(p) of the process term p are calculated as: (1) the racing delays in the context of the timed delays connected by the outermost composition that are not in any scope and (2) as the ones that are in the intersection of the dependence sets of all encompassing dependence scope operators. The independent racing delay names cannot be calculated directly, as we need to keep track of the intersection of the dependence scopes. For that purpose we extend I(p) with an auxiliary set D and obtain I(p, D). Now, the set of independent racing delay names can be computed as the set of dependent racing delays of p without the ones in D. Initially, we put D = ∅ as by default all racing delay names are treated as dependent, i.e., I(p) = I(p, ∅). The newly enabled independent delay names N(p) are the independent delay names that are introduced in the race because of an expiration of a winner. Note that the losers of the prefixing timed delay are the only dependent delays in the resulting term. The dependence binding delay names B(p) are extracted as the names in the dependence sets of the scope operators encompassing racing delays of the topmost race.

<table>
<thead>
<tr>
<th>D(ε) = D(δ) = D(∅, p) = ∅,</th>
<th>D(p ∩ D) = D(p) \ D,</th>
<th>D(σ_p^e, p) = W \cup L,</th>
</tr>
</thead>
<tbody>
<tr>
<td>D(θ_H(p)) = D(p),</td>
<td>D(p_1 + p_2) = D(p_1 \parallel p_2) = D(p_1) \cup D(p_2)</td>
<td></td>
</tr>
<tr>
<td>I(ε, D) = I(δ, D) = I(∅, p, D) = ∅,</td>
<td>I(θ_H(p), D) = I(θ_H(p), D) = I(p, D)</td>
<td></td>
</tr>
<tr>
<td>I(p_1 + p_2, D) = I(p_1 \parallel p_2, D) = I(p_1, D) \cup I(p_2, D)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I(p, D \cap D') = I(p, D) \cap D',</td>
<td>I(σ_p^e, p, D) = (W \cup L) \setminus D</td>
<td></td>
</tr>
<tr>
<td>N(ε) = N(δ) = N(∅, p) = ∅,</td>
<td>N(</td>
<td>p</td>
</tr>
<tr>
<td>N(</td>
<td>p</td>
<td>_D = I(</td>
</tr>
<tr>
<td>B(ε) = B(δ) = B(∅, p) = B(σ_p^e, p) = ∅,</td>
<td>B(</td>
<td>p</td>
</tr>
<tr>
<td>B(θ_H(p)) = B(θ_H(p)) = B(p),</td>
<td>B(p_1 + p_2) = B(p_1 \parallel p_2) = B(p_1) \cup B(p_2)</td>
<td></td>
</tr>
</tbody>
</table>

**Table 1.** Auxiliary operations

We illustrate the situation by a simple example.

**Example 12.** Let $p = |||σ_{X,Y}^x \sigma_{X,Y}^x \delta_{X,Y}|_{X,Y}|_{X,Y,Z}$. Then (1) D(p) = \{X\} and (2) I(p) = I(p, V) = \{Y, Z\} because V \cap \{X, Z\} \cap \{X, Y\} \cap \{X, Y, Z\} = \{X\}. (3) N(p) = I(σ_{X,Y}^x \delta_{X,Y}|_{X,Y,Z} = \{X\}$, and (4) B(p) = \{X, Z\} \cup \{X, Y\} \cup \{X, Y, Z\} = \{X, Y, Z\}.

**Remark 13.** We note that in case there is a maximal progress operator in the term, then it may happen that not all timed delay transitions are actually taken because of prioritization of undelayable actions. Hence, the auxiliary operators may actually result in more stochastic delay names than actually observed in the racing contexts of the timed delay transitions. To model this behavior punctually, the operators have to become more complicated in order to examine the behavior.
of the maximal progress. However, this does not contribute in any sense to the semantics and the only side effect is that the \( \alpha \)-conversion and the checkouts for naming conflicts defined below cater for more delays in some cases. For that reason and for the sake of clarity and compactness we leave these redundant stochastic delay names in place.

We proceed by identifying the naming conflicts that may lead to inconsistent probabilistic behavior as discussed above.

### 3.6 Naming Conflicts

When an independent and a dependent delay or multiple independent delays have the same name, naming conflicts arise that influence the probabilistic behavior of the race. Moreover, naming conflicts arise in the environment when a loser with an age and a newly enabled independent delay have the same name. In principle, all naming conflicts in closed terms can be statically resolved by giving unique names to independent delays [13]. In the current setting, however, we adopt a dynamic approach by using \( \alpha \)-conversion in the vein of [22] to support renaming for guarded recursion as well, which cannot easily be handled statically. The set of conflicting names \( C(p) \) of a term \( p \in C(\text{TCP}^{\text{drst}}) \) is given in Table 2.

<table>
<thead>
<tr>
<th>Term</th>
<th>Conflicting Names</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \epsilon )</td>
<td>( \emptyset )</td>
</tr>
<tr>
<td>( \delta )</td>
<td>( \emptyset )</td>
</tr>
<tr>
<td>( \theta_{L}(p) )</td>
<td>( C(p) )</td>
</tr>
<tr>
<td>( \theta_{H}(p) )</td>
<td>( C(p) )</td>
</tr>
</tbody>
</table>
| \( p_1 + p_2 \) | \( (I(p_1) \cup N(p_1)) \cap R(p_2)) \cup (R(p_1) \cap (I(p_2) \cup N(p_2))) \cup C(p_1) \cup C(p_2) \).

| Table 2. Set of conflicting names |

Conflicts arise when the set of losers and the set of newly enabled independent delays have a common name as given by \( C([\frac{\epsilon}{2}], p) \). Also, compositions can introduce conflicting names as independent or newly enabled independent delay names of one component can overlap with the racing delay names of the other one. Here, the search for conflicting names must continue in the components as well, as they also might comprise alternative or parallel compositions.

In case naming conflicts arise, we resolve them using \( \alpha \)-conversion as discussed in Section 3.8. For the time being, we give the operational semantics for process terms without naming conflicts. In case naming conflicts arise, the process term can only deadlock.

### 3.7 Structural Operational Semantics

The semantics of a term \( p \in C(\text{TCP}^{\text{drst}}) \) in an environment \( \alpha \in \mathcal{E} \) is given by the racing timed transition scheme \( (C(\text{TCP}^{\text{drst}}) \times \mathcal{E}, \mathcal{A}, \mathcal{V}, \longrightarrow, \rightarrow, \downarrow) \), where \( \rightarrow \).
action transitions labeled by the action term does not exhibit naming conflicts. The resulting environment contains the prefixes enable action transitions and reset the ages of the racing delays to the constant terminates independent of the environment. Rule 2 states that action operators, and the alternative composition. Rule 1 states that the termination alternative composition operator

<table>
<thead>
<tr>
<th>Table 3. Operational rules for the termination constant, the prefix operators, and the alternative composition operator</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1) $\llbracket \epsilon, \alpha \rrbracket$, $\llbracket p, \alpha \rrbracket$ $\xrightarrow{a} \llbracket p, \alpha \rrbracket$</td>
</tr>
<tr>
<td>(2) $\llbracket p, \alpha \rrbracket$ $\xrightarrow{\sigma_{L}^{W}, p} \llbracket p, \alpha \rrbracket$</td>
</tr>
<tr>
<td>(3) $\llbracket p, \alpha \rrbracket$ $\xrightarrow{\nu_{L}, \alpha} \llbracket \sigma_{L}^{W}, p \rrbracket$</td>
</tr>
<tr>
<td>(4) $\llbracket p, \alpha \rrbracket$ $\xrightarrow{\nu_{L}, \alpha} \llbracket \sigma_{L}^{W}, p \rrbracket$</td>
</tr>
<tr>
<td>(5) $\llbracket p, \alpha \rrbracket$ $\xrightarrow{\nu_{L}, \alpha} \llbracket \sigma_{L}^{W}, p \rrbracket$</td>
</tr>
<tr>
<td>(6) $\llbracket p, \alpha \rrbracket$ $\xrightarrow{\nu_{L}, \alpha} \llbracket \sigma_{L}^{W}, p \rrbracket$</td>
</tr>
<tr>
<td>(7) $\llbracket p, \alpha \rrbracket$ $\xrightarrow{\nu_{L}, \alpha} \llbracket \sigma_{L}^{W}, p \rrbracket$</td>
</tr>
<tr>
<td>(8) $\llbracket p, \alpha \rrbracket$ $\xrightarrow{\nu_{L}, \alpha} \llbracket \sigma_{L}^{W}, p \rrbracket$</td>
</tr>
<tr>
<td>(9) $\llbracket p, \alpha \rrbracket$ $\xrightarrow{\nu_{L}, \alpha} \llbracket \sigma_{L}^{W}, p \rrbracket$</td>
</tr>
<tr>
<td>(10) $\llbracket p, \alpha \rrbracket$ $\xrightarrow{\nu_{L}, \alpha} \llbracket \sigma_{L}^{W}, p \rrbracket$</td>
</tr>
<tr>
<td>(11) $\llbracket p, \alpha \rrbracket$ $\xrightarrow{\nu_{L}, \alpha} \llbracket \sigma_{L}^{W}, p \rrbracket$</td>
</tr>
<tr>
<td>(12) $\llbracket p, \alpha \rrbracket$ $\xrightarrow{\nu_{L}, \alpha} \llbracket \sigma_{L}^{W}, p \rrbracket$</td>
</tr>
<tr>
<td>(13) $\llbracket p, \alpha \rrbracket$ $\xrightarrow{\nu_{L}, \alpha} \llbracket \sigma_{L}^{W}, p \rrbracket$</td>
</tr>
<tr>
<td>(14) $\llbracket p, \alpha \rrbracket$ $\xrightarrow{\nu_{L}, \alpha} \llbracket \sigma_{L}^{W}, p \rrbracket$</td>
</tr>
<tr>
<td>(15) $\llbracket p, \alpha \rrbracket$ $\xrightarrow{\nu_{L}, \alpha} \llbracket \sigma_{L}^{W}, p \rrbracket$</td>
</tr>
</tbody>
</table>

Table 3 gives the operational rules for the termination constant, the prefix operators, and the alternative composition operator. Rule 1 states that the termination constant terminates independent of the environment. Rule 2 states that action prefixes enable action transitions and reset the ages of the racing delays to the zero environment. Rule 3 states that timed delay prefixes enable timed transitions with racing contexts induced by the winners and the losers provided the term does not exhibit naming conflicts. The resulting environment contains the
Table 4. Operational rules for the parallel composition, the encapsulation, and the maximal progress operator
Table 4 gives the operational rules for the alternative composition, the encapsulation, and the maximal progress operator. Rule 16 states that the parallel composition can terminate only when both components can. Rules 17–20 enable interleaving of action transitions in the parallel composition. Rules 17 and 18 state that the environment is reset when the other component cannot perform a timed delay transition. This is to preserve the desired property that only the ages of the losers persist in the environment. However, the environment must be preserved in case the other component can perform a timed delay as given by rules 19 and 20. Rule 21 allows for synchronization of action transitions if defined by the synchronization function. Similarly to the alternative composition, synchronization of timed delays is allowed when the racing contexts can be merged as given by rule 22 provided that there are no naming conflicts. Rule 23 states that the termination option is not affected by the encapsulation operator. Rule 24 states that action transitions are allowed only if they are not labeled by actions that should be suppressed. Rule 25 states that the encapsulation does not affect the timed delays. Rules 26 and 27 state that the maximal progress operator does not affect the termination options nor the action transitions. Timed delay transitions, however, are exhibited only if the term cannot perform a transition labeled by a prioritized action as given by rule 28.

Next, we give a racing timed bisimulation relation on closed TCP\textsuperscript{drst} terms. Intuitively, the names of the dependent racing delays must be preserved, whereas the independent ones must have the same distributions.

**Definition 14.** Two terms \( p_1, p_2 \in C(\text{TCP}^{\text{drst}}) \) are racing timed bisimilar, notation \( p_1 \approx_1 p_2 \) if there exists a racing timed bisimulation relation \( R \) such that \((\langle p_1, \alpha_0 \rangle, \langle p_2, \alpha_0 \rangle, r) \in R \) for some \( r \in \mathcal{V} \rightarrow \mathcal{V} \) satisfying \( r(X) = X \) for \( X \in \mathcal{D}(p_1) \).

The condition that \( r(X) = X \) for \( X \in \mathcal{D}(p_1) \) states that bisimilar terms must have the same dependent delays. This preserves the congruence property as dependent delays are explicitly aged by the timed delay prefix \( \sigma^X_W \), whereas independent delays cannot have an explicit age dependence. The definition may seem restrictive as it deals with process terms only in the zero environment \( \alpha_0 \). However, by an inspection of the operational rules it is easily observed that the environment does not influence the outgoing transitions nor the predicates. It is only used to properly define the underlying probabilistic timed transitions system. To show this we have the following lemma, which also justifies the use of the zero environment.

**Lemma 15.** Let \( R \) be a strong bisimulation and \((\langle p_1, \alpha_1 \rangle, \langle p_2, \alpha_2 \rangle, r) \in R \). Then there exist a bisimulation \( R' \) such that \((\langle p_1, \alpha'_1 \rangle, \langle p_2, \alpha'_2 \rangle, r) \in R' \) for every \( \alpha'_1, \alpha'_2 \in E \) satisfying \( \alpha'_1(X) = \alpha'_2(r(X)) \).

**Proof.** It is clear that the initial environments \( \alpha'_1 \) and \( \alpha'_2 \) satisfy the conditions of Definition 9 for the bisimulation relation, i.e., corresponding stochastic delays have the same ages. By direct inspection of the operational rules, one concludes that the termination options, the action, and the timed delay transitions do not depend on the aging of the delays, i.e., \( \langle p, \alpha \rangle \rightarrow_1 \langle p, \alpha \rangle \), \( \langle p, \alpha \rangle \rightarrow_{a} \langle p', \alpha_0 \rangle \), and \( \langle p, \alpha \rangle \rightarrow_{w} \).
\( \langle p'', \alpha'' \rangle \) for some \( a \in A \), \( W, L \subseteq V, p', p'' \in C(TCP^{drst}) \), and \( \alpha'' \in \mathcal{E} \), if and only if 
\( \langle p, \alpha' \rangle \leftarrow \langle p, \alpha' \rangle \rightarrow \langle p', \alpha_0 \rangle \), and \( \langle p, \alpha' \rangle \rightarrow \langle p'', \alpha'' \rangle \) for some \( \alpha', \alpha'' \in \mathcal{E} \). Thus, the states \( \langle p_1, \alpha_1 \rangle \) and \( \langle p_1, \alpha'_1 \rangle \), and \( \langle p_2, \alpha_2 \rangle \) and \( \langle p_2, \alpha'_2 \rangle \), respectively, have the same termination options and perform the same action and timed delay transitions. We conclude that the bijections that relate the stochastic delay names in the racing context of the timed delays in \( R \) and \( R' \) are the same. Now by following the operational rules for \( \langle p_1, \alpha_1 \rangle \), \( \langle p_1, \alpha'_1 \rangle \), \( \langle p_2, \alpha_2 \rangle \), and \( \langle p_2, \alpha'_2 \rangle \) it should not be difficult to see that the relation \( R' \) that has triples built of the same process terms and bijections relating the random variables of the racing delays as \( R \), but different initial environments, is a bisimulation.

Before we define the term model of TCP\textsuperscript{drst} we provide means to give operational semantics of process terms that exhibit naming conflicts. We follow the approach of [22] and we use \( \alpha \)-conversion to rename independent delay names and resolve naming conflicts.

### 3.8 \( \alpha \)-Conversion

Intuitively, two terms can be \( \alpha \)-converted if they have the same dependent delays and the names of the independent ones are consistently renamed. We illustrate the situation by an example.

**Example 16.** The term \( \sigma_2^X(\sigma_2^X \delta + \sigma_2^X \cdot \delta) \) is \( \alpha \)-convertible to \( \sigma_2^X(\sigma_2^X \delta + \sigma_2^X \cdot \delta) \) provided that \( F_X = F_S = F_T, F_Y = F_U, \) and \( F_Z = F_V \). The stochastic delay \( X \) of the topmost prefix \( \sigma_2^X \) can be renamed to \( S \), whereas \( X \) in \( \sigma_2^X \delta + \sigma_2^X \cdot \delta \) can be renamed to \( T \). These two occurrences of \( X \) are independent of each other, having in common only that they are guided by the same distribution function \( F_X \). Both \( X \) and \( Y \) in the subterm \( \sigma_2^X \delta + \sigma_2^X \cdot \delta \) must be consistently renamed to \( T \) and \( U \) in the subterm \( \sigma_2^X \delta + \sigma_2^X \cdot \delta \), respectively, to preserve the correct probabilistic behavior in the subterm as they are dependent delays. Similarly, \( Z \) is a dependent delay name that is aged by the transition of the prefixing timed delay \( \sigma_2^X \), so it must be consistently renamed in the whole term to \( V \).

To formalize the renaming as illustrated by Example 16 we introduce a predicate ccr\textsubscript{1,1}(\( p_1, D_1, p_2, D_2 \)) that checks whether the stochastic delays of the closed terms \( p_1 \) and \( p_2 \) have been consistently renamed. Renaming of dependent racing delays is represented by a bijection \( d \) between the union of the dependent racing and dependence binding delay names of the terms. It is a bijection because dependent delays of one term can have only one counterpart in the other. The renaming of the independent racing delay names is given by a total surjective relation \( i \). It is a relation because there might be multiple stochastic delays with the same name related to their counterpart with different names, e.g., the renaming of \( X \) to both \( S \) and \( T \) in Example 16. It must be a total and surjective relation as all independent delay names from one term must be related to some independent delay names of the other. Still, the renaming must be consistent.
with respect to the subterm in which independent delay names are renamed, e.g., the renaming of \( X \) to \( T \) in the subterm \( \sigma_x^L \delta + \sigma_x^R \delta \) in Example 16. As in the definition of \( I(p) \), to extract the independent delay names, we need an auxiliary set of delays \( D_1 \) and \( D_2 \) that keeps track of the intersections of the dependence binding sets. At last, two states can be \( \alpha \)-converted if the process terms can be \( \alpha \)-converted and, moreover, the environments differ only on the independent delay names provided that corresponding delays have the same age.

**Definition 17.** Two closed terms \( p_1, p_2 \in \mathcal{C}(\text{TCP}^{\text{drst}}) \) are \( \alpha \)-convertible, notation \( p_1 \sim_\alpha p_2 \), if the predicate \( \text{ccr}_{d,i}(p_1, V, p_2, V) \) given in Table 5 holds, for the identity bijection \( d: D(p_1) \cup B(p_1) \leftrightarrow D(p_2) \cup B(p_2) \) and a total surjective relation \( i \subseteq I(p_1) \times I(p_2) \).

Two states \( \langle p_1, \alpha_1 \rangle, \langle p_2, \alpha_2 \rangle \in \mathcal{C}(\text{TCP}^{\text{drst}}) \times \mathcal{E} \) are \( \alpha \)-convertible, notation \( \langle p_1, \alpha_1 \rangle \sim_\alpha \langle p_2, \alpha_2 \rangle \), if \( p_1 \sim_\alpha p_2 \) and the environment differs only on the racing independent delays provided that renamed delays have the same age, i.e., \( \alpha_1(X) = \alpha_2(Y) \) for every \( (X, Y) \in i \), and \( \alpha_0\{\alpha_1/(V \setminus I(p_1))\} = \alpha_0\{\alpha_2/(V \setminus I(p_2))\} \).

\[
\begin{align*}
\text{ccr}_{d,i}(\epsilon, D_1, \epsilon, D_2) &= \text{ccr}_{d,i}(\delta, D_1, \delta, D_2) = \top \\
\text{ccr}_{d,i}(\gamma, p_1, D_1, \gamma, p_2, D_2) \text{ if } \text{ccr}_{d,i'}(p_1, V, p_2, V) \\
&\quad \text{for a bijection } d' : D(p_1) \cup B(p_1) \leftrightarrow D(p_2) \cup B(p_2) \text{ and} \\
&\quad \text{a total surjective relation } i' \subseteq I(p_1) \times I(p_2) \\
\text{ccr}_{d,i}(\sigma^{w_1}_{l_1}, p_1, D_1, \sigma^{w_2}_{l_2}, p_2, D_2) \text{ if there exists a bijection } j : (W_1 \cup L_1) \setminus D_1 \leftrightarrow (W_2 \cup L_2) \setminus D_2 \\
&\quad \text{satisfying } j(X) = Y \text{ if } (X, Y) \in i, j(W_1 \setminus D_1) = W_2 \setminus D_2, j(L_1 \setminus D_1) = L_2 \setminus D_2, \\
&\quad d(W_1 \cap D_1) = W_2 \cap D_2, \ d(L_1 \cap D_1) = L_2 \cap D_2, \\
&\quad F_X = F_{j(X)} \text{ for } X \in (W_1 \cup L_1) \setminus D_1, \text{ and} \\
&\quad a \text{ total surjective relation } i' \subseteq I(p_1) \times I(p_2) \\
\text{ccr}_{d,i}([p_1|D_1, D'_1|p_2|D'_2] D_2) \text{ if } d(D_1) = D_2 \text{ and } \text{ccr}_{d,i}(p_1, D_1 \cap D_1, p_2, D_2 \cap D_2) \\
\text{ccr}_{d,i}(p_1 + p_1', D_1, D_2 + p_2', D_2) \text{ if } \text{ccr}_{d,i}(p_1, D_1, D_2) \text{ and } \text{ccr}_{d,i}(p_1', D_1, p_2') \text{ and} \\
\text{ccr}_{d,i}(\vartheta_H(p_1), D_1, \vartheta_H(p_2), D_2) \text{ if } \text{ccr}_{d,i}(p_1, D_1, p_2, D_2) \\
\text{ccr}_{d,i}(\vartheta_H(p_1), D_1, \vartheta_H(p_2), D_2) \text{ if } \text{ccr}_{d,i}(p_1, D_1, p_2, D_2) \\
\end{align*}
\]

| Table 5. Definition of \( \text{ccr}_{d,i}(\_\) |
The action prefix is \( \alpha \)-convertible as long as the remaining processes are \( \alpha \)-convertible. The timed delay prefix is \( \alpha \)-convertible, if there are bijections that relate the dependent and the independent delays of the losers and the winners, respectively, and the remaining processes are also \( \alpha \)-convertible. The bijection \( j \) that relates the independent delays of the losers and the winners, respectively, and the remaining processes must respect the names of the losers as given by the last two conditions.

We add one more operational rule to the ones in Table 3 and Table 4 that exploits \( \alpha \)-conversion to resolve naming conflicts as follows:

\[
\begin{align*}
(29) \quad &\langle p, \alpha \rangle \sim_{\alpha} \langle p'', \alpha'' \rangle, \quad \langle p'', \alpha'' \rangle \xrightarrow{w} \langle p', \alpha' \rangle, \quad \text{C}(p'') = \emptyset \\
\end{align*}
\]

This rule renames the independent delays that cause conflicts, thus keeping the timed delay transitions locally free of conflict. The approach is similar to the one of [22].

Remark 18. Here, however, we get a little bit sloppy as rule 29 can potentially produce infinitely many transitions, although its purpose is to support the resolution of possible naming conflicts. More precisely, the rule allows for a renaming of an independent delay to every other non-conflicting stochastic delay, whereas the intention is to use it only once. One way to formally resolve this would be to alter the logic that drives the deduction of the operational rules by introducing the \( \nabla \) operator of [23] that enables local scopes. This operator locally binds an arbitrary name, enabling a choice of names for the conflicting stochastic delay that resolves the naming conflicts. In [23] an embedding of late \( \pi \)-calculus is given in the extended logic that formalizes \( \alpha \)-conversion in that setting. Another approach would be to adopt the approach of history-based automata in order to explicitly represent the dependencies between variable names by means of relations that keep the past behavior of the system [24]. Also in this setting, an a translation of \( \pi \)-calculus to the proposed theory is given that shows how to explicitly model \( \alpha \)-conversion. In the current setting, however, we decide not to explicitly model the one-time usage of the \( \alpha \)-conversion rule as this goes beyond the scope of our work and does not contribute to the presentation of ideas in the current setting.

The following theorem shows that \( \alpha \)-conversion is a congruence relation for the closed TCP\(_{\text{drst}}\) terms. This theorem in combination with Theorem 20, which shows that \( \alpha \)-congruence implies bisimulation, enables the treatment of the process terms modulo \( \alpha \)-conversion, i.e., modulo naming of independent delays.

**Theorem 19.** \( \alpha \)-conversion is a congruence on \( C(\text{TCP}_{\text{drst}}) \).

**Proof.** It should be clear that \( \alpha \)-conversion is an equivalence relation as it is based on bijections to provide renaming of the stochastic delays. To show reflexivity, take \( d \) to be the identity bijections and \( i \) the identity relation. To show symmetry, suppose that \( p_1 \sim_{\alpha} p_2 \) for some bijection \( d \) and some total surjective
relation \( i \). Now, \( p_2 \sim_{\alpha} p_1 \) by using the reverse bijection \( d^{-1} \) and the total surjective relation \( i^{-1} \). Transitivity follows from the fact that composition of two bijections is again a bijection and a composition of two total surjective relations is again a total surjective relation.

It is straightforward that \( \alpha \)-conversion is congruence for the trivial contexts of \( \delta \) and \( \epsilon \).

Now, suppose that \( p_1 \sim_{\alpha} p_2 \) and that \( \text{ccr}_{d',i'}(p_1, p_2) \) holds for an identity bijection \( d' : D(p_1) \cup B(p_1) \rightarrow D(p_2) \cup B(p_2) \) and some total surjective relation \( i' \subseteq I(p_1) \times I(p_2) \).

For the action prefixed terms we readily have that \( \alpha.p_1 \sim_{\alpha} \alpha.p_2 \) because the conditions are trivially satisfied for the empty bijection and the empty total surjective relation on \( \emptyset \times \emptyset \) as there are no racing delays. The predicate \( \text{ccr}_{\emptyset,\emptyset}(\alpha.p_1, \alpha.p_2) \) holds.

For the timed delay prefixed terms \( \sigma_{\tau_1}^w.p_1 \) and \( \sigma_{\tau_2}^w.p_2 \) we have that the dependent delays are the same in both terms and that there are no independent terms. Thus, \( \sigma_{\tau_1}^w.p_1 \sim_{\alpha} \sigma_{\tau_2}^w.p_2 \) as \( \text{ccr}_{\emptyset,\emptyset}(\sigma_{\tau_1}^w.p_1, \sigma_{\tau_2}^w.p_2) \) holds for the identity bijection \( d : W \cup L \rightarrow W \cup L \) that is respected by the identity bijection \( d' \) on \( L \cup D(p_1) \).

For the encapsulation operator and the maximal progress operator it is straightforward that \( \partial_H(p_1) \sim_{\alpha} \partial_H(p_2) \) and \( \theta_H(p_1) \sim_{\alpha} \theta_H(p_2) \) as the dependent, dependence binding, and independent delays are the same as for \( p_1 \) and \( p_2 \). Therefore, \( \text{ccr}_{d',i'}(\partial_H(p_1), \partial_H(p_2)) \) and \( \text{ccr}_{d',i'}(\theta_H(p_1), \theta_H(p_2)) \) hold.

The dependence delays scope operator \( [\_]_D \) can introduce additional independent and dependence binding delays. We obtain the bijection \( d \) as the identical bijection on \( D(\text{dom}(d')) \). The total surjective relation \( i \) is obtained by extending \( i' \) with the additional independent delays as \( i = i' \cup \{ (X, X) \mid X \in D(p_1) \setminus D \} \).

Trivially \( d(D) = D \). We proceed by analyzing \( \text{ccr}_{d,i}(p_1|_D, D, p_2|_D, D) \). Assume that \( p_1 = \sigma_{\tau_1}^w.p'_1 \) and \( p_2 = \sigma_{\tau_2}^w.p'_2 \). Then \( d'(W_1) = W_2, d'(L_1) = L_2 \) and \( i' = \emptyset \) for the identical bijection \( d' \). It is not difficult to see that in the case the bijection \( j \) is the identical bijection on \( (W_1 \cup L_1) \setminus D \), and that \( \text{ccr}_{d,i}(p_1, D, p_2, D) \) holds. Next, assume that \( p_1 = |p'_1|_{D_1} \) and \( p_2 = |p'_2|_{D_2} \). Then, \( d'(D_1) = D_2 \) and \( \text{ccr}_{d',i'}(p_1, D_1, p_2, D_2) \) holds. Again, the same cases repeat except for the timed delay prefix. In this case we extend the existing bijection \( j \) with \( \{ (X, X) \mid X \in D(p_1) \setminus D \} \) to obtain \( j \), which is covered by the definition of \( i \). Thus, we conclude that \( \text{ccr}_{d,i}(p_1, D, p_2, D) \) holds.

Now, suppose that \( p'_1 \sim_{\alpha} p'_2 \) and that \( \text{ccr}_{d',i'}(p'_1, p'_2) \) holds for an identity bijection \( d'' : D(p'_1) \cup B(p'_1) \rightarrow D(p'_2) \cup B(p'_2) \) and some total surjective relation \( i'' \subseteq I(p'_1) \times I(p'_2) \).

To show that \( p_1 + p'_1 \sim_{\alpha} p_2 + p'_2 \) and \( p_1 \parallel p'_1 \sim_{\alpha} p_2 \parallel p'_2 \) we put \( d \) to be the identical bijection on \( D(p_1 + p'_1) \). It should be clear that it conforms to \( d' \) and \( d'' \). We build \( i \) as the union of \( i' \) and \( i'' \), i.e., \( i = i' \cup i'' \). Now, we have that \( \text{ccr}_{d,i}(p_1 + p'_1, p_2 + p'_2) \) and \( \text{ccr}_{d,i}(p_1 \parallel p'_1, p_2 \parallel p'_2) \) hold as both \( \text{ccr}_{d,i}(p_1, p_2) \) and \( \text{ccr}_{d,i}(p'_1, p'_2) \) hold, which completes the proof.

\[ \square \]

**Theorem 20.** If \( \langle p_1, \alpha_1 \rangle \sim_{\alpha} \langle p_2, \alpha_2 \rangle \) then \( \langle p_1, \alpha_1 \rangle \equiv_t \langle p_2, \alpha_2 \rangle \).
Proof. If \( C(p_1) = \emptyset \) and \( C(p_2) = \emptyset \) hold, then the relation \( i \) becomes a one-to-one total surjection and, therefore, a bijection. Moreover, \( \text{dom}(i) \cap \text{dom}(d) = \emptyset \).

Now, it should not be difficult to observe that the union \( d \cup i \) is the renaming bijection \( r \) of the bisimulation relation between the states, whereas the condition on the environments is satisfied by the definition of \( \alpha \)-conversion. \( \square \)

We will show that the bisimulation relation is a congruence, which paves the way for defining a term model for the process theory.

3.9 Term Model

The congruence property of the racing timed bisimilarity is stated in the following theorem, which is a requirement for the definition of the term model.

**Theorem 21.** The bisimilarity relation \( \approx \) is a congruence on \( C(\text{TCP}^{\text{drst}}) \).

**Proof.** Suppose that \( p_1 \approx p_2 \) and \( p_1' \approx p_2' \). Then there exist racing timed bisimulation relations \( R \) and \( R' \), respectively, such that \( ((p_1, \alpha_0), (p_2, \alpha_0), r) \in R \) and \( ((p_1', \alpha_0), (p_2', \alpha_0), r') \in R' \). The trivial contexts \( \delta \) and \( \epsilon \) are clearly bisimilar.

- \( [\cdot \cdot \cdot] \) Define \( R'' = \{(\bar{a}p_1, \alpha_0), (\bar{a}p_2, \alpha_0), \emptyset)\} \cup R \). That \( R'' \) is a bisimulation relation follows from the fact that only the following action transitions are possible: \( \bar{a}p_1, \alpha_0) \xrightarrow{a} (p_1, \alpha_0) \) and \( \bar{a}p_2, \alpha_0) \xrightarrow{a} (p_2, \alpha_0) \), and that the rest is captured by the bisimulation \( R \).

- \( [\sigma_L^w \cdot \cdot \cdot] \) By using Lemma 15, let \( R''' \) be the bisimulation relation \( p_1, \alpha_0(\alpha_0 + 1/L) \} \) and \( (p_2, \alpha_0(\alpha_0 + 1/L) \}. \). Define \( R'''' = \{(\bar{a}p_1, \alpha_0), (\bar{a}p_2, \alpha_0), \emptyset)\} \cup R'''' \). It should be clear that \( R'''' \) is a racing timed bisimulation relation.

- \( [\cdot \cdot \cdot \cdot \cdot] \) Define \( R'''' = \{(\bar{a}p_1, \alpha_0), (\bar{a}p_2, \alpha_0), \emptyset)\} \cup R \). By direct inspection of the operational rules we have that the processes \( (\bar{a}p_1, \alpha) \) and \( (p_1, \alpha) \) have the same termination options and action transitions, and result in the same states. Putting the term \( \bar{a} \) in a dependence scope may just turn a dependent delay into an independent one. However, the racing delay names remain the same. Thus, \( (p_1, \alpha) \) and \( (p_1, \alpha) \) have the same timed delay transitions as \( p_1 \) and \( p_2 \), respectively, which makes \( R'''' \) a bisimulation relation.

- \( [\partial_H \cdot \cdot \cdot] \) Define \( R'''' = \{(\partial_H(p_1), \alpha_1), (\partial_H(p_2), \alpha_2), r) \} \cup (p_1, \alpha_1), (p_2, \alpha_2), r) \). By inspection of the operational rules for \( \partial_H(\cdot) \) it should be clear that \( R'''' \) is a bisimulation by using the same bijection for the stochastic delays as \( R \).

- \( [\partial_L \cdot \cdot \cdot] \) Define \( R'''' = \{(\partial_L(p_1, \alpha_1), (\partial_L(p_2, \alpha_2), r), (p_1, \alpha_1), (p_2, \alpha_2), r) \in R) \). Now, the proof is the same as for \( \partial_H(\cdot) \).

- \( [\cdot \cdot \cdot \cdot \cdot] \) Before defining the bisimulation relation we analyze the alternative composition of \( p_1 \) and \( p_1' \). If \( (p_1 + p_1', \alpha_0) \), then \( (p_1, \alpha_0) \) or \( (p_1', \alpha_0) \), and consequently, either \( (p_2, \alpha_0) \) or \( (p_2', \alpha_0) \). It easily follows that \( (p_2 + p_2', \alpha_0) \). Similarly for the other direction.

If \( (p_1, \alpha_0) \xrightarrow{a} (p_1', \alpha_0) \), then \( R'''' \) is \( R \) on this part of the transition scheme. In the symmetric case when \( (p_1', \alpha_0) \xrightarrow{a} (p_1', \alpha_0) \), we have \( R'''' = R' \).
Possible naming conflicts arise in $p_1 + p'_1$ if $C(p_1 + p'_1) \neq \emptyset$. Let $p''_1$ and $p'''_1$ be the $\alpha$-converted versions of $p_1$ and $p'_1$, respectively, such that $C(p''_1 + p'''_1) = \emptyset$ holds. By Theorem 20 there exist bisimulation relations $R_1$ and $R'_1$ such that $((p_1, \alpha_0), (p''_1, \alpha_0), r_1) \in R_1$ and $((p'_1, \alpha_0), (p'''_1, \alpha_0), r'_1) \in R'_1$. Similarly for $p_2$ and $p'_2$ we have that $((p_2, \alpha_0), (p''_2, \alpha_0), r_2) \in R_2$ and $((p'_2, \alpha_0), (p'''_2, \alpha_0), r'_2) \in R'_2$, for some bisimulation relations $R_2$ and $R'_2$ and $p''_2 \sim_\alpha p_2$ and $p'''_2 \sim_\alpha p'_2$.

Now, we define $r''$ as $r'' = r_2 \circ r_1 \cup r'_2 \circ r'_1$. It is well defined as $C(p''_1 + p'''_1) = \emptyset$ and $C(p''_2 + p'''_2) = \emptyset$.

Now, we construct the bisimulation relation $R''$. If $p_1$ or $p'_1$ performs an action transition then the alternative composition degrades to a part of the transitions schemes of $p_1$ or $p'_1$, respectively. Similarly, if $p_1$ or $p'_1$ perform a timed delay in a resolved race. Thus, we initially put $R'' = R \cup R' \cup \{(p_1 + p'_1, \alpha_0), (p_1 + p'_1, \alpha_0), r''\}$. However, when the summands synchronize on performing a timed delay transition, i.e., when the summands perform timed delays that induce an unresolved race, then both results of the timed delay transitions persist in the final term. In this case, one proceeds in the same manner as before by induction and, again, the union of the composition of the bijections induced by the $\alpha$-conversion is computed to obtain the bijection for the bisimulation relation between the alternative compositions.

As in the case of the alternative composition, with the exception that an action transition does not make a choice.

\[ \overline{[\cdot | \cdot]} \]

Now, we can define the term model of TCP\textsuperscript{drst}.

**Definition 22.** The term model of TCP\textsuperscript{drst} is the quotient algebra $\overline{\text{P}(\text{TCP}\textsuperscript{drst})_{/\sim_1}}$, where $\text{P}(\text{TCP}\textsuperscript{drst}) = (\mathcal{C}(\text{TCP}\textsuperscript{drst}), \delta, \epsilon, \theta, \sigma^W_\text{drst}$ for $W \subseteq V$, satisfying $W \cap L = \emptyset, [\cdot]_D$ for $D \subseteq V, \partial_H(\cdot)$ for $H \subseteq A$, $\theta_H(\cdot)$ for $H \subseteq A, _+ \cdot +, _- \| \cdot_-$).

**Remark 23.** We note that because of the congruence property of the $\alpha$-conversion and because it implies bisimilarity, we could also take the set of processes to be $(\overline{\text{P}(\text{TCP}\textsuperscript{drst})_{/\sim_1}})$, as in [19, 11]. However, in the current setting we opt for explicit equations that show the $\alpha$-conversion, as we believe that this provides an additional insight in the nature of the process theory here.

## 4 Equational Theory

As we mentioned before, the associativity of the alternative composition does not hold for process terms that induce incomplete races. Moreover, the expansion of the parallel composition and the resolution of the maximal progress operator require resolved races. This forces us to give the theory TCP\textsuperscript{drst} in terms of equations on normal forms. First, we give axioms for manipulation with the dependence scope operator. Then, we introduce an intermediate normal form that enumerates all possible outcomes of a race and we use it to give expansion laws for the rest of the operators. Afterwards, we define a head norm form, in
which every operator except the alternative composition and the prefix operators is eliminated. Relying on it, we show that the equational theory presented is ground-complete. At the end of this section, we introduce guarded recursion by means of guarded recursive specifications, which have unique solutions in the term model of TCP_{drst}.

4.1 Renaming of Independent Delays

As already elaborated, the main idea behind having two types of race condition is that systems are modeled by independent delays whereas, the race condition is resolved by assigning unique names to racing delays and afterwards treating them as dependent. Thus, we need a mechanism for renaming independent delays and turning them into dependent ones. We give a simple example to illustrate the situation.

Example 24. Given the simple component $|σ_X.τ.δ|_0$, we can use it as a building block of the system $|σ_X.τ.δ|_0 \parallel |σ_Y.τ.δ|_0$. However, for analysis we revert to the system $|σ_X.τ.δ|_0 \parallel |σ_Y.τ.δ|_0$, where $F_X = F_U$ and $F_Y = F_V$. The advantage of encompassing the whole term within a single dependence scope is that all independent delays are given unique names. Moreover, the dependent delays are ‘declared’ in the dependence binding set, the parameter of the dependence scope.

Proper resolution of the race condition requires uniqueness of names of the racing delays as suggested by Definition 6 (for more details also refer to the maximal distinct representation of the terms in [11]). The mechanism that enforces all independent delays to be assigned a different name is to encompass them using a single dependence scope.

It is clear that naming conflicts may arise in such a situation, as in Example 24 above. Therefore, it has to be checked whether there are independent racing delays with conflicting names and the ones introducing the clash must be renamed. Care has to be taken to rename losing delays consistently as their names are made dependent and bound by the winners in the first race that they lost. To this end, we define a renaming operation $p[X/\bar{X}]$ (cf. Table 6) that consistently renames the stochastic delay $X$ into $Y$ in the term $p \in C(TCP_{drst})$.

By now, we have gathered all the prerequisites to present the axioms and the expansion laws for the operators.

4.2 Dependence Scope

The dependence scopes are manipulated using the axioms in Table 7. Axioms A1–A3 deal with terms that have no timed delays, so they impose an empty dependence scope. Axiom A4 states that there is no dependence of timed delays that are enabled by an action transition. Axiom A5 states that all delays are treated as dependent by default. Axiom A6 states that the losers of a timed delay retain their names and that they are treated as dependent in the remaining process.
The axioms in Table 7 are sound. We give a bisimulation relation that relates the left-hand and the right-hand side of every axiom. By using Table 7 below, we have
\[
\begin{align*}
\delta[Y/X] &= \delta, \quad \epsilon[Y/X] = \epsilon, \quad (g.p)[Y/X] = g.p \\
\partial_H(p)[Y/X] &= \partial_H(p[Y/X]), \quad \theta_H(p)[Y/X] = \theta_H(p[Y/X]) \\
(p_1 + p_2)[Y/X] &= p_1[Y/X] + p_2[Y/X], \quad (p_1 || p_2)[Y/X] = p_1[Y/X] || p_2[Y/X] \\
(\sigma_{L_1}^w.p)[Y/X] &= \sigma_{L_1}^w.p \quad \text{if } X \notin W \cup L \\
(\sigma_{L_1}^w.p)[Y/X] &= \sigma_{L_1}^w.p[\delta \cup (X) \cup \epsilon] \quad \text{if } X \in W \\
(\sigma_{L_1}^w.p)[Y/X] &= \sigma_{L_1}^w.p[\epsilon \cup (X) \cup \epsilon] \quad \text{if } X \in L \\
\langle p |_{L_2} \rangle[Y/X] &= \langle p |_{L_2} \rangle \quad \text{if } X \notin D \\
\langle p |_{L_2} \rangle[Y/X] &= \langle p |_{L_2} \rangle[\delta \cup (X) \cup \epsilon] \quad \text{if } X \in D
\end{align*}
\]

**Table 6. Renaming operation**

- A1 \( |\delta|_0 = \delta \)
- A2 \( |\epsilon|_0 = \epsilon \)
- A3 \( |g.p|_0 = g.p \)
- A4 \( |g.p|_0 = g |p|_0 \)
- A5 \( \sigma_{L_1}^w.p = \sigma_{L_1}^w.p[\epsilon \cup W \cup L] \)
- A6 \( \sigma_{L_1}^w.p = \sigma_{L_1}^w.p[\delta \cup L] \)
- A7 \( \langle p |_{L_2} \rangle |_{L_2} = \langle p |_{L_2} \rangle |_{L_2 \cup L_2} \)

**Table 7. Axioms for the dependence scope operator**

Axiom A7 states that multiple scope operators intersect. It enables the replacement of iterative application of scope operators by a single simultaneous one.

First we show that the axioms in Table 7 are sound. Afterwards we derive intermediate normal forms that enable the merger of the racing contexts of timed delay prefixed terms in the alternative composition.

**Theorem 25.** The axioms in Table 7 are sound.

**Proof.** We give a bisimulation relation that relates the left-hand and the right-hand side of every axiom. By \( \Delta(p) \) we denote the bisimulation relation satisfying \( (p, \alpha_0), (p, \alpha_0), r) \in \Delta(p) \).

A1 Define \( R = \{ (|\delta|_0, \langle \delta \rangle_0, \emptyset) \} \). It is clear that \( R \) is a bisimulation relation as both sides can do nothing.

A2 Define \( R = \{ (|\epsilon|_0, \langle \epsilon \rangle_0, \emptyset) \} \). It is clear that \( R \) is a bisimulation relation as both sides can only terminate.

A3 Define \( R = \{ (|g.p|_0, \langle g.p \rangle_0, \emptyset) \} \cup \Delta(p) \). It is clear that \( R \) is a bisimulation relation as both sides do only action transitions with a label \( g \) to \( (p, \alpha_0) \).

A4 Define \( R = \{ (|g.p|_0, \langle g.p \rangle_0, \emptyset), (|p|_0, \alpha_0), (|p|_0, \alpha_0), (\langle p |_{L_2} \rangle, \alpha_0), r) \} \cup \Delta(p) \). It is clear that \( |g.p|_0 \) and \( |g.p|_0 \) can do only action transitions labeled by \( g \) to \( (p, \alpha_0) \) and \( (\langle p |_{L_2} \rangle, \alpha_0) \), respectively. By using A7 below, we have that \( R \) is a bisimulation.

28
A5 Analogous to A3 having in mind that the dependent and independent delays of both states are the same.
A6 Analogous to A4.
A7 Define \( R = \{ (\{ p \} |_{D_1}, \alpha_0), (\{ p \} |_{D_1 \cap D_2}, \alpha_0), \tau ) \} \cup \Delta(p) \). By a direct inspection of the operational rules one concludes that both side have the same termination options, action, and timed delay transitions. Moreover, the dependent and independent racing delay names are the same.

The axioms in Table 7 enable manipulation of iterated applications of the dependence scope operator and scopes encompassing action or timed delay prefixed processes. Next, we deal with the alternative composition.

4.3 Alternative Composition

In general, associativity does not hold for the alternative composition. Intuitively, the condition for resolved racing contexts is problematic as it may depend on the order we combine the racing timed delays in incomplete races. The following example illustrates the situation.

Example 26. Consider the terms \((\sigma X, Z . \delta + \sigma Y, Z . \delta + \sigma Y, Z . \delta + \sigma X, Z . \delta)\) and \((\sigma X, Z . \delta + \sigma Y, Z . \delta + (\sigma X, Z . \delta + \sigma Y, Z . \delta))\). The transition scheme of the first term has two outgoing transitions, viz.

\[
(\sigma X, Z . \delta + \sigma Y, Z . \delta) \rightarrow_X Y \delta + \delta
\]

and

\[
(\sigma X, Z . \delta + \sigma Y, Z . \delta) \rightarrow_Y X \delta + \delta
\]

because \( \{X \} \cup \{Z\} \cap \{Y\} = \emptyset \) and \( rr(\sigma X, Z . \delta, \sigma Y, Z . \delta) \). However, the second process only deadlocks as the timed delay transitions \( \rightarrow_X Y \) of \( \sigma X, Z . \delta \) and \( \rightarrow_Y X \) of \( \sigma Y, Z . \delta \) are in inconsistent racing contexts.

Nevertheless, associativity holds for terms that comprise alternative composition of action prefixed terms and timed delay prefixed terms that are already in a context of resolved races. In this case, there is no merging of the timed delays as they are in resolved racing contexts and, therefore, the timed delay transitions are distinctly modeled by the prefixes. Such a term \( p \) can be represented in a ‘normal’ form that is unique only for the timed delays modulo commutativity, associativity, and naming of independent delays (see Remark 27 below), as

\[
p = \sum_{i=1}^{m} a_i p_i + \sum_{j=1}^{n} \sigma^{w_j}_{L_j} q_j ( + \epsilon) ( + \delta)|_{D_j},
\]

where \( W_j \cup L_j = R(p) \) for all \( 1 \leq j \leq n \) is the set of racing delay names. \( D \subseteq R(p) \) determines the dependent delay names, the summand \( \epsilon \) may or may not exist, the summand \( \delta \) exists if none of the other summands does, and \( rr(\sigma^{w_j}_{L_j}, \sigma^{w_{j'}}_{L_{j'}}) \) holds for \( 1 \leq j < j' \leq n \). The notation \( \sum_{i=1}^{m} p_i \) is shorthand for \( p_1 + \ldots + p_m \) if \( m > 0 \), and otherwise the summand does not exist.

It should be clear that \( W_j \cup L_j = W_{j'} \cup L_{j'} \) and therefore \( W_j \cup L_j = R(p) \) for every \( 1 \leq j, j' \leq n \). Then, it immediately follows that \( D(p) = D \) and \( I(p) = R(p) \setminus D \).
Remark 27. Unlike standard head normal forms, like e.g., the ones from [21, 25], we do not have \( q_j, p_i \neq q_j, p_i \) for \( 1 < i' \leq m \), at this point. This is a prerequisite for the uniqueness of the normal form and we discuss it later on. Similarly, associativity still holds if we relax the condition of the timed delay prefixes to require that \( \text{rr}(\sigma^w_{L_j}, \sigma^w_{L_j'}) \) holds or \( (W_j = W_{j'} \text{ and } L_j = L_{j'}) \) for \( 1 \leq j, j' \leq n \), relying on the fact that \( \sigma^w_{L_j} \cdot p_1 + \sigma^w_{L_{j'}} \cdot p_2 \approx \sigma^w_{L_j} \cdot (p_1 + p_2) \). We also note that when restricting to race-complete process specifications that induce only complete races, the associativity of the alternative composition holds [11]. In this special case, the situation of Example 26 cannot arise as, in that setting, the timed delays with the remaining resolved racing contexts would also be available.

We give the following law for an alternative composition of two terms in a normal form. We refer to it as axiom A8.

Theorem 28. Let \( p \) and \( p' \) have the normal forms

\[
p = \left\{ \sum_{i=1}^{m} a_i \cdot p_i + \sum_{j=1}^{n} \sigma^w_{L_j} \cdot q_j (+\epsilon)(+\delta) \right\}_D, \quad p' = \left\{ \sum_{k=1}^{m'} a_k' \cdot q_k' + \sum_{\ell=1}^{n'} \sigma^w_{L_{\ell'}} \cdot q_{\ell'} (+\epsilon)(+\delta) \right\}_{D'}
\]

with \( D \subseteq W_j \cup L_j \) and \( \text{rr}(\sigma^w_{L_{j}}, \sigma^w_{L_{j'}}) \) for \( 1 \leq j < j' \leq n \), and \( D' \subseteq W_{\ell'} \cup L_{\ell'} \) and \( \text{rr}(\sigma^w_{L_{\ell}}, \sigma^w_{L_{\ell'}}) \) for \( 1 \leq \ell < \ell' \leq n' \). If \( I(p) \cap R(p') = R(p) \cap I(p') = \emptyset \), then the normal form of their alternative composition is given by

\[
p + p' = \left\{ \sum_{i=1}^{m} a_i \cdot p_i + \sum_{k=1}^{m'} a_k' \cdot q_k' + \sum_{j, \ell} \sigma^w_{L_j \cup L_{\ell'}} \cdot (|q_j|_{L_j} + |q_{\ell'}|_{L_{\ell'}}) + \sum_{j : \text{rr}(\sigma^w_{L_j}, \sigma^w_{L_{j'}}) \text{ for all } 1 \leq j \leq n} \sigma^w_{L_j} \cdot q_j + \sum_{\ell : \text{rr}(\sigma^w_{L_{\ell}}, \sigma^w_{L_{\ell'}}) \text{ for all } 1 \leq \ell \leq n'} \sigma^w_{L_{\ell'}} \cdot q_{\ell'} (+\epsilon)(+\delta) \right\}_{D \cup D'}
\]

where the summand \( \epsilon \) exists if \( p \) or \( p' \) contain it, and \( \delta \) exists if none of the other summands does.

Proof. The required form of the dependence scope operator is easily obtained by using the axioms A1–A7 as a rewriting system from left to right. The condition \( I(p) \cap R(p') = R(p) \cap I(p') = \emptyset \) ensures that there are no naming conflicts and, thus, enables the consistent merger of the dependence scope operators.

Trivially, \( p + p' = \delta \) if \( p = p' = \delta \). Also, \( p + p' \) deadlocks if \( p \) and \( p' \) induce inconsistent races, e.g., \( \sigma^x \cdot \delta + \sigma^x \cdot \delta \). The state \( \langle p + p', \alpha_0 \rangle \) has a termination option if at least one of states \( \langle p, \alpha_0 \rangle \) or \( \langle p', \alpha_0 \rangle \) have a termination option. The termination option depends on the optional summand \( \epsilon \).
By inspection of the operational rules we have the outgoing transitions of \( \langle p, \alpha_0 \rangle \) are \( \langle p, \alpha_0 \rangle \xrightarrow{a_1} \langle \bar{p}_i, \alpha_0 \rangle \) for \( 1 \leq i \leq m \) and \( \langle p, \alpha_0 \rangle \xrightarrow{w_{f_j}} \langle q_j, \alpha_j \rangle \) for \( 1 \leq j \leq n \).

Similarly, for \( \langle p', \alpha_0 \rangle \) we have \( \langle p', \alpha_0 \rangle \xrightarrow{a_1} \langle \bar{p}_i', \alpha_0 \rangle \) for \( 1 \leq k \leq m' \) and \( \langle p', \alpha_0 \rangle \xrightarrow{w_{f'_j}} \langle q'_j, \alpha_j \rangle \) for \( 1 \leq \ell \leq n' \). Thus, the outgoing action transitions of the alternative composition are given by the term \( \sum_{i=1}^m a_i p_i + \sum_{k=1}^{m'} a'_k p'_k \).

As the timed delays in \( p \) and \( p' \) are in the context of resolved races, then they can induce a joint race only with timed delays of the other term. This is expressed by the term \( \sum_{j} \ell \cdot (W_j \cup W'_j) \cap (L_j \cup L'_j) = \emptyset \sigma_{L_j \cup L'_j} \cdot (q_j + q'_j) \).

Finally, a timed delay is considered to have a resolved racing context in the alternative composition if it is in a resolved race with all timed delays of the other summand. This is expressed by the term \( \sum_{j} \ell \cdot \tau \sigma_{L_j \cup L'_j} \cdot \sigma_{L_j} \cdot q_j \) when the timed delay in the resolved race originates from \( p \) and by the term \( \sum_{j} \ell \cdot \tau \sigma_{L_j \cup L'_j} \cdot \sigma_{L_j} \cdot q_j \) when it originates from \( p' \).

To show that \( p + p' \) is in normal form, we have to show that the race is resolved for the timed delays. Suppose that the timed delay is in the racing context induced by the winners \( W \) and the losers \( L \). First, suppose that \( W \neq \emptyset \).

Then \( (W \cap W_j) \neq \emptyset \) and \( (W \cup W_j) \cap (L \cup L_j) \neq \emptyset \) or \( (W \cap W_j) = \emptyset \) and \( W \cap L_j = \emptyset \) and \( L \cap W_j = \emptyset \).

When \( j \leq n \) and, similarly, \( (W \cup W'_j) \cap (L \cup L'_j) = \emptyset \) or \( (W \cap W'_j) = \emptyset \) and \( W \cap L'_j = \emptyset \) and \( L \cap W'_j = \emptyset \).

Now it should not be difficult to see that by inspecting all possible cases, the condition \( \tau \sigma_{L_j \cup L'_j} \cdot \sigma_{L_j} \cdot q_j \) is fulfilled for any \( W_j, L_j, W'_j, \) and \( L'_j \) satisfying \( (W_j \cup W'_j) \cap (L_j \cup L'_j) = \emptyset \).

For example, suppose that \( (W_j = \emptyset \) and \( W \cap L_j \neq \emptyset \)) and \( (W \cap W'_j = \emptyset \) and \( W \cap L'_j \neq \emptyset \) and \( L \cap W'_j \neq \emptyset \)).

Then \( (W_j \cup W'_j) = W \cap W'_j = \emptyset \) and \( (L \cap L'_j) \cap (L \cup W'_j) = \emptyset \). In the case when \( W = \emptyset \) we have only one possible case, viz. \( (W = \emptyset \) and \( L \cap W_j = \emptyset \) and \( L \cap W'_j = \emptyset \)).

Finally, it is not difficult to see that by construction the timed delays are uniquely determined modulo commutativity, associativity, and naming of independent delays, which completes the proof.

□

Using Theorem 28 we can represent every term comprising alternative composition of deadlock, termination, action, and timed prefixed terms in a normal form provided there are no naming conflicts of the independent delays. In case there are conflicts, we have to resolve them by renaming the independent delays.

### 4.4 Renaming of Independent Delays

The following theorem shows how to rename the independent delays in a consistent manner as given by Definition 17 for the \( \alpha \)-conversion. We give the renaming directly on normal forms as for incomplete races it is not possible to propagate the dependence scope operator in the alternative composition. For example, the
Let \( \sum \) be independent delay.

\text{Proof.}

We proceed by giving axioms that deal with the encapsulation operator. As the timed delays are in the context of resolved races, so the timed delay transitions such that \( r \) is captured ind\( \text{i} \)ctively by using the racing timed transition scheme of \( p \). All timed delays are in the scope of the same dependence delay operator, so there are no naming conflicts. Also, the timed delays are in the context of resolved races, so the timed delay transitions coincide with the timed delay prefixes. As \( X \notin D \), we have that \( X \) is an independent delay.

Initially, we put \( R = \{ (p, a_0, r) \} \) satisfying \( (p, a_0) \in \Delta(p) \). For outgoing action transitions and timed delay transitions such that \( X \notin W_j \cup L_j \) the resulting states coincide. If \( X \in W_j \), then the renaming of \( X \) to \( Y \) in the timed delay transition of \( p \) is captured by \( r \{ X \mapsto Y \} \) and both transition schemes result in the same state. Now, suppose that there is a state \( (\tilde{p}, \alpha) \) in the transition scheme of \( p \) that can be reached by doing timed delay transitions in which \( X \) is a loser. Then by the definition of the renaming operation and by an inspection of the operational rules it is not difficult to see that there is a state \( (p', \alpha') \) in the transition scheme of \( p' \) that can be reached by taking timed transitions in the same racing context, with the exception that \( X \) is replaced by \( Y \). We note that if the process in \( (\tilde{p}, \alpha) \) performs an action or a timed delay transition where \( X \) is not a loser, then those parts of the racing timed transition scheme are the same for both processes, which is covered by \( \Delta(p) \). Suppose \( (\tilde{p}, \alpha) \in \Delta(p) \). Then we include the triple \( (\tilde{p}, \alpha, r') \in \Delta[p] \).

We proceed by giving axioms that deal with the encapsulation operator.

Theorem 29. Let \( p \) have the normal form

\[
p = \left| \sum_{i=1}^{m} a_i \cdot p_i + \sum_{j=1}^{n} \sigma_{L_j}^{-w_j} \cdot q_j \right|_{D},
\]

where \( D \subseteq R(p) = W_j \cup L_j \) and \( \sigma_{L_j}^{-w_j} \) holds for \( 1 \leq j \leq n \). Then the independent racing delay \( X \notin D \) can be renamed to \( Y \) as follows:

\[
p = \left| \sum_{i=1}^{m} a_i \cdot p_i + \sum_{j \in \Gamma(p)} \sigma_{L_j}^{-w_j} \cdot q_j + \sum_{j : X \in W_j} \sigma_{L_j}^{-w_j} \cdot q_j \right|_{D},
\]

where the optional summands are as for \( p \).

\text{Proof.}

We build the bisimulation relation \( R \) that relates \( p = \left| \sum_{i=1}^{m} a_i \cdot p_i + \sum_{j=1}^{n} \sigma_{L_j}^{-w_j} \cdot q_j \right|_{D} \) and \( p' = \left| \sum_{i=1}^{m} a_i \cdot p_i + \sum_{j \in \Gamma(p)} \sigma_{L_j}^{-w_j} \cdot q_j + \sum_{j : X \in W_j} \sigma_{L_j}^{-w_j} \cdot q_j \right|_{D} \).

We refer to the equality given by the following theorem as axiom A9.

\[
|\sigma_{X}^{\delta} \sum_{i=1}^{m} a_i \cdot p_i + \sum_{j=1}^{n} \sigma_{L_j}^{-w_j} \cdot q_j ( + \epsilon)( + \delta)|_{D}.
\]
4.5 Encapsulation

The encapsulation operator suppresses unwanted action transitions. Unlike the alternative composition, the encapsulation operator does not require resolved races, and it freely propagates through the timed delay prefixes. It is handled using the axioms in Table 8.

\[
\begin{align*}
\partial_H(\delta) & = \delta & \text{A10} \\
\partial_H(\epsilon) & = \epsilon & \text{A11} \\
\partial_H(a.p) & = a.\partial_H(p) \text{ if } a \not\in H & \text{A12} \\
\partial_H(\delta) & = \delta \text{ if } a \in H & \text{A13} \\
\partial_H(\sigma^\nu_p) & = \sigma^\nu.\partial_H(p) & \text{A14} \\
\partial_H(p_1 + p_2) & = \partial_H(p_1) + \partial_H(p_2) & \text{A15}
\end{align*}
\]

Table 8. Axioms for the encapsulation operator

Axioms A10 and A11 deal with the deadlock and successful termination that cannot perform action transitions. If the action prefix should not be suppressed, the encapsulation operator is propagated to the remaining process \( p \) as stated in axiom A12. On the contrary, the whole process is turned to deadlock as given by A13. Axioms A14 and A15 state that encapsulation propagates through the timed delay prefixes and the alternative composition as it does not require resolved racing contexts.

**Theorem 30.** The axioms in Table 8 are sound.

**Proof.** We give a bisimulation relation that relates the left-hand and the right-hand side of every axiom. By \( \Delta(p) \) we denote the bisimulation relation satisfying \((\langle p, \alpha_0 \rangle, \langle p, \alpha_0 \rangle, r) \in \Delta(p)\).

A10 Define \( R = \{ (\langle \partial_H(\delta), \alpha_0 \rangle, \langle \delta, \alpha_0 \rangle, \emptyset) \} \).

A11 Define \( R = \{ (\langle \partial_H(\epsilon), \alpha_0 \rangle, \langle \epsilon, \alpha_0 \rangle, \emptyset) \} \).

A12 Define \( R = \{ (\langle \partial_H(a.p), \alpha_0 \rangle, \langle a.\partial_H(p), \alpha_0 \rangle, \emptyset) \} \cup \Delta(\partial_H(p)) \). As \( a \not\in H \) the left-hand state has only one possible transition \( \langle \partial_H(a.p), \alpha_0 \rangle \xrightarrow{a} \langle \partial_H(p), \alpha_0 \rangle \), which is same as the one on the right-hand side, i.e., \( \langle a.\partial_H(p), \alpha_0 \rangle \xrightarrow{a} \langle \partial_H(p), \alpha_0 \rangle \).

A13 Define \( R = \{ (\langle \partial_H(a.p), \alpha_0 \rangle, \langle \delta, \alpha_0 \rangle, \emptyset) \} \). As \( a \in H \) the left-hand state has no outgoing transitions.

A14 Define \( R = \{ (\langle \partial_H(\sigma^\nu_p), \alpha_0 \rangle, \langle \sigma^\nu.\partial_H(p), \alpha_0 \rangle, \text{id}_{W \cup L} \} \cup \Delta(\partial_H(p)) \) and proceed analogous to the proof of A12.

A15 We define \( R \) inductively on the racing timed transition scheme of \( \partial_H(p_1 + p_2) \). Initially we put \( R = \{ (\partial_H(p_1 + p_2), \partial_H(p_1) + \partial_H(p_2), r) \} \cup \Delta(\partial_H(p_1 + p_2)) \) for \( r \) satisfying \((\partial_H(p_1 + p_2), \alpha_0), (\partial_H(p_1 + p_2), \alpha_0), r) \in \Delta(\partial_H(p_1 + p_2)) \). Suppose that a state \( \langle \partial_H(p_1 + p_2), \alpha' \rangle \) is reached by taking none or more timed delay
transitions. By direct inspection of the operational rules, if \( \langle p'_1, \alpha' \rangle \) takes an action transition then the resulting state exists in the transition scheme of \( \partial_H(p_1 + p_2) \) and this case is covered by \( \Delta(\partial_H(p_1 + p_2)) \). Similarly, for resolved timed delay transitions. Unresolved timed delays synchronize for both summands, and the resulting state has the form \( \langle \partial_H(p'_1 + p'_2), \alpha'' \rangle \). We include the triple \( \langle \partial_H(p'_1 + p'_2), \alpha'', \partial_H(p'_1 + p'_2), \alpha'' \rangle \), \( \langle \partial_H(p'_1 + p'_2), \alpha'' \rangle \) in \( R \) for \( r'' \) satisfying \( \langle \partial_H(p'_1 + p'_2), \alpha'' \rangle \), \( \langle \partial_H(p'_1 + p'_2), \alpha'' \rangle \), \( \langle \Delta(\partial_H(p_1 + p_2)) \rangle \), and this case is covered by \( \Delta(\partial_H(p_1 + p_2)) \) and proceed with \( \langle \partial_H(p'_1 + p'_2), \alpha'' \rangle \). By construction \( R \) is a bisimulation.

\[ \square \]

Using the axioms from above it should not be difficult to see the application of the encapsulation operator on normal forms is given by the following corollary.

**Corollary 31.** Let \( p \) have the normal form

\[ p = \sum_{i=1}^{m} a_i p_i + \sum_{j=1}^{n} \sigma_{L_j}^{w_j} q_j (+ \epsilon)(+ \delta) \] 

where \( D \subseteq W_j \cup L_j \) and \( \text{rr}(\sigma_{L_j}^{w_j}, \sigma_{L'_j}^{w'_j}) \) holds for \( 1 \leq j < j' \leq n \). Then,

\[ \partial_H(p) = \sum_{a_i \in H} a_i \partial_H(p_i) + \sum_{j=1}^{n} \sigma_{L_j}^{w_j} \partial_H(q_j) (+ \epsilon)(+ \delta) \]

where the optional summands are as for \( p \).

**Proof.** By straightforward application of the axioms in Table 8. The normal form is preserved as there are no changes in the timed delay prefixes.

Next, we give an expansion law for the parallel composition.

### 4.6 Parallel Composition

The expansion law of the parallel composition requires resolved racing contexts. The following example illustrates the matter.

**Example 32.** Let \( p = (\sigma_{x,y}^z + \sigma_{x,y}^z, \delta) \parallel \sigma_{x,y}^z, \delta \). By first resolving the race in the left operand and afterwards eliminating the parallel composition according to the operational rules one readily obtains that \( p = \sigma_{x,y}^z \parallel \sigma_{x,y}^z = \delta \). However, if we attempt to naively expand the parallel composition as it is done in the timed process theories, we would wrongly obtain that \( p = \sigma_{x,y}^z \parallel \sigma_{x,y}^z + \sigma_{x,y}^z = \delta + \sigma_{x,y}^z \). By the following theorem gives the expansion. The expansion law is named \( \Lambda_{16} \).

**Theorem 33.** Let \( p \) and \( p' \) have the normal forms

\[ p = | \sum_{i=1}^{m} a_i p_i + \sum_{j=1}^{n} \sigma_{L_j}^{w_j} q_j (+ \epsilon)(+ \delta) |_{D'} \]

\[ p' = | \sum_{k=1}^{m'} a_k' p_k + \sum_{\ell=1}^{n'} \sigma_{L'_\ell}^{w'_\ell} q_{\ell'} (+ \epsilon)(+ \delta) |_{D'} \]

34
with \( D \subseteq W_j \cup L_j, D' \subseteq W'_l \cup L'_l \), \( \text{rr}(\sigma_{L_j}^{W_j}, \sigma_{L'_j}^{W'_j}) \) for \( 1 \leq j < j' \leq n \), and \( \text{rr}(\sigma_{L'_{j'}}^{W_{j'}}, \sigma_{L'_{j'}}^{W'_{j'}}) \) for \( 1 \leq \ell < \ell' \leq n' \). If \( I(p) \cap R(p') = R(p) \cap I(p') = \emptyset \), then the normal form of their parallel composition is given by

\[
p \parallel p' = \sum_{i=1}^{m} a_i \cdot (|p_i|_0 \parallel p') + \sum_{k=1}^{m'} b_k \cdot (|p'_k|_0) + \sum_{\gamma(\alpha, \alpha') = \delta, k, h} \sum_{j, \ell: (W_j \cup W'_j) \cap (L_j \cup L'_j) = \emptyset} \sigma_{L_j \cup L'_j}^{W_j \cup W'_j} \cdot (|q_j|_L \parallel |q'_\ell|_{L'_\ell}) \cdot ( + \epsilon)( + \delta)|_{D \cup D'} \tag{A16}
\]

where the summand \( \epsilon \) exists only if it exists in both \( p \) and \( p' \) and the summand \( \delta \) exists if none of the other summands does.

**Proof.** The first three summands of the parallel composition are directly derivable from the structural operational semantics of the action prefix operator. Also it should be clear that the parallel composition has a termination option only if both components have a termination option. As both terms are in normal form the stochastic delays from one component can only race with the stochastic delays of the other component. The condition \( I(p) \cap R(p') = R(p) \cap I(p') = \emptyset \) ensures that there are no naming conflicts. If it is not fulfilled, then we use the \( \alpha \)-conclusion law A9 to rename the conflicting independent racing delay names. The last summand captures the synchronized timed delays when both delays can delay together without any racing conflicts. Again, uniqueness of the timed delays modulo commutativity, associativity, and naming of independent delays follows by construction, which completes the proof. \( \square \)

Unlike the alternative composition, the parallel composition is associative for closed TCP\textsuperscript{dist} terms. This is an important property that supports compositional modeling. Intuitively, the parallel composition is associative as it does not allow for resolved races that obstructed the associativity of the alternative composition. This is captured in the following theorem.

**Theorem 34.** The parallel composition is associative, i.e., \( (p \parallel p') \parallel p'' = p \parallel (p' \parallel p'') \) for all \( p, p', p'' \in C(\text{TCP}\textsuperscript{dist}). \)

**Proof.** Let \( p, p', \) and \( p'' \) have the normal forms

\[
p = \sum_{i=1}^{m} a_i \cdot p_i + \sum_{j=1}^{n} \sigma_{L_j}^{W_j} \cdot q_j( + \epsilon)( + \delta)|_D
\]

\[
p' = \sum_{k=1}^{m'} a_k \cdot p'_k + \sum_{\ell=1}^{n'} \sigma_{L'_\ell}^{W'_\ell} \cdot q'_\ell( + \epsilon)( + \delta)|_{D'}
\]

\[
p'' = \sum_{r=1}^{m''} a_r \cdot p'_r + \sum_{s=1}^{n''} \sigma_{L'_s}^{W'_s} \cdot q'_s( + \epsilon)( + \delta)|_{D''}
\]
with $D \subseteq W_j \cup L_j$, $D' \subseteq W'_j \cup L'_j$, $D'' \subseteq W''_j \cup L''_j$, $rr(\sigma_{W_j} \sigma_{L_j})$ for $1 \leq j < j' \leq n$, $rr(\sigma_{W'_j} \sigma_{L'_j})$ for $1 \leq \ell < \ell' \leq n'$, and $rr(\sigma_{W''_j} \sigma_{L''_j})$ for $1 \leq s < s' \leq n''$.

Without any loss of generality, we assume that $I(p) \cap R(p') = R(p) \cap I(p') = \emptyset$, $I(p') \cap R(p'') = R(p') \cap I(p'') = \emptyset$, and $I(p) \cap R(p'') = R(p) \cap I(p'') = \emptyset$. In the opposite case, one can always use the $\alpha$-conversion law A9 of Theorem 29 to rename conflicting independent delay names. We prove the claim by total induction on the length of the terms. The initial cases are trivially satisfied as the parallel composition has a termination option only when all components have a termination option.

One calculates for $(p \parallel p') \parallel p''$ using Theorem 33 in the first and second step of the derivation, the inductive hypothesis and associativity of the synchronization function and the union set operation in the third step, and Theorem 33.
in the reverse direction in the last step:

\[
(p \parallel p') \parallel p''
\]

\[
= \left| \sum_{i=1}^{m} a_i \cdot (|p_i|_0 \parallel p') + \sum_{k=1}^{m'} a_k' \cdot (p \parallel |p_k'|_0) + \sum_{\gamma(\alpha_i \alpha_k')=b_{ik}} \sigma_{W_j;W_j'}^{(L_j;L_j')} \cdot (|q_j|_{L_j} \parallel |q_j'|_{L_j'}) \right| + \epsilon \left( + \delta \right)
\]

\[
(\parallel D) \cup (D'')
\]

\[
= \left| \sum_{i=1}^{m} a_i \cdot (|p_i|_0 \parallel (p' \parallel p'')) + \sum_{k=1}^{m'} a_k' \cdot (p \parallel |p_k'|_0 \parallel |p''|_0) + \sum_{r=1}^{m''} a_r'' \cdot (p' \parallel |p_r''|_0) + \sum_{\gamma(\alpha_{2i} \alpha_{2k}')=b_{2ik}'} \sigma_{W_j;W_j'}^{(L_j;L_j')} \cdot (|q_j|_{L_j} \parallel |q_j'|_{L_j'}) \right| + \epsilon \left( + \delta \right)
\]

\[
(\parallel D) \cup (D'')
\]

\[
= p \parallel (p' \parallel p'').
\]

\[
\square
\]

We continue with the resolution of the maximal progress operator.

4.7 Maximal Progress

The typical resolution of the maximal progress operator in timed process theory requires an additional operator that ascertains that a process has no timed delays transitions [21]. Alternatively, one can use normal forms that make the immediate action transitions and the timed delay transitions explicit. For example, \(\theta_2(p_1 + p_2 + \sigma, p_3) = \theta_2(p_1) + \theta_2(p_2)\) because the action \(a\) is prioritized.
over passage of time. Note that the maximal progress does not prioritize actions, so the second summand prefixed by the undelayable action $b$ remains.

Unlike the alternative and the parallel composition, the resolution of the maximal progress does not require resolved races. Nevertheless, for the sake of compactness we give a law on the existing normal forms without introducing an additional more relaxed type of normal form.

**Theorem 35.** Let $p$ have the normal form

$$p = \left| \sum_{i=1}^{m} a_i \cdot p_i + \sum_{j=1}^{n} \sigma_{Wj} \cdot Lj \cdot (q_j + \epsilon)(+\delta) \right|_D$$

where $D \subseteq W_j \cup L_j$ and $rr(\sigma_{Wj}, \sigma_{Wj}')$ holds for $1 \leq j < j' \leq n$. Then,

$$\theta_I(p) = \left| \sum_{i=1}^{m} a_i \cdot \theta_I(p_i)(+\epsilon)(+\delta) \right|_D \text{ if } a_i \in I \text{ for some } i \in \{1, \ldots, m\} \quad A17$$

$$\theta_I(p) = \left| \sum_{i=1}^{m} a_i \cdot \theta_I(p_i) + \sum_{j=1}^{n} \sigma_{Wj} \cdot Lj \cdot \theta_I(q_j)(+\epsilon)(+\delta) \right|_D \text{ if } \bigcup_{i=1}^{m} \{a_i\} \cap I = \emptyset \quad A18,$$

where the optional summands are as for $p$.

**Proof.** It should be clear that when at least one enabled action transitions has priority then the stochastic delay transitions are no longer available. In the opposite case, the maximal progress operator propagates through the timed delay prefix as given by the operational rules. The normal form is preserved as there are no changes in the timed delay prefixes.

Now that we provided expansion laws for all operators, we proceed by giving head normal forms for closed TCP\textsuperscript{drst} terms that support the further development of the theory.

### 4.8 Head Normal Form

Using the axioms/expansion laws for every operator, it should not be difficult to see that every closed TCP\textsuperscript{drst} term can be represented in the normal form used in the previous derivations. To eliminate multiple instances of bisimilar action prefixed terms in the alternative composition we introduce an additional axiom:

$$a \cdot p + a \cdot p = a \cdot p \quad A19.$$  

It gives idempotence of action prefixed terms in the alternative composition. It should be clear that axiom A19 is sound as $\langle a \cdot p + a \cdot p, \alpha \rangle \xrightarrow{a} \langle p, \alpha_0 \rangle$ and $\langle a \cdot p, \alpha \rangle \xrightarrow{a} \langle p, \alpha_0 \rangle$ and no other transitions are possible. It enables unique normal forms as discussed above in Remark 27. We proceed by giving a head normal form that is unique modulo commutativity, associativity, and naming of independent delays.
Corollary 36. Every closed term \( p \in \mathcal{C}(\text{TCP}^{\text{drst}}) \) can be represented in a unique head normal form modulo commutativity, associativity, and naming of independent delays, viz.

\[
p = | \sum_{i=1}^{m} a_i . p_i + \sum_{j=1}^{n} \sigma^w_{W_j} . q_j \right|_{D},
\]

where \( a_i . p_i \neq a_i . p_i' \) for \( 1 \leq i < i' \leq m \), \( D \subseteq \mathbb{R}(p) = W_j \cup L_j \), \( \mathbb{r}(\sigma^w_{W_j}, \sigma^w_{W'_j}) \) holds for \( 1 \leq j < j' \leq n \), the summand \( \epsilon \) is optional, and the summand \( \delta \) exists if none of the other summands does.

Proof. By the axioms A1 – A7 in Table 7 for manipulation with the dependence scope operator, the expansion law A8 of the alternative composition of Theorem 28, the \( \alpha \)-conversion law A9 for renaming of independent delays of Theorem 29, axioms A10 – A15 in Table 8 that deal with the encapsulation operator, the expansion law A16 of the parallel composition of Theorem 33, the expansion laws A17 and A18 of the maximal progress of Theorem 35 every closed TCP\( ^{\text{drst}} \) term can be reduced to the temporary normal form that is unique only for timed delays modulo commutativity, associativity, and naming of independent delays. By using axiom A19 for idempotence of the action prefixed terms in the alternative composition as a rewriting rule from left to write we also obtain uniqueness for the action prefixed terms modulo commutativity and associativity. \( \square \)

The availability of a head normal form is technically important. It is instrumental for proving ground-completeness and showing uniqueness of solutions of guarded recursive specifications in the term model [26].

4.9 Ground Completeness

As every term can be reduced in the head normal form given by Corollary 36, which makes all transitions explicit, it should come as no surprise that the equations given in this section form a ground-complete theory.

Theorem 37. Axioms A1 – A7 in Table 7, the \( \alpha \)-conversion law A9, axioms A10 – A15 in Table 8, the expansion laws A8, A16–A18, and axiom A19 are ground-complete for the term model \( \mathbb{I}(\text{TCP}^{\text{drst}})/\approx_t \).

Proof. The theorem is proven by natural induction on the total number of symbols in \( p_1, p_2 \in \mathcal{C}(\text{TCP}^{\text{drst}}) \). The base case is when \( p_1 \) and \( p_2 \) are either \( \delta \) or \( \epsilon \). Trivially \( \delta \approx_t \delta \) implies \( \delta = \delta \). Analogous for \( \epsilon \). Suppose that the total number of symbols is \( s \) and \( p_1 \approx_t p_2 \). By Corollary 36 we have that the head normal forms of \( p_1 \) and \( p_2 \) are given by \( p \approx_t p_1 \) and \( p' \approx_t p_2 \):

\[
p = | \sum_{i=1}^{m} a_i . p_i + \sum_{j=1}^{n} \sigma^w_{W_j} . q_j \right|_{D}, \quad p' = | \sum_{k=1}^{m'} a'_k . p'_k + \sum_{\ell=1}^{n'} \sigma^w_{W_{\ell}} . q'_\ell \right|_{D'},
\]
with \( a_i p_i \neq a_{i'} p_{i'} \) for \( 1 \leq i < i' \leq m \), \( \bar{a}_k p_k' \neq \bar{a}_{k'} p_{k'}' \) for \( 1 \leq k < k' \leq m' \), \( D \subseteq R(p) = W_j \cup L_j \), \( D' \subseteq R(p') = W'_j \cup L'_j \), \( \text{rr}(\sigma_{a_j}, \sigma_{a'_j}, \sigma_{\ell_j}) \) for \( 1 \leq j < j' \leq n \), and \( \text{rr}(\sigma_{a_i}^{W_i}, \sigma_{a_i}^{W'_i}) \) for \( 1 \leq \ell < \ell' \leq n' \).

From \( p_1 \equiv_{\tau} p_2 \) and Theorem 10 that the bisimulation relation is an equivalence it immediately follows that \( p \equiv_{\tau} p' \). Then there exists a bisimulation relation \( R \), such that \( (p, \alpha), (p', \alpha'), \tau) \in R \) for some bijection \( \tau \). Note that we use an arbitrary environment instead of the zero environment because of the inductive step, which is allowed by Lemma 15.

If \( p_j \) then it must be that \( p'_j \) and vice versa, so \( p \) contains an \( \epsilon \) summand if and only if \( p' \) contains an \( \epsilon \) summand.

Suppose that \( (p, \alpha) \xrightarrow{a} (p, \alpha_0) \). Then it must be also that \( (p', \alpha') \xrightarrow{a} (p', \alpha_0) \) and vice versa. Suppose \( \alpha_j = \alpha = \alpha' \) for some \( 1 \leq j \leq m \) and \( 1 \leq \ell \leq m' \). Then from the hypothesis \( p_j = p'_j \). Because of the idempotence of action prefixed terms in the alternative composition, the correspondence must be one-to-one, so we have that \( m = m' \). Moreover, the summands can be renumbered such that \( a_i p_i = a_i' p_i' \) for \( 1 \leq i \leq m \).

As the dependent delays must be identical in bisimilar terms, it follows that \( D = D' \). In the normal form all races are resolved, so it is not possible to merge the timed delay transitions. Thus, to every timed delay of \( p \), there corresponds exactly one timed delay from \( p' \) as the relation between the delays is given by the bijection \( \tau \). Thus, \( n = n' \) and there must be one-to-one correspondence between the timed delay transitions. The dependent delays are identical, so only the independent delays can be guided by variables with different names. However, one can use the \( \alpha \)-conversion law A9 to rename this delays, such that \( \tau \) becomes an identical bijection. Thus, we can renumber the summands such that \( W_j = W'_j \), \( L_j = L'_j \) and \( a_j = a'_j \) for \( 1 \leq j \leq n \), which completes the proof.

We proceed by introducing guarded recursion in the process theory, which enables the introduction of delayable actions and stochastic delay prefixes.

### 4.10 Guarded Recursive Specifications

We introduce guarded recursion in the process theory TCP\textsuperscript{drst} by means of guarded recursive specifications and we obtain the process theory TCP\textsuperscript{rec}.

Guardedness is a well-known concept that typically guarantees unique solution of the recursive specifications. The prerequisite is that every recursive variable must be prefixed, which ensures well-defined (predictable) behavior of the process.

A guarded recursive equation is an equation of the form \( A = p \), where \( A \in \mathcal{R} \) is a recursive variable, and \( p \) is a term over the signature of TCP\textsuperscript{drst} that additionally contains variables from \( \mathcal{R} \). Moreover, the term can be rewritten in such a way that the variables only appear in subterms prefixed by \( a_{-} \) or \( \sigma^W_{a-} \) for \( a \in A \) and \( W, L \in \mathcal{V} \) provided that \( W \cap L = \emptyset \). A guarded recursive specification \( S \in \mathcal{G}, \mathcal{G} \) denoting the set of guarded recursive specifications of our interest, is a set of guarded recursive equations with one equation for every variable. The set of recursive variables of a specification \( S \) is denoted by \( \mathcal{R}(S) \).
The definitions of dependent racing, independent racing, dependence binding, and newly enabled independent delay names are straightforwardly extended to guarded recursive specifications as $I(A) = I(p)$, $D(A) = D(p)$, $B(A) = B(p)$, and $N(A) = N(p)$, respectively, assuming that $A = p$. For the renaming of delays we have that $A[X/Y] = A'$, where $A' = p[X/Y]$ provided that $A = p$. For the notion of $\alpha$-conversion we have:

$$cc_{\alpha}(A_1, D_1, A_2, D_2) \text{ if } cc_{\alpha}(p_1, D_1, p_2, D_2) \text{ for } A_1 = p_1 \text{ and } A_2 = p_2.$$  

Solutions of recursive specifications in the term model are process terms that when replaced for the recursive variables give valid equations in the term model. By the constant $\mu A.S$ we denote a process term that is a solution for the recursive variable $A \in R(S)$ defined by the guarded recursive specification $S$. Typically, a solution of a single variable is of interest, which we refer to as the solution of the guarded recursive specification. We extend the signature of $\PiTCP^{dirst}$ with the constants $\mu A.S$ that are of our interest for $A \in R(S)$ and $S \in \mathcal{G}$. The structural operational semantics is given in Table 9.

We can generalize the notation $\mu A.S$ to $\mu p.S$, for an arbitrary term $p \in C(TCP^{dirst}_{rec})$ that contains recursive variables from $R(S)$. The definition of $\mu p.S$ is given using structural induction in Table 10 and it is supported by the operational semantics in Table 9.

$$\mu \delta.S = \delta \quad \mu \epsilon.S = \epsilon \quad \mu (\sigma^W_r p).S = \sigma^W_r (\mu p.S) \quad \mu (\sigma^W_l p).S = \sigma^W_l (\mu p.S)$$

$$\mu (\mu A.S).S = \mu A.S \quad \mu (\partial_H (p)).S = \partial_H (\mu p.S) \quad \mu (\theta_H (p)).S = \theta_H (\mu p.S)$$

$$\mu (p_1 + p_2).S = \mu p_1.S + \mu p_2.S \quad \mu (p_1 \parallel p_2).S = \mu p_1.S \parallel \mu p_2.S$$

It is straightforward from the structural operational semantics that every equation of some guarded recursive specification has a solution. Thus, the restrictive recursive definition principle, abbreviated as $\text{RDP}^-$, that every guarded recursive specification has a solution is sound in $TCP^{dirst}_{rec}$. Also, it should come as no surprise that the bisimulation relation is a congruence for recursion and
that the axioms and expansion laws are sound for $\mathbb{P}(\text{TCP}^{\text{drst}})$. Thus, we have the following term model for $\text{TCP}^{\text{drst}}$.

**Definition 38.** The term model of $\text{TCP}^{\text{drst}}$ is the quotient algebra $\mathbb{P}(\text{TCP}^{\text{drst}})/\equiv_t$ for $\mathbb{P}(\text{TCP}^{\text{drst}}) = (\mathbb{C}(\text{TCP}^{\text{drst}}), \delta, \epsilon, \mu, \sigma)$ for $S \in \mathcal{G}$ and $A \in \mathcal{R}(S), a \in \mathbb{A}, \sigma \in \mathbb{V}$ satisfying $W \cap L = \emptyset, |_{D}$ for $D \subseteq \mathbb{V}, \partial_H(\cdot)$ for $H \subseteq \mathbb{A}$, $\theta_H(\cdot)$ for $H \subseteq \mathbb{A}$, $+$, $\parallel$.

It is readily observed that $\text{TCP}^{\text{drst}}$ is a conservative extension of $\text{TCP}^{\text{drst}}$ [21, 27]. Additionally, it is not difficult to show that the head normal form of Corollary 36 is preserved. Now, by an adaptation of the proofs of [26] along the lines of [27] it can be shown that the recursive specification principle holds, relying on the existence of the head normal norm. This principle, abbreviated as RSP, states that every guarded recursive specification has at most one solution in the model. As a consequence of the validity of the principles RDP$^-$ and RSP in the model, all guarded recursive specifications have a unique solution in $\mathbb{P}(\text{TCP}^{\text{drst}})/\equiv_t$.

In the following section we employ guarded recursive specification to embed delayable actions and stochastic delays into the theory.

## 5 Process Theory DTCP$^{\text{drst}}_{\text{rec}}$

In this section we derive delayable action and stochastic delay prefixes by means of guarded recursive specifications comprising undelayable actions and timed delays as hinted above in Section 2.5. Afterwards, we analyze process specification that comprise them. We will show that when dealing with such process specifications, we need not resort to the specifications that comprise timed delay prefixes and we can manipulate with the higher order constructions directly. This gives rise to the ground-complete derived theory of communicating process with discrete stochastic time – DTCP$^{\text{drst}}_{\text{rec}}(A, V, R, \gamma)$. We illustrate the approach by modeling and solving the $\mathcal{G}/\mathcal{G}/1/\infty$ queue as an example.

### 5.1 Delayable Action Prefix and Passive Passage of Time

We define delayable action prefix scheme $\pi_-$ for $a \in \mathbb{A}$ by taking the approach of [21] and putting:

$$\pi.p = \mu A.\{A = \underline{a}.p + \sigma.A\}.$$

This process allows for an undelayable action transition at every point in time. If the action transition is taken, then the process continues to behave as $p$ and, otherwise, the process is delayed one unit of time. As the semantics of the processes is given per time unit, the process captures the intuition of a delayable action.

Of interest is the application of the encapsulation and the maximal progress operator on the delayable action prefix. For the encapsulation one obtains:

$$\partial_a(A) = \partial_a(\underline{a}.p + \sigma.A) = \delta + \sigma.\partial_a(A) = \sigma.\partial_a(A).$$

42
So, the resulting process, can only delay arbitrary long. From the discussion on stochastic delays above, it should be clear that this process is not a stochastic delay as there are no winners. However, it plays a role in the theory as it occurs as an encapsulation of a delayable action. We represent this process in the theory as the constant process passive passage of time \( \sigma^\omega \) where

\[
\sigma^\omega = \mu B. \{ B = \sigma B \}.
\]

It is a neutral element in the alternative and parallel composition for a delayable action. To see this, assume that the definitions of \( \pi.p \) and \( \sigma^\omega \) are as above. Then for \( \pi.p + \sigma^\omega \) we have that

\[
A + B = (a.p + \sigma.A) + \sigma.B = a.p + \sigma(A + B),
\]

i.e., \( \pi.p + \sigma^\omega = \pi.p \). Analogously we obtain that \( \pi.p \parallel \sigma^\omega = \pi.p \).

For the application of the maximal progress we have

\[
\theta_s(A) = \theta_s(a.p + \sigma.A) = a.\theta_s(p),
\]

i.e., its application turns a delayable action prefix into an undelayable one.

Next, we analyze the interaction between undelayable action, delayable action, and stochastic delay prefixes.

### 5.2 Stochastic Delay Prefix

We specify stochastic delays as hinted in Section 2.5, i.e., as an expiration observed per unit of time in the same racing context.

**Definition 39.** The stochastic delay prefix \([W \upsilon].p\) is defined as the solution of the following guarded recursive specification:

\[
[W \upsilon].p = \mu A. \{ A = \sigma^W_L.p + \sigma_{W \cup \upsilon}.A \}.
\]

The solution of this guarded recursive specification is an infinite racing timed transition scheme. However, the probability that a path of infinite length is taken in the probabilistic timed transition system that is induced by some assignment of distributions is equal to zero. This is because the probability distributions of the racing delays are aged by 1 in every state by the expiration of the timed delay \( \sigma_{W \cup \upsilon} \) from above and \( \lim_{n \to \infty} F(n) = 1 \) for every \( F \in \mathcal{F} \).

We illustrate by means of an example how to specify the desired stochastic behavior in this fashion.

**Example 40.** Let \( p_1 = [X].p + [Y].q \) and \( p_2 = [\upsilon].([p|_0] + [Y].q) + [X, Y].(p + q) + [\upsilon].([X].p + [q|_0]) \). Let \( [X].p = \mu A_1.S \), \( [Y].q = \mu A_2.S \), \( [\upsilon].([p|_0] + [Y].q) = \mu A_3.S \), \( [X, Y].(p + q) = \mu A_4.S \), and \( [\upsilon].([X].p + [q|_0]) = \mu A_5.S \) for

\[
S = \{ A_1 = \sigma^X.p + \sigma^X_A1, \\
 A_2 = \sigma^\psi.q + \sigma^\psi.A_2, \\
 A_3 = \sigma^X.(p|_0 + A_2) + \sigma^X_A3, \\
 A_4 = \sigma^X.(p + q) + \sigma^X_A4, \\
 A_5 = \sigma^X(A_1 + [q|_0] + \sigma^X_A5) \}.
\]
Then $p_1 = \mu(A_1 + A_2).S$ and $p_2 = \mu(A_3 + A_4 + A_5).S$. By using the expansion law A8 for the alternative composition one calculates:

$$\begin{align*}
A_1 + A_2 \\
= (\sigma^x.p + \sigma_x.A_1) + (\sigma_y.q + \sigma_y.A_2) \\
= \sigma^x.(p + q) + \sigma_y.(p|q) + \sigma_y.(A_1 + A_2) \\
+ \sigma_x.(A_3 + A_4 + A_5) \\
= \sigma^x.(p + q) + \sigma_y.(p|q) + \sigma_y.(A_1 + A_2) + \sigma_x.(A_3 + A_4 + A_5)
\end{align*}$$

Now, by following the principles RDP$^-$ and RSP for the solutions of guarded recursive specifications, $p_1$ and $p_2$ have the same solution.

Example 40 shows how to manipulate with stochastic delays by using guarded recursive specifications. However, we note that $p_1$ and $p_2$ do not specify any recursive equations and use only a stochastic delay prefix of the form $\sigma^y.B$. If, for $W, L \subseteq V$ with $W \neq \emptyset$ and $W \cap L = \emptyset$, Actually, we can manipulate with stochastic delay prefixed terms directly in any context without having to resort to the recursive specifications at all (as originally proposed in [13, 11]).

However, the interaction between timed and stochastic delays almost always requires the representation of the stochastic delays in terms of the guarded recursive specifications. We give a simple example of the interaction between stochastic and timed delay prefixes.

**Example 41.** We consider the alternative composition $\theta_I(\sigma^x.a.p + [l^y].b.q)$ for $I = \{a, b\}$. Let $[l^y].q = \mu.B \{B = \sigma^y.b.q + \sigma_{w,u}.B\}$. Then

$$\begin{align*}
\theta_I(\sigma^x.a.p + B) \\
= \theta_I(\sigma^x.a.p + \sigma_y.B) \\
= \theta(I(\sigma^x.a.p + \sigma_y.B)) \\
= \theta(I(\sigma^x.a.p + \sigma_{w,u}.B) + \sigma_{w,u}.B) \\
= \sigma^x.(\theta_I(\sigma^x.a.p + \sigma_y.B) + \sigma_{w,u}.B) \\
= \sigma^x.(\theta_I(\sigma^x.a.p + \sigma_y.B) + \sigma_{w,u}.B)
\end{align*}$$

The interaction between the timed and stochastic delays can also be used to specify a probabilistic behavior after a passage of time. An example is given in Section 7.1 where we give the specification of CACP in TCP$^{rec}$. However, the theory cannot express a standard probabilistic choice between processes that do not allow passage of time.

Next, we take a closer look at the interaction between undelayable action, delayable action, and stochastic delay prefixes.
5.3 Interaction between Undelayable Action, Delayable Action, and Stochastic Delay Prefixes

First, we investigate a common type of synchronization between delayable action and stochastic delay prefixes in the parallel composition by means of an example derivation.

Example 42. We consider the synchronization of the passage of time of the delayable action and a stochastic delay given by the term \( \partial_I(\pi.p \parallel [\gamma]q) \). We put \( I = \{a\} \), i.e., we suppress the synchronizing action as in standard compositional modeling. Let \( \pi.p = \mu A.\{A = \omega.p + \sigma.A\} \) and \( [\gamma]q = \mu B.\{B = \sigma.C + \sigma_{w.l.} B\} \). Then,

\[
\begin{align*}
\partial_I(A \parallel B) &= \partial_I((\omega.p + \sigma.A) \parallel (\sigma.C + \sigma_{w.l.} B)) \\
&= \partial_I(\omega.p \parallel B) + \sigma.C(\partial_I(A \parallel B)) \\
&= \sigma.C(a.p \parallel B) + \sigma.C(\partial_I(A \parallel B)),
\end{align*}
\]

i.e., \( \partial_I([\pi.p \parallel [\gamma]q]) = [\pi.p \parallel [\gamma]q] \).

If \( q = \tilde{b}.q' \) and the synchronization of \( a \) and \( b \) is defined, i.e., \( (a,b) = c \) for some \( c \in A \), then it is also common to prioritize this communication. For example, this can be a communication via a channel, so naturally one wants this communication to happen as soon as it is enabled. In that case, one typically has a specification of the form \( \theta_I(\partial_I(\pi.p \parallel [\gamma]q)) \) for \( I = \{a,b\} \) and \( I = \{c\} \). Then by extending the previous derivation with \( \tilde{b}.q' = \mu C.\{C = \omega.C + \sigma.C\} \) one obtains:

\[
\begin{align*}
\theta_I(\partial_I(A \parallel B)) &= \theta_I((\omega.p + \sigma.A) \parallel (\sigma.C + \sigma_{w.l.} B)) \\
&= \omega.p(a.p \parallel A) + \sigma.C(\partial_I(A \parallel B)) \\
&= \sigma.C(a.p \parallel B) + \sigma.C(\partial_I(A \parallel B)),
\end{align*}
\]

i.e., \( \theta_I([\pi.p \parallel [\gamma]q]) = [\pi.p \parallel [\gamma]q] \).

The composition of a stochastic delay prefixed process and the passive passage of time constant can also be resolved in terms of stochastic delay processes. Unlike the compositions with delayable actions, the passive passage of time propagates through the stochastic delay. We show the case of the alternative composition where \( \sigma.C = \mu C.\{C = \sigma.C\} \) and the stochastic delay is defined as above:

\[
B + C = (\sigma.C + \sigma_{w.l.} B) + \sigma.C = \sigma.C(q + \sigma.C + \sigma_{w.l.} (B + C),
\]

i.e., \( [\gamma]q + \sigma.C = [\gamma]q + \sigma.C \).

Example 42 and the previous discussion illustrate that the synchronization of passage of time of stochastic delay and delayable action prefixed terms can be handled without resorting to guarded recursive specifications comprising timed delay prefixes. Together with Example 40 and the discussion involving in Section 5.1 for delayable actions motivated us to develop a theory in the framework of TCP_{rec}^{last} that directly manipulates with delayable action and stochastic delay prefixes.

45
\[ |\sigma^{-} \omega| = \sigma^{-} \omega \quad \text{A20} \]
\[ |\pi.p|_{\emptyset} = \pi.p \quad \text{A21} \]
\[ \pi.p = \pi.|p|_{\emptyset} \quad \text{A22} \]
\[ [Y].p = [Y].p|_{W \cup L} \quad \text{A23} \]
\[ [Y].p = [Y].p|_{L} \quad \text{A24} \]
\[ \partial_H([Y].p) = [Y].\partial_H(p) \quad \text{A25} \]

Table 11. Axioms for the dependence scope encompassing stochastic delay prefixes

### 5.4 Signature

The signature of DTCP\textsuperscript{dst\_rec} comprises separate delayable action and stochastic delay prefixes, but their semantics is based on the interpretation as guarded recursive specifications in TCP\textsuperscript{drst}. The signature is given in the following definition.

**Definition 43.** The signature of DTCP\textsuperscript{dst\_rec} is given by

\[ P ::= \delta | \epsilon | \sigma^{-} \omega | | \emptyset | P | \pi.p | [Y].P | [P]_D | \partial_H(P) | \theta_I(P) | P + P | P \parallel P | \mu A.S, \]

where \( a \in A \), \( W, L, D \subseteq V \) with \( W \neq \emptyset \), \( W \cap L = \emptyset \), \( H, I \subseteq A \), \( S \in \mathcal{G} \), and \( A \in \mathcal{R}(S) \). The set of closed terms that do not contain term variables is denoted by \( \mathcal{C}(\text{DTCP}_{\text{drst}}) \) and it is ranged over by \( p \) and \( q \).

By the definition of the passive passage of time constant, the delayable action, and the stochastic delay prefix, the process theory DTCP\textsuperscript{dst\_rec} is embedded in TCP\textsuperscript{dst}. The semantics of closed DTCP\textsuperscript{dst\_rec}-terms is given by the racing timed transition scheme induced by the solutions of guarded recursive specifications that model the above constructs.

All auxiliary operations straightforwardly extended to the restriction of the theory to DTCP\textsuperscript{dst\_rec} by an application to the corresponding recursive specification. The renaming operation is extended as:

\[ \sigma^{-}[Y/X] = \sigma^{-} \]
\[ (\pi.p)[Y/X] = \pi.p \]
\[ ([W].p)[Y/X] = [W].p \quad \text{if } X \not\in W \cup L \]
\[ ([W].p)[Y/X] = ([W \setminus \{X\}] \cup \{Y\}).p \quad \text{if } X \in W \]
\[ ([W].p)[Y/X] = ([W \setminus \{Y\}] \cup \{X\}).p \quad \text{if } X \in W \]

Next, we give the additional axioms for the dependence scope and the encapsulation operator.

### 5.5 Dependence Scope and Encapsulation

The additional axioms that manage the dependence scope and encapsulation operator in DTCP\textsuperscript{dst\_rec} are given in Table 11. In the proof of the following theorem we show that the axioms are sound.
Theorem 44. The axioms in Table 11 are sound.

Proof. We prove the soundness of the axioms by showing that both sides can be rewritten to recursive specifications that have the same solution.

\[ [A20] \text{Let } \sigma = \mu A. \{ A = \sigma A \}. \text{ Then} \]
\[ |A|_\emptyset = |\sigma A|_\emptyset = \sigma A = A \]

\[ [A21] \text{Let } \bar{a}.p = \mu A. \{ A = \bar{a}.p + \sigma A \}. \text{ Then} \]
\[ |A|_\emptyset = |\bar{a}.p + \sigma A|_\emptyset = |\bar{a}.p|_\emptyset + |\sigma A|_\emptyset = \bar{a}.p + \sigma A = A \]

\[ [A22] \text{Let } \bar{a}.p = \mu A. \{ A = \bar{a}.p + \sigma A \}. \text{ Then} \]
\[ |A|_\emptyset = |\bar{a}.p + \sigma A|_\emptyset = \bar{a}.p + \sigma A = A \]

\[ [A23] \text{Let } \sigma L.p = \mu A. \{ A = \sigma L.p + \sigma_{W\cup L} A \}. \text{ Then} \]
\[ A = \sigma L.p + \sigma_{W\cup L} A = |\sigma L.p|_{W\cup L} + |\sigma_{W\cup L} A|_{W\cup L} = |\sigma L.p + \sigma_{W\cup L} A|_{W\cup L} = |A|_{W\cup L}. \]

\[ [A24] \text{Let } \sigma L.p = \mu A. \{ A = \sigma L.p + \sigma_{W\cup L} A \}. \text{ Then} \]
\[ A = \sigma L.p + \sigma_{W\cup L} A = \sigma L.p |L\emptyset + \sigma_{W\cup L} A. \]

\[ [A25] \text{Let } \sigma L.p = \mu A. \{ A = \sigma L.p + \sigma_{W\cup L} A \}. \text{ Then} \]
\[ \partial_H(A) = \partial_H(\sigma L.p + \sigma_{W\cup L} A) = \partial_H(\sigma L.p |L\emptyset) + \partial_H(\sigma_{W\cup L} A) = \sigma L.p \partial_H(|L\emptyset) + \sigma_{W\cup L} \partial_H(A). \]

Next, we deal with the expansion laws of the rest of the operators.

5.6 Alternative Composition

We derive expansion laws for the alternative composition, \(\alpha\)-conversion, the parallel composition, and the maximal progress operator for stochastic delays that deal only with undelayable action and stochastic delay prefixed terms along the lines of the expansion laws A8 for the alternative composition, A9 for the \(\alpha\)-conversion, A16 for the parallel composition, and A17 and A18 for the maximal progress operator in the timed delay setting, respectively. Again, the laws are based on normal forms in which the stochastic delays are in resolved races. The
normal forms have additional delayable action prefixes and the optional passive passage of time constant. The constant is present if no summands prefixed by a delayable action or a stochastic delay exist because it is the neutral element for the delayable action prefix and it propagates through the stochastic delays prefix as shown above in Section 5.1.

A normal form of a term \( p \in \text{DTCP}_{\text{dest}}^{\text{dist}} \) that is unique for the stochastic delays modulo commutativity, associativity, and naming of independent delays is given by

\[
p = \sum_{i=1}^{u} a_i \cdot p_i + \sum_{j=1}^{d} b_j \cdot q_j + \sum_{k=1}^{s} [W_k], r_k (+\sigma^\prime) (+\epsilon) (+\delta) |_{D},
\]

where \( D \subseteq R(p) = W_k \cup L_k \), \( \text{rr}([W_k], [\ell_k]) \) holds for \( 1 \leq k < \ell \leq s \), the summand \( \sigma^\prime \) may or may not exist provided that there are no delayable action or stochastic delay prefixed summands, the summand \( \epsilon \) may or may not exist, and the summand \( \delta \) exists if none of the other summands does.

Next, we give the expansion law for the alternative composition \( p + p' \), where

\[
p' = \sum_{i'=1}^{u'} a'_{i'} \cdot p'_{i'} + \sum_{j'=1}^{d'} b'_{j'} \cdot q'_{j'} + \sum_{k'=1}^{s'} [W'_{k'}], r'_{k'} (+\sigma^\prime) (+\epsilon) (+\delta) |_{D'},
\]

with \( D' \subseteq R(p') = W'_{k'} \cup L'_{k'} \) and \( \text{rr}([W'_{k'}], [\ell'_{k'}]) \) holds for \( 1 \leq k' < \ell' \leq n' \).

The expansion is presented in three steps: (1) for the action prefixed terms, (2) for the stochastic delay prefixed terms that form a joint race, and (3) for the stochastic delay prefixed terms in resolved races. As for the standard semantics of the alternative composition, the action transitions from both terms are available, expressed by the term \( \text{act}(p + p') \) given by

\[
\text{act}(p + p') = \sum_{i=1}^{u} a_i \cdot p_i + \sum_{i'=1}^{u'} a'_{i'} \cdot p'_{i'} + \sum_{j=1}^{d} b_j \cdot q_j + \sum_{j'=1}^{d'} b'_{j'} \cdot q'_{j'}.
\]

Recall that in a joint race of two stochastic delays \( [W_1] \) and \( [W_2] \) there are three possible outcomes: \([W_1 \cup W_2], L_1 \cup L_2\], \([W_1 \cup L_2], W_2 \cup L_1\], and \([W_1 \cup L_2], L_1 \cup W_2\]. The existence of the outcomes depends on the relation between the losers and winners of the delays (cf. Section 2.2). If one term can only allow passage of time according to the passive passage of time constant, then the stochastic delays synchronize on the passage of time, whereas the constant propagates through the prefixes. The term \( jrc(p + p') \) gives the joint outcomes of the races between the racing delays of \( p \) and \( p' \). It is given by:

\[
jrc(p + p') = \sum_{k, k': (W_k \cup W'_{k'}) \cap (L_k \cup L'_{k'}) = \emptyset} \sum_{k: W_k \cap R(p') = \emptyset} [W_k], [(r_k | L_k) + |r'_{k'}| L'_{k'}] (+\sigma^\prime) + \sum_{k': R(p) \cap L'_{k'} = \emptyset} [W'_{k'}], [(r_k | L_k) + \sum_{k''=1}^{s'} [W'_{k''}] |r'_{k''}] + \sum_{k' \cap R(p) = \emptyset} [W'_{k'}], [(\sum_{k=1}^{s} [W_k]) r_k + |r'_{k'}| L'_{k'}].
\]
The first sum expresses the case when the winners from both delays win together. The optional \( \sigma \) constant is propagated if there are no winners from one side, i.e., if the index set of either \( k \) or \( k' \) is empty. In that case the last two sums do not exist. If both summands do not have stochastic delay prefixed terms, then no sum exists. In the second sum the left delay coming from the term \( p \) wins the race, which also means that it wins the race for every stochastic delay prefixed summand of \( p' \). The third sum is the symmetric case of the second situation.

The racing delays of \( p \) and \( p' \) are in a resolved race in \( p \) and \( p' \), respectively. Thus, a racing delay from \( p \) in a resolved race in \( p + p' \) if it is in a resolved race with every racing delay of \( p' \). This is expressed by the term \( \text{rds}(p + p') \) given by:

\[
\text{rds}(p + p') = \sum_{k : \text{rr}(\ell_k^W L_k^W)} [W_k^W] \cdot r_k + \sum_{k' : \text{rr}(\ell_k^W L_k^W)} [W_k^W] \cdot r_{k'}. \]

Now, we have all the ingredients to state the expansion law of the alternative composition.

**Theorem 45.** Let \( p \) and \( p' \) have the normal forms from above. If \( I(p) \cap R(p') = R(p) \cap I(p') = \emptyset \), then the normal form of the alternative composition \( p + p' \) is given by

\[
p + p' = |\text{act}(p + p') + |\text{rsc}(p + p') + \text{rds}(p + p')(+\sigma)(+\epsilon)(+\delta)\ |_{D \cup D'} \tag{A26}
\]

where the summand \( \sigma \) exists if \( p \) or \( p' \) contain it and both of them do not have delayable action or stochastic delay prefixed summands, the summand \( \epsilon \) exists if \( p \) or \( p' \) contain it, and \( \delta \) exists if none of the other summands does.

**Proof.** To see that \( p + p' \) is again in normal form it is sufficient to observe that

1. \( \text{rr}(\ell_k^W L_k^W, [W_k^W]) \cap (L_k^W \cup L_k^W) = \emptyset \)
2. \( \text{rr}(\ell_k^W L_k^W, [W_k^W]) \cap (L_k^W \cup L_k^W) = \emptyset \)
3. \( \text{rr}(\ell_k^W L_k^W, [W_k^W]) \cap (L_k^W \cup L_k^W) = \emptyset \)
4. \( \text{rr}(\ell_k^W L_k^W, [W_k^W]) \cap (L_k^W \cup L_k^W) = \emptyset \)
5. \( \text{rr}(\ell_k^W L_k^W, [W_k^W]) \cap (L_k^W \cup L_k^W) = \emptyset \)
6. \( \text{rr}(\ell_k^W L_k^W, [W_k^W]) \cap (L_k^W \cup L_k^W) = \emptyset \)

So, \( p + p' \) is in a resolved race in \( p + p' \) and \( p' \). This is expressed by the term \( \text{act}(p + p') \) given by:

\[
\text{act}(p + p') = \sum_{k : \text{rr}(\ell_k^W L_k^W)} [W_k^W] \cdot \ell_k.
\]
Next, we show that the recursive specification of \( p+p' \) in terms of timed delay prefixed terms and its expansion have the same solution. Suppose \( \sigma' = \mu A, \{ A = \sigma A \} \), \( \bar{b}_j, q_j = \mu B_j \{ B_j = b_j, q_j + \sigma B_j \} \) for \( 1 \leq j \leq d \), \( \bar{b}_j', q_j' = \mu B_j', \{ B_j' = b_j', q_j' + \sigma B_j' \} \) for \( 1 \leq j' \leq d' \), \([ W_k ] r_k = \mu C_k \{ C_k = \sigma_{W_k} r_k + \sigma_{R(k)} C_k \} \) for \( 1 \leq k \leq s \), and \([ W_k ] r_k' = \mu C_k' \{ C_k' = \sigma_{W_k} r_k' + \sigma_{R(k)} C_k' \} \) for \( 1 \leq k' \leq s' \).

First, we analyze the alternative composition of a delayable action and a stochastic delay prefixed term. By Theorem 28 for the alternative composition of \( \bar{b}_j, q_j \) and \([ W_k ] r_k \) of \( p \) one calculates

\[
B_j + C_k = (b_j, q_j + \sigma B_j) + (\sigma_{W_k}^k r_k + \sigma_{R(k)} C_k) + \mu B_j + \sigma_{W_k}^k r_k + \sigma_{R(k)} (B_j + C_k)
\]

for some \( 1 \leq j \leq d \) and \( 1 \leq k \leq s \). It should not be difficult to see that such an alternative composition is associative.

Similarly, for the alternative composition of \([ W_k ] r_k \) and \([ W_k ] r_L \) one has:

\[
C_k + C_L = (\sigma_{W_k}^k r_k + \sigma_{R(k)} C_k) + (\sigma_{W_k}^L r_L + \sigma_{R(L)} C_L) \mu C_k + \sigma_{W_k}^L r_L + \sigma_{R(L)} (C_k + C_L)
\]

for some \( 1 \leq k, \ell \leq s \). Again, this type of alternative composition is associative. Similar results are obtained for the interaction between the delayable actions, and the interaction with the passive passage of time constant.

The normal forms of \( p \) and \( p' \) in terms of timed delay prefixed terms can be given as

\[
p = \mu (\sum_{i=1}^{u} a_i p_i + \sum_{j=1}^{d} b_j, q_j + \sum_{k=1}^{s} \sigma_{W_k}^k (r_k ( + A))) + \mu \sigma_{R(p)} ((A +) \sum_{j=1}^{d} B_j + \sum_{k=1}^{s} C_k) ( + \epsilon)( + \delta)| \mu \) \cdot S.
\]

\[
p' = \mu (\sum_{i'=1}^{u'} a'_{i'} p'_{i'} + \sum_{j'=1}^{d'} b'_{j'}, q'_{j'} + \sum_{k'=1}^{s'} \sigma_{W_{k'}} (r'_{k'} ( + A)) + \mu \sigma_{R(p')} ((A +) \sum_{j'=1}^{d'} B'_{j'} + \sum_{k'=1}^{s'} A'_{k'}) ( + \epsilon)( + \delta)| \mu \) \cdot S
\]

where the optional recursive variable \( A \) exists if the term contains the \( \sigma' \) summand and the guarded recursive specification \( S \) contains the equations for \( A, B_j, B'_j, C_k \) and \( C'_k \) for \( 1 \leq j \leq d, 1 \leq j' \leq d', 1 \leq k \leq s, \) and \( 1 \leq k' \leq s' \).
Then by Theorem 28 the expansion of \( p + p' \) is given by

\[
|\sum_{i=1}^{u} a_i p_i + \sum_{j=1}^{u'} b_j q_j + \sum_{j'=1}^{d'} b_{j'} q_{j'} + \\
\sum_{k,k':(W_k \cup W_{k'}) \cap (L_k \cup L_{k'}) = \emptyset} \sigma_{L_k \cup L_{k'}}^{W_k \cup W_{k'}} | r_k | L_k + | r_{k'} | L_{k'} \ ( + A) + \\
\sum_{k': W_k \cap R(p') = \emptyset} \sigma_{L_k^{p'}}^{W_k^{p'}} | r_k | L_k + | (A + ) \sum_{j'=1}^{d'} B_{j'} + \sum_{k'=1}^{s'} C_{k'} | R(p') + \\
\sum_{k': R(p) \cap W_{p'} = \emptyset} \sigma_{L_{p'}}^{W_{p'}} | (A + ) \sum_{j=1}^{d} B_j + \sum_{k=1}^{n} C_k | R(p) + | r_{k'} | L_{k'} + \\
\sum_{k': rr(\sigma_{L_k^{p'}}^{W_k^{p'}})} \text{ for all } 1 \leq k' \leq u' \\
\sum_{k': rr(\sigma_{L_{p'}}^{W_{p'}})} \text{ for all } 1 \leq k \leq n
\]

where the recursive variable \( A \) exists if \( p \) or \( p' \) contain it and they do not have delayable action or stochastic delay prefixed summands, the summand \( \epsilon \) exists if \( p \) or \( p' \) contain it, and \( \delta \) exists if none of the other summands does.

Now, having in mind that \( | C_k | R(p) = C_k \) and \( | C_{k'} | R(p') = C_{k'} \), and along the lines of the derivations in Examples 40 and 42 it is straightforward, but meticulous, to calculate that the expansion \( \Delta 26 \) and the expansion of \( p + p' \) using timed delay prefixed terms coincide, which completes the proof. \( \Box \)

Next, we give the \( \alpha \)-conversion in terms of stochastic delays.

### 5.7 \( \alpha \)-Conversion

Similarly to \( \alpha \)-conversion law A9 of Theorem 29 we have the following theorem for renaming independent racing stochastic delays.

**Theorem 46.** Let \( p \) have the normal form

\[
p = \left| \sum_{i=1}^{u} a_i p_i + \sum_{j=1}^{d} b_j q_j + \sum_{k=1}^{s} \left[ l_k \right] r_k ( + \sigma^\alpha) ( + \epsilon) ( + \delta) \right|_D
\]

51
where $D \subseteq R(p) = W_k \cup L_k$, $rr([W_k], [L_k])$ holds for $1 \leq k < \ell \leq s$. Then $p = p'$, where the independent delay $X \notin D$ is renamed to $Y$ in $p'$ as follows:

$$p' = \sum_{i=1}^{u} q_i \cdot p_i + \sum_{j=1}^{d} \sum_{k: X \in R(p)} [W_k] \cdot r_k + \sum_{k: X \in W_k} [(W_k \setminus \{X\}) \cup \{Y\}] \cdot r_k + \sum_{k: X \in L_k} \left[[(W_k \setminus \{X\}) \cup \{Y\}] \cdot r_k \right]_{|D}$$

**Proof.** A direct consequence of Theorem 29 as the disjoint sums range over all stochastic delay prefixed terms. □

We proceed with the resolution of the parallel composition.

### 5.8 Parallel Composition

As for the alternative composition, we split the expansion of the parallel composition in three parts: (1) resolution of an action prefix, (2) synchronization of action prefixes, and (3) resolution of the race condition. Again, unlike the alternative composition, the parallel composition is associative as a direct consequence of Theorem 34. Also, we assume that $p$ and $p'$ are in normal form as above.

By $\text{pre}(p \parallel p')$ we denote the term that takes the action prefixes out of the parallel composition. It is given by:

$$\text{pre}(p \parallel p') = \sum_{i=1}^{u} q_i \cdot (|p_i|_\emptyset \parallel p') + \sum_{i'=1}^{u'} q_{i'} \cdot (p \parallel |p'_i|_\emptyset) + \sum_{j=1}^{d} \sum_{j'=1}^{d'} b_{j'} \cdot (|q_j|_\emptyset \parallel q'_j) + \sum_{j'=1}^{d'} \sum_{j=1}^{d} b_{j'} \cdot (q \parallel |q'_j|_\emptyset).$$

As action transitions reset races, the racing delays of the summand that was prefixed by the action transition have to be made independent.

The synchronization of the action transitions is represented by the term $\text{syn}(p \parallel p')$. It is given by:

$$\text{syn}(p \parallel p') = \sum_{\gamma(a_i, a'_i) = a a_i} a a_i \cdot (|p_i|_\emptyset \parallel p'_i) + \sum_{\gamma(a_i, b'_i) = a b_i} a b_i \cdot (|p_i|_\emptyset \parallel |q'_i|_\emptyset) + \sum_{\gamma(b_j, a'_j) = b a_j} b a_j \cdot (|q_j|_\emptyset \parallel |p'_i|_\emptyset) + \sum_{\gamma(b_j, b'_j) = b b_j} b b_{j'} \cdot (|q_j|_\emptyset \parallel |q'_j|_\emptyset).$$

Cross-synchronization of undelayable and delayable actions is possible, but in that case the resulting action must be undelayable.

The stochastic delay prefixes are merged in the same manner as for the alternative composition. The joint outcomes are given by the term $\text{std}(p \parallel p')$. 

52
where $\text{std}(p \parallel p') = \sum_{k, k' : \{W_k \cup W_{k'}\} \cap \{L_k \cup L_{k'}\} = \emptyset} [W_k \cup W_{k'}]_{L_k \cup L_{k'}}.(\langle r_k | L_k \rangle ( + \sigma^-)) \parallel (\langle r_{k'} | L_{k'} \rangle ( + \sigma^-)) + \sum_{k : W_k \cap R(p')} \sum_{k'} [W_k]_{L_k}.(\langle r_k | L_k \rangle \parallel W_{k'}, r_{k'}) + \sum_{k, k' : R(p) \cap \{W_k \cup W_{k'}\} = \emptyset} [W_k \cup W_{k'}]_{L_k \cup L_{k'}}.\langle r_k | L_k \rangle \parallel W_{k'}, r_{k'}).

The leading stochastic delay determines the set of losers in the term it prefixes as in the timed setting. The optional summand $\sigma^-$ exists if one of the components does not have stochastic delay prefixes as for the alternative composition above.

Similarly to the alternative composition, we combine the three parts from above to give an expansion law for the parallel composition.

**Theorem 47.** Let $p$ and $p'$ have the normal forms as above. If $\text{I}(p) \cap R(p') = R(p) \cap \text{I}(p') = \emptyset$, then the normal form of the parallel composition of $p$ and $p'$ is given by

$$p \parallel p' = |\text{pre}(p \parallel p')| + \text{syn}(p \parallel p') + \text{std}(p \parallel p') ( + \sigma^-)( + \epsilon)( + \delta)_{\text{D} \cup \text{D}'} \text{A28},$$

where the summands $\sigma^-$ and $\epsilon$ exist if both $p$ and $p'$ contain them, respectively, and $\delta$ exists if none of the other summands does.

**Proof.** Along the lines of the proof of Theorem 45 and using the expansion law A16 of Theorem 33 for expanding the timed delay prefix representations of $p$ and $p'$. Note that if both $p$ and $p'$ have the $\sigma^-$ summand, then they cannot have stochastic delay or delayable action prefixed terms. \qed

Next, we give the expansion of the maximal progress operator.

### 5.9 Maximal Progress

Unlike the timed delay prefixed processes for which it is not important to resolve the races in order to apply the maximal progress operator, when dealing with stochastic delay prefixes all races must be resolved. We illustrate the situation by an example.

**Example 48.** Let $p = [X]_\theta.\delta + [Y]_\theta.\delta$. If we directly apply $\theta_{a,b}(p)$ and assume that it propagates through stochastic delay prefixes as for timed delay prefixes, we have $\theta_{a,b}(p) = \theta_{a,b}([X]_\theta.\delta + [Y]_\theta.\delta) = [X]_\theta.\delta + [Y]_\theta.\delta = p$. Now, assume $[X]_\theta.\delta = \mu A.\{A = \sigma^x \theta_{a,b} + \sigma^x.A\}$ and $[Y]_\theta.\delta = \mu B.\{B = \sigma^x \theta_{a,b} + \sigma^x.B\}$.

Then, by using Theorem 28 for the expansion of $\theta_{a,b}(p)$ one calculates

$$\theta_{a,b}(A + B) = \theta_{a,b}(\sigma^x_\theta.\delta + \sigma^x_.A) + (\sigma^y_\theta.\delta + \sigma^y_.B) = \theta_{a,b}(\sigma^x_\theta.\delta + \sigma^x_.A) + \sigma^x_.\theta_{a,b}(A + \theta^x_.\delta) = \sigma^x_.\theta_{a,b}(A + B) + \sigma^x_.\theta_{a,b}(A + \theta^x_.\delta) + \sigma^y_.\theta_{a,b}(A + \theta^x_.\delta) + \sigma^y_.\theta_{a,b}(A + B)$$

$$= \sigma^x_.\theta_{a,b}(A + B) + \sigma^x_.\theta_{a,b}(A + \theta^x_.\delta) + \sigma^y_.\theta_{a,b}(A + \theta^x_.\delta) + \sigma^y_.\theta_{a,b}(A + B).$$

53
Thus, \( \theta_{a,b}(p) = [X]_a \delta + [X,Y]_a (a \delta + b \delta) + [Y]_b \delta. \)

Following the guidelines of Example 48 and having in mind that the maximal progress operator turns delayable actions to undelayable ones (cf. Section 5.1) we have the following theorem.

**Theorem 49.** Let the normal form of \( p \) be as above. Then the expansion law of the maximal progress \( \theta_I(p) \) is given by

\[
\theta_I(p) = \left| \sum_{i=1}^{u} a_i . \theta_I(p_i) + \sum_{j=1}^{d} b_j . \theta_I(q_j)(+\epsilon)(+\delta) \right|_D
\]

if \( \bigcup_{i=1}^{u} \{a_i\} \cup \bigcup_{j=1}^{d} \{b_j\} \cap I \neq \emptyset \) A29

\[
\theta_I(p) = \left| \sum_{i=1}^{u} a_i . \theta_I(p_i) + \sum_{j=1}^{d} b_j . \theta_I(q_j) + \sum_{s} [W_s] \theta_I(r_k)
\]

\[
(+\sigma)(+\epsilon)(+\delta) \right|_D
\]

if \( \bigcup_{i=1}^{u} \{a_i\} \cup \bigcup_{j=1}^{d} \{b_j\} \cap I = \emptyset \) A30,

where the conditions apply for the optional summands as in \( p \).

**Proof.** By direct application of Theorem 33 for the expansion of the maximal progress operator for timed prefixed delays. \( \Box \)

Similar to the timed process theory we can give head normal forms that pave the way for a ground-complete result and unique solutions to the guarded recursive specifications.

### 5.10 Head Normal Form

It should come as no surprise that every closed DTCP\textsubscript{det} term can be rewritten in a head normal form, in a similar way as the one of Corollary 36, as all operators can be expressed using an alternative composition of undelayable action, delayable action, and stochastic delay prefixed terms. However, to show this we require two more idempotent axioms that deal with undelayable and delayable action prefixed terms. They can be stated as follows:

\[
a.p + \overline{a}.p = \overline{a}.p \text{ A31,} \quad \overline{a}.p + \overline{a}.p = \overline{a}.p \text{ A32.}
\]

To show the soundness of the axioms assume that \( \overline{a}.p = \mu A \cdot \{ A = \overline{a}.p + \sigma A \} \).

Then by using axiom A19 and the expansion A8 of the alternative composition:

\[
\begin{align*}
[A31] \quad & a.p + A = a.p + (a.p + \sigma A) = a.p + a.p + \sigma A = a.p + \sigma A = A \\
[A32] \quad & A + A = (a.p + \sigma A) + (a.p + \sigma A) = a.p + \sigma (A + A). 
\end{align*}
\]

By the principles of RDP\textsuperscript{-} and RSP, we have that \( A \) and \( A + A \) have the same solution.

The head normal form is stated in the following corollary.
Corollary 50. Every closed term \( p \in \mathcal{C}(\text{DTCP}_{\text{dst}}) \) can be represented in a unique head normal form modulo commutativity, associativity, and naming of independent delays, viz.

\[
p = \left| \sum_{i=1}^{u} a_i \cdot p_i + \sum_{j=1}^{d} b_j \cdot q_j + \sum_{k=1}^{s} [W_k] L_k \cdot r_k (\alpha) (\sigma^\omega) (\epsilon^\omega) (\delta) \right|_D,
\]

where \( a_i \cdot p_i \neq a_{i'} \cdot p_{i'} \) and \( a_i \cdot p_i \neq b_j \cdot q_j \) for \( 1 \leq i, i' \leq u \) with \( i \neq i' \), and \( 1 \leq j \leq d \), \( D \subseteq R(p) = W_k \cup L_k \), \( \tau (\left( [W_k] \right), \left( [L_k] \right)) \) holds for \( 1 \leq k < \ell \leq s \), the summand \( \sigma^\omega \) may or may not exist provided that there are no delayable action or stochastic delay prefixed summands, the summand \( \epsilon \) may or may not exist, and the summand \( \delta \) exists if none of the other summands does.

Proof. The proof is analogous to the one of Corollary 36 by replacing the axioms and expansion laws that deal with timed delays with ones that deal with stochastic delays. By the axioms A1 – A4 and A7 in Table 7 and axioms A23 and A24 in Table 11 for manipulation with the dependence scope operator, the expansion law A26 of the alternative composition of Theorem 45, the \( \alpha \)-conversion law A27 for renaming of independent delays of Theorem 46, axioms A10 – A13 and A15 in Table 8 and axiom A25 in Table 11 that deal with the encapsulation operator, the expansion law A28 of the parallel composition of Theorem 47, the expansion laws A29 and A30 of the maximal progress of Theorem 49 every closed \( \text{TCP}_{\text{dst}} \) term can be reduced to the temporary normal form that is unique only for timed delays modulo commutativity, associativity, and naming of independent delays. By using axioms A19, A31, and A32 for idempotence of the action prefixed terms in the alternative composition as a rewriting rule from left to write we also obtain uniqueness for the action prefixed terms modulo commutativity and associativity.

As before, since every term can be reduced in the head normal form given by Corollary 50 the equations for a ground-complete theory.

Theorem 51. Axioms A1 – A4 and A7 in Table 7 and axioms A23 and A24 in Table 11, axioms A10 – A13 and A15 in Table 8 and axiom A25 in Table 11, the expansion laws A26–A30, and axioms A19, A31, and A32 are ground-complete for the term model \( \text{IP}(\text{DTCP}_{\text{dst}})/\tau_t \).

Proof. Analogous to the proof of Theorem 37 for the timed setting.

Next, we show the simplifications that can be applied in the case of race-complete process specifications that induce only races with all possible outcomes.

5.11 Race-Complete Process Specifications

Race-complete process specifications can be characterized as specifications that can be rewritten such that only stochastic delay prefixes of the form \( [X]_r \), for
\( X \in \mathcal{V} \) occur in the process term. This restriction assures that all possible outcomes of the race are given and it enables the associativity of the alternative composition. As a consequence, the equational theory becomes much more elegant as we do not have to resort to normal forms from the start. However, the expansion of the parallel composition still requires a (head) normal form in which the timed/stochastic delays are in resolved racing contexts. Also, to resolve the maximal progress either an additional operator or a normal form that makes explicit the undelayable action prefixes and the timed delays is required.

We present in Table 12 the alternative simplified axioms for the alternative composition and renaming of independent racing delays of race-complete process specifications.

\[ [X].p |_0 = [Y].p |_0 \quad \text{if} \ F_X = F_Y \]  
\[ |p_1 + p_2|_D = |p_1|_D + |p_2|_D \quad \text{if} \ I(|p_1|_D) \cap R(|p_2|_D) = R(|p_1|_D) \cap I(|p_2|_D) = \emptyset \]  
\[ \epsilon + \epsilon = \epsilon \]  
\[ p + \delta = p \]  
\[ p + q = q + p \]  
\[ (p + q) + r = p + (q + r) \]  
\[ [W].p_1 + [W].p_2 = [W].([p_1]_{L_1} + [p_2]_{L_2}) \]  
\[ \text{if} \ W_1 \cap W_2 \neq \emptyset \quad \text{and} \quad W_1 \cap L_2 = L_1 \cap W_2 = \emptyset \]  
\[ [W].p_1 + [W].p_2 = [L_1 \cup L_2].([p_1]_{L_1} + [p_2]_{L_2}) \]  
\[ \text{if} \ L_1 \cap W_2 \neq \emptyset \quad \text{and} \quad W_1 \cap W_2 = W_1 \cap L_2 = \emptyset \]  
\[ [W].p_1 + [W].p_2 = [W_1 \cap W_2].([p_1]_{L_1} + [p_2]_{L_2}) + [W_1 \cap W_2].([p_1]_{L_1} + [p_2]_{L_2}) \]  
\[ \text{if} \ W_1 \cap W_2 = L_1 \cap W_2 = W_1 \cap L_2 = \emptyset \]  

Table 12. Alternative simplified axioms in case of race-complete process specifications

Let us first comment the additional axioms for the timed setting. Axiom A33 shows how to rename independent racing delays. Axiom A34 is a simplified version of the merging of dependence scopes of the expansion law A8 of the alternative composition. The naming conflict condition remains the same. Axioms A35 and A36 are the standard axioms for the idempotence of the termination and the neutrality of the deadlock in the alternative composition. Axioms A37 and A38 state the commutativity and associativity of the alternative composition. Axiom A39 shows the merging of the stochastic delays when the winners have a common stochastic delay. Axioms A40 give the resolution of the race when the winner comes from the left summand. In that case, the stochastic delays of the right summand must be made dependent on the winners of the first summand. The dependent racing delays of the remaining process of the left summand can come only from the set of losers \( L_1 \). Finally, axiom A41 gives all possible outcomes
when there are no restrictions on the merging of the racing delays. The axioms can be turned into a rewriting system to give the normal form from Section 4.3 and they replace the expansion laws A26 and A27 for the alternative composition and \( \alpha \)-conversion, respectively.

To illustrate the features of the process theory \( \text{DTCP}^{\text{det}} \) we specify and solve the \( G/G/1/\infty \) queue example.

### 5.12 \( G/G/1/\infty \) Queue

We proceed by specifying and solving the \( G/G/1/\infty \) queue, also discussed in [13]. The queue can be compactly modeled by a generalized semi-Markov process [9] given in Fig. 1. Here, \( a \) denotes the event of an arrival job and \( s \) is an event of a processed job. The states are labeled by the clocks that correspond to events. In every state the clocks with which the state is labeled are reset, whereas the others are updated typically using spent-lifetime semantics. After an expiration of a clock, the transition labeled by the name of the event is taken. The model shows that jobs arrive constantly in the queue and the server processes one job at a time.

![Fig. 1. Generalized semi-Markov model of the \( G/G/1/\infty \) queue](image)

We specify the \( G/G/1/\infty \) queue in our setting by using three components given by the recursive equations for \( A, Q_0, \) and \( S \). The equation for \( A \) models the arrival process that is delayed by the stochastic delay \( [X] \). This delay corresponds to the clock \( a \) in the generalized semi-Markov representation of the process in Fig. 1. The process modeled by \( Q_0 \) is the standard representation of a queue. It comprises delayable actions and it is always able to receive a new job or to offer a job that has already been queued. Finally, the process given by \( S \) models the server that has processing time distributed according to \( Y \). Its counterpart in Fig. 1 is given by the clock \( s \). It is always ready to accept a job when it is idle.

\[
\begin{align*}
A & = \|[X], s_1 \cdot A\|_g \\
Q_0 & = r_1, Q_1 \\
Q_{k+1} & = r_1, Q_{k+2} + s_2, Q_k \quad \text{if } k \geq 0 \\
S & = r_2, [Y], s_3, S
\end{align*}
\]

and \( H = \{s_1, r_1, s_2, r_2\} \) and \( I = \{c_1, c_2, s_3\} \). The specification of the \( G/G/1/\infty \) queue itself is given by

\[
Q = \Theta_I(\partial_H(A \parallel Q_0 \parallel S))
\]
Along the lines of Examples 40 and 42 one calculates:

\[
Q = S_0 = \theta_1(\partial_H(A \parallel Q_0 \parallel S)) \\
= \theta_1(\partial_H([X],\pi_1.A \parallel \pi_1.Q_1 \parallel \pi_2,[Y],\pi_3.S)) \\
= \theta_1(\partial_H([X],\pi_1.A \parallel \pi_1.Q_1 \parallel \pi_2,[Y],\pi_3.S)_{\theta_1}) \\
= \|X\|_{\xi_1} \theta_1(\partial_H(A \parallel (\pi_1.Q_2 + \pi_2.Q_0) \parallel \pi_2,[Y],\pi_3.S))_{\theta_1} \\
= \|X\|_{\xi_1} \theta_1(\partial_H(A \parallel Q_0 \parallel [Y],\pi_3.S)_{\theta_1}) \\
= \|X\|_{\xi_1} \theta_1(\partial_H(A \parallel Q_0 \parallel [Y],\pi_3.S)_{\theta_1})_{\theta_1} \\
= \|X\|_{\xi_1} \theta_1(\partial_H(A \parallel Q_0 \parallel [Y],\pi_3.S)_{\theta_1})_{\theta_1}
\]

\[S_1 = \theta_1(\partial_H(A \parallel Q_0 \parallel [Y],\pi_3.S)_{\theta_1})\]

\[= \theta_1(\partial_H([X],\pi_1.A \parallel \pi_1.Q_1 \parallel [Y],\pi_3.S)) + [X].([X],\pi_1.A \parallel \pi_1.Q_1 \parallel \pi_3.S) + [X].(\pi_1.A \parallel \pi_1.Q_1 \parallel \pi_3.S)_{\theta_1})\]

\[= \theta_1(\partial_H([X],\pi_1.A \parallel [Y],\pi_3.S) + [X].(\pi_1.A \parallel \pi_1.Q_1 \parallel S) + [X].(\pi_1.A \parallel [Y],\pi_3.S)_{\theta_1})\]

\[= \theta_1(\partial_H([X],\pi_1.A \parallel [Y],\pi_3.S) + [X].(\pi_1.A \parallel \pi_1.Q_1 \parallel S) + [X].(\pi_1.A \parallel [Y],\pi_3.S)_{\theta_1})_{\theta_1}\]

\[= \|X\|_{\xi_1} \theta_1(\partial_H(A \parallel Q_0 \parallel [Y],\pi_3.S)) + [X].(\pi_1.A \parallel \pi_1.Q_1 \parallel S) + [X].(\pi_1.A \parallel [Y],\pi_3.S)_{\theta_1})_{\theta_1}\]

Similarly, one can show that:

\[S_k = \|X\|_{\xi_1} S_{k+1} + \|X\|_{\xi_1} S_{k} + [X,Y].(\xi_1,\xi_2,S_k) + \|X\|_{\xi_1} \theta_1(\partial_H(A \parallel Q_0 \parallel [Y],\pi_3.S)_{\theta_1})_{\theta_1}\]

which completes the solution for the G/G/1/∞ queue, where

\[S_{k+1} = \theta_1(\partial_H(A \parallel Q_k \parallel [Y],\pi_3.S)_{\theta_1})_{\theta_1}\]

We note, however, that the specification is simply more elegant, whereas the underlying racing timed transition scheme abstracted on delayable actions and stochastic delays is still as complicated as the corresponding stochastic transition scheme in [13].

Next, we take the opposite view and attempt to establish a stochastic process theory in such a way that it extends the standard real-time setting. However, first we need to introduce the notion of context-sensitive interpolation that represents a restriction of time additivity that conforms to the race condition.

6 Extending Real Time with Stochastic Time

In this section we take the viewpoint of stochastic time and we attempt to mold real-time process algebras so that they can easily be extended to a stochastic setting. We give a simple example to illustrate the situation.

Example 52. Suppose we wish to extend the term \(\sigma^x \cdot \sigma^y \cdot p\) with stochastic time. If we make use of time additivity, i.e., only observe the accumulative delays, we may consider, e.g., the term \(\sigma^x \cdot p\) or even \(\sigma^y \cdot \sigma^x \cdot p\). However, from the properties of the race condition (cf. Section 2) we have that \([X_2],[X_3],p\) is different from \([X_5],p\)
and \([X_1],[X_3],[X_1].p\) for any non-Dirac random variable \(X_1, X_2, X_3, X_5 \in \mathcal{V}\), suitably chosen to represent the delays of duration 1, 2, 3, and 5, respectively. The reason is that in a context of every race \([X_5]\) produces different probabilities and samples for the winning delays than \([X_2],[X_3]\).

One solution is to consider timed delays as atomic, i.e., to explicitly state the delay that we want to model. In that way timed and stochastic delays are put on the same level and they are viewed as discrete events. The motivation for such an approach stems from a discussion on the overlapping properties of prominent stochastic bisimulation relations.

### 6.1 Stochastic Bisimulation Relations

In general, timed bisimulation relations require that bisimilar processes delay the same amount of time. They typically employ time additivity, i.e., merging of subsequent timed delays into a joint single delay with the same accumulative duration, to compare the delays \([20, 21]\). For example, \(\sigma^3 \cdot \sigma^2 \cdot p\) and \(\sigma^5 \cdot p\) are typically considered to be equivalent.

On the contrary, stochastic bisimulation relations are set up as discrete event bisimulation relations (which is inherent to the underlying performance model), i.e., they consider passage of time per an atomic stochastic delay transition. To the best of our knowledge, with the exception of \([28]\), all stochastic process theories consider stochastic bisimulation that is atomic in this sense: in \([1]\) the actions are coupled with the stochastic clocks, in \([3]\) there is an alternation between clocks and action transitions, whereas in \([2, 4]\) the merging is impeded by the combination of the pre-selection policy and start-termination semantics. Although originally introduced as an atomic stochastic bisimulation \([5]\), an effort is made in \([28]\) to define a notion of weak stochastic bisimulation that merges subsequent stochastic delays. Unfortunately, such an approach is not compositional as merging of stochastic delays does not support the race condition. A simple example illustrates the problem. The process \([X].[Y].p\) intuitively has the same stochastic properties as the process \([Z].p\) provided that \(F_Z = F_{X+Y}\). However, standard compositions involving these processes are not bisimilar. For example, \([X].[Y].p + [U].p\) is not bisimilar to \([Z].p + [U].p\) because the race of \(X\) and \(U\) induces a different probabilistic choice on the winner compared to the race between \(Z\) and \(U\).

We conclude that from the viewpoint of stochastic process theories that employ the race condition, it is more convenient to treat timed delays as atomic, discrete event constructs, which levels the semantic differences with their stochastic counterpart.

### 6.2 Extending Real Time with Stochastic Time

The treatment of timed delays as atomic requires a new more restrictive notion of time additivity. Again, we illustrate the situation by an example.
Fig. 2. a) A timed delay prefix $\sigma^n.p$, b) Arbitrary interpolation of $\sigma^n$ into $\sigma^{n'}$, $\sigma^{n''}$, and $\sigma^{n'''}$, c) Parallel composition of $\sigma^n.p$ and $\sigma^m.q$, and d) Context-sensitive interpolation of $\sigma^n$ in the context of the parallel composition with $\sigma^m.q$

Fig. 2b depicts arbitrary interpolation of the timed delay $\sigma^n$ of the process $\sigma^n.p$ of Fig. 2a to three timed delays $\sigma^{n'}$, $\sigma^{n''}$, and $\sigma^{n'''}$ satisfying $n' + n'' + n''' = n$. If interpreted as an atomic timed delay, the delay must be left intact, unless it is in a context of a composition that would induce a race. A race with another timed delay $\sigma^m$ of the process $\sigma^m.q$ induced by a parallel composition is depicted in Fig. 2c. Only then we can interpolate the longer delay (in this case $n > m$, as depicted in Fig. 2d, conforming to race condition semantics. We note that the resulting process $(\sigma^n - m.p) \parallel q$ accounts for the remaining delay $\sigma^n - m$.

Fig. 3. a) Stochastic extension of the composition in Fig. 2c), b) Independent race condition with every possible outcome, c) Stochastic extension of $\sigma^n.p$ in accordance with Fig. 2d), and d) Dependent race condition synchronizing the dependent delays

In the stochastic setting of this paper, such behavior can be interpreted both for the independent or dependent race condition as depicted in Fig. 3. Suppose that the original timed delay $\sigma^n$ is replaced by the stochastic delay $[X].p$ as depicted in Fig. 3a, and $\sigma^m.q$ is extended to $[Y].q$. In Fig. 3b we consider an independent race given by the term $[X].p \parallel [Y].q$, which results in all possible outcomes as discussed in Section 2. Here, we label the transitions with the winners on top and the losers below the arrow. This approach conveniently models independent components competing for the same resource.

Now, suppose that the components are considered dependent regarding their timing aspects. For example, $\sigma^n.p$ is a controller that has a timeout greater than the tolerated response time of the process that it controls. This can be represented in the timed model as $\sigma^m.q$ and conditioned by the fact that $n > m$. In such a situation the stochastic modeling using the independent race condition leads to undesirable behavior. For example, the premature expiration of the stochastic delay of the controller given by the outcome $[X]$ could introduce non-

60
existent deadlock behavior as it did not wait for the result of the process that successfully finished its task. In this case, relying on the context-sensitive interpolation, the correct modeling of $\sigma^n.p$ would be $[Y].[Z].p$ as depicted in Fig. 3c. The idea is that both, the controller and the process, should synchronize on the dependent stochastic delay $[Y]$. The delay is followed by the short timeout $[Z]$ that models the extra timed delay $\sigma^n - m$ in the context-sensitive interpolated representation $(\sigma^n.(\sigma^n - m).p) || (q)$ of $(\sigma^n.p) || (\sigma^n.q)$. The situation is depicted in Fig. 3d.

Another way of modeling the above system is to explicitly state that the stochastic delay $[Y]$ should be the winner of the race between $[X]$ and $[Y]$. This is done by specifying $\sigma^m.q$ in stochastic time as $[Y].[X].p$. Such a specification expresses the result of the race between $[X]$ and $[Y]$. The parallel composition $[X].p || [Y].q$ is resolved as $[Y].[X].p$. In this case, however, the race is incomplete, i.e., the other disjoint outcomes $[X] . ([X].p || q)$ are not present. As discussed above, a major consequence is that the equational theory of terms exhibiting incomplete races is more intricate as the alternative composition is no longer associative and one must rely on normal form representations.

We conclude that the use of context-sensitive interpolation helps in identification of the nature of the stochastic delays by allowing the treatment of timed delays as atomic. However, it should be noted that the its use cannot always reveal whether the delays should be interpreted as independent or dependent in stochastic time. This still remains the task of the one who is extending the specification.

### 6.3 Context-Sensitive Interpolation

From a process theoretical point of view, fundamental properties of time are time determinism and time additivity, i.e., passage of time does not make a choice by itself and subsequence timed delays can be merged together to the accumulative delay, respectively [20, 21]. They are captured by the following operational rules.

\[
\begin{align*}
\sigma^n.p & \rightarrow p \\
\sigma^m.p & \rightarrow p' \\
\frac{p_1 \rightarrow p'_1, p_2 \rightarrow p'_2}{p_1 + p_2 \rightarrow p'_1 + p'_2}, \\
\frac{p_1 \rightarrow p'_1, p_2 \rightarrow p'_2}{p_1 + p_2 \rightarrow p'_1 + p'_2}, & m < n \\
\frac{p_1 \rightarrow p'_1, p_2 \rightarrow p'_2}{p_1 + p_2 \rightarrow p'_1 + p'_2}, & m > n
\end{align*}
\]

When treating timed delays as atomic, rule 33 holds again, but rule 34 for time additivity now fails. Therefore, we add instead of rule 34, two new rules similar to rule 35 for time determinism that enable context-sensitive interpolation when competing delays have different durations:

\[
\begin{align*}
\frac{p_1 \rightarrow p'_1, p_2 \rightarrow p'_2, m < n}{p_1 + p_2 \rightarrow p'_1 + \sigma^{n - m}.p'_2} \\
\frac{p_1 \rightarrow p'_1, p_2 \rightarrow p'_2, m > n}{p_1 + p_2 \rightarrow \sigma^{m - n}.p'_1 + p'_2}
\end{align*}
\]

Note the emphasis on performing the shortest winning duration first. Rules 36 and 37 give rise to the following axioms.

$\sigma^n.p_1 + \sigma^m.p_2 = \sigma^m.(p_1 + p_2) \ A42$ \hspace{1cm} $\sigma^n.p_1 + \sigma^m + n.p_2 = \sigma^m.(p_1 + \sigma^n.p_2) \ A43$.
Axiom A42 enables time determinism, whereas Axiom A43 replaces the standard axiom for time additivity $\sigma^m \cdot \sigma^n \cdot p = \sigma^{m+n} \cdot p$. Together with commutativity the latter allows for context-sensitive interpolation. If zero delays are allowed, then rule 35 and axiom A42 become obsolete. More details can be found in [29].

To conclude, at first sight context-sensitive interpolation may seem too restrictive compared to time additivity. However, context-sensitive interpolation does exactly what time additivity is typically used for: merging of delays with the same duration by taking the shortest/minimal possible delay in a context with compositional operators. Moreover, context-sensitive interpolation fits naturally in the expansion of the parallel composition, which makes it a suitable candidate for a finer notion of time additivity in real-time process algebras. Finally, the bisimulation relation remains unchanged as context-sensitive interpolation is handled in the operational semantics on the model level. However, it is noted that the resulting process equivalence is finer. For example, $\sigma^2 \cdot \sigma^3 \cdot p$ and $\sigma^2 \cdot p$ are no longer related, though $\sigma^2 \cdot \sigma^3 \cdot p$ and $\sigma^2 \cdot p + \sigma^2 \cdot \delta$ are, where $\delta$ represents the deadlock process constant obeying $p + \delta = p$.

We proceed by presenting a stochastic process theory that makes use of the concepts discussed above to deal with stochastic time as an extension of the real-time process theory. The gain lays in an expansion law that respects time determinism and the explicit treatment of the maximal progress.

6.4 Stochastic Process Theory TCP$_{st}$

We proceed with the presentation of TCP$_{st}$($A, \mathcal{V}, R, \gamma$), the theory of communicating processes with (discrete) stochastic time. Unlike the process theory DTCP$_{dst}$, which was molded from primitives of TCP$_{first}$ from which the delayable action and stochastic delay prefixes were derived, here, we give semantics of closed TCP$_{st}$-terms from scratch. In return, we obtain a greater insight into the relationship between real time and the race condition, leading to a new notion of context-sensitive interpolation. The semantics is given in terms of stochastic transition schemes that, in essence, represent stochastic automata with explicit symbolic representation of the race condition and passage of time. We focus on the handling of the race condition, the expansion for the parallel composition, and the maximal progress operator. We note that we obtain the same equational theory as for DTCP$_{dst}$, but the semantics is given in terms of finite objects as passage of time is observed on an atomic delay scale and not per unit of time.

6.5 Environments and Distributions of Racing Delays

As before, we use an environment to keep track of the dependencies between the racing delays. Recall, $[W, L]$ denotes an outcome of a race that was won by $W$ and lost by $L$ for disjoint $W, L \subseteq \mathcal{V}$ with $W \neq \emptyset$. However, in view of time determinism, time has passed equally for all racing delays in $W \cup L$. To denote that after a stochastic delay $[W, L]$, the same amount of time that has passed for the winners $W$ has also passed for the losers $L$, we use an environment $\eta: \mathcal{V} \to 2^{\mathcal{V}}$. 

62
For each \( X \in \mathcal{V} \), \( \eta(X) \) is a set that contains one representative of the winners of every race that \( X \) lost. One representative suffices, because all winners share the same sample in the winning race. If \( \eta(X) = \emptyset \), then \( X \) has never lost a race. We write \( \mathcal{H} \) for the set of all such environments. We illustrate the use of these environments by means of an example.

**Example 53.** The process term \([X, Y, Z] \cdot U.Z.p\) has a stochastic delay transition in which \( X \) and \( Y \) are the winners and \( Z \) is the loser. In the resulting process \([Z] \cdot U.Z.p\), the variable \( Z \) must be made dependent on the amount of time that has passed for \( X \) and \( Y \) before. This can be denoted either by \( \eta(Z) = \{X\} \) or \( \eta(Z) = \{Y\} \), assuming that initially \( \eta(Z) = \emptyset \). As \( Z \) again loses a race, this time to \( U \), the transition induced by \([Z] \cdot U.Z.p\) updates \( \eta(Z) \) to \( \eta(Z) = \{X, U\} \), provided \( X \) was chosen as a representative in the first race.

Again, the environment does not affect the outgoing transitions. It is used only to calculate the correct distribution of the racing delays. Suppose that \( \rho \in \mathcal{E} \) keeps track of the exhibited samples for an expired delay \([X]\) of a race as in the racing timed transitions schemes. Then the racing delay \([Y]\) in the environment \( \eta \) has an age, i.e., it participated in races that it lost with the total amount of time

\[
a_{\eta,\rho}(Y) = \sum_{X \in \eta(Y)} (\rho(X) + a_{\eta,\rho}(X)).
\]

By convention, \( a_{\eta,\rho}(Y) = 0 \) if \( \eta(Y) = \emptyset \). The distribution of \( Y \) at that point in time is \( F_Y \mid a_{\eta,\rho}(Y) \), provided that \( F_Y(a_{\eta,\rho}(Y)) < 1 \). Thus, in order to compute the updated distribution of the racing delay \([Y]\), one has to know its racing history, i.e., the names of all delays that contribute in the derivation of its age \( a_{\eta,\rho}(Y) \). The racing history in a environment \( \eta \) of a set of racing delays \( R \) is defined by

\[
H_\eta(R) = R \cup \bigcup_{X \in R} (\eta(X) \cup H_\eta(\eta(X))).
\]

### 6.6 Stochastic Transition Schemes

Closed TCP\(^\text{rec} \) terms are given semantics by means of stochastic transition schemes that treat passage of time of stochastic delays as atomic. The stochastic transitions schemes are based on the same idea as the racing timed transition schemes, i.e., keeping track of the aging of the racing delays. As passage of time in stochastic transition schemes is observed as a discrete event in terms of expired winners, then keeping track of the ages amounts to preserving the racing history. To represent passive (unbounded) passage of time for delayable actions we use an additional transition relation \( \Rightarrow \). It also gives the semantics of the passive passage of time constant \( \sigma^\omega \). Outgoing passive delay transitions exist only if the state does not have outgoing stochastic delay transitions.

**Definition 54.** A *stochastic transition scheme* \((S \times \mathcal{H}, \mathcal{L}, \mathcal{V}, \rightarrow, \equiv, \Rightarrow, \downarrow, I)\) is a tuple, where \( u = \langle s, \eta \rangle \in S \times \mathcal{H} \) is a state in an environment \( \eta \) and
\(- \rightarrow \subseteq (S \times \mathcal{H}) \times \mathcal{L} \times (S \times \mathcal{H})\) is the labeled transition relation;

\(- \implies \subseteq (S \times \mathcal{H}) \times (2^V \setminus \{\emptyset\}) \times 2^V \times (S \times \mathcal{H})\) is a stochastic delay transition relation satisfying that for every \(u \xrightarrow{w}{\mathcal{L}} u'\) it holds that the winners and the losers are disjoint, i.e., \(W \cap L = \emptyset\), and for every two different transitions originating from the same state \(u \xrightarrow{w_1}{\mathcal{L}} u_1 \neq u \xrightarrow{w_2}{\mathcal{L}} u_2\) the predicate \(rr([W_1], [W_2])\) holds.

\(- \rightsquigarrow \subseteq (S \times \mathcal{H}) \times (S \times \mathcal{H})\) is the passive delay transition. It can relate two states \(u \rightsquigarrow u'\) only if the state \(u\) does not have any outgoing stochastic delay transitions.

\(- \downarrow \subseteq S \times \mathcal{H}\) is the termination option predicate;

\(- 1: S \rightarrow \mathcal{V}\) is the independent racing delays function. It satisfies that \(I(s) \subseteq \bigcup (s, n) \xrightarrow{w}{\mathcal{L}} (s', \eta')\) \((W \cup L)\), for every \(\eta \in \mathcal{H}\).

Again, we have that \(W_1 \cup L_1 = W_2 \cup L_2\) for every \(u \xrightarrow{w_1}{\mathcal{L}} u_1\) and \(u \xrightarrow{w_2}{\mathcal{L}} u_2\) as \(rr([W_1], [W_2])\) holds. Thus, for every state \(s\) there exists a set of racing delays \(R(s)\) and \(I(s) \subseteq R(s)\). Then, the set of dependent racing delays is given by \(D(s) = R(s) \setminus I(s)\).

Each stochastic transition scheme coupled with an assignment of probability distributions to the stochastic delays induces a probabilistic timed transition system. As the racing history is represented symbolically, we have to calculate the age of the racing delays as given above. For that reason we also need an initial environment, standardly set to the zero sample environment \(\rho_0\). The action transitions and the termination predicate are adopted from the stochastic transition scheme as above in Definition 8. The probability measure of the probabilistic timed delay is induced by the winners, the losers, and the sample of the winning delay. We make use of the notation

\[ RC_n(W, L) = P(W = n, L > n), \]

extending the one of Definition 8. The formal definition is as follows.

**Definition 55.** Let \(R = (S \times \mathcal{H}, A, V, \rightarrow, \implies, \downarrow, I)\) be a stochastic transition scheme and \(d: V \rightarrow \mathcal{F}\) a distribution assignment function. Then, \((R, d)\) induces the probabilistic timed transition system \(P = ((S \times \mathcal{H}) \times \mathcal{E}, A, d, \rightarrow, \implies, \downarrow)\), where the action transition and termination options \(\rightarrow\) and \(\downarrow\) of \(P\) are induced by \(\rightarrow\) and \(\downarrow\) of \(R\), respectively, and \(\implies(v) = (\mathcal{N} \times (S \times \mathcal{H}) \times \mathcal{E}, P)\) with \(v = (\langle s, \eta \rangle, \rho)\), is the probability space induced by the race condition. The probability measure \(P\) is given by

\[
P(n, v') = \begin{cases} 
\frac{RC_n(W', L')}{\sum_{(s, \eta)} \frac{n, v'}{\mathcal{L}}(W < L)} & \text{if } \langle s, \eta \rangle \xrightarrow{w'} (s', \eta') \text{ or} \\
0 & \text{if } \langle s, \eta \rangle \xrightarrow{w'} (s', \eta'), 
\end{cases}
\]

\[
P(1, v') = 1 \quad \text{if } \langle s, \eta \rangle \xrightarrow{w'} (s', \eta'),
\]

where \(v' = (\langle s', \eta' \rangle, \rho')\), the distribution functions of \(X \in R(s)\) are given by \(F_X = d(X)\langle n/\rho(X)\rangle\), and \(\rho' = \rho_0(\rho \{n/W'\}/H_{\rho}(L'))\).
The probabilistic transition system is built from the states of the stochastic transition scheme coupled with a sample environment. The samples are used to calculate the aged distributions of the racing delays. The race induces a probabilistic choice, which is normalized by the accumulative probability of the outgoing stochastic delay transitions. The normalization is required in case of incomplete races. Each probabilistic timed delay transition updates the sample environment by first assigning the winning sample to the winners. Afterwards, only the meaningful part of the environment given by the racing history of the losers is retained. The passive delay transitions are modeled as in the timed setting by self-loops of probabilistic timed transitions with probability 1.

We define a strong bisimulation relation on stochastic transition schemes that requires stochastic delays to have the same dependence history modulo names of the independent delays. It is the counterpart of the racing timed bisimulation relation.

6.7 Bisimulation

We define a strong bisimulation relation on stochastic transition schemes that requires stochastic delays to have the same dependence history modulo names of the independent delays. It is the counterpart of the racing timed bisimulation relation for the stochastic setting. As before, the condition ensures that the induced races have the same probabilistic behavior. In the current setting, we must also account for the behavior of the passive delay transitions.

**Definition 56.** Let \( R \subseteq (\mathcal{S} \times \mathcal{H})^2 \times (\mathcal{V} \hookrightarrow \mathcal{V}) \) be a symmetric relation. Then \( R \) is a stochastic bisimulation relation if for all \((s_1, \eta_1), (s_2, \eta_2), r) \in R\) it holds that \( r: \mathcal{H}_{\eta_1}(R(s_1)) \leftrightarrow \mathcal{H}_{\eta_2}(R(s_2))\) is a bijection with \( r(I(s_1)) = I(s_2)\), and \( F_X = F_{r(X)}\) and \( r(\eta_1(X)) = \eta_2(r(X))\) for \( X \in \text{dom}(r)\), and:

1. if \( u_1 \downarrow \) then \( u_2 \uparrow \);
2. if \( u_1 \leadsto u'_1 \) for some \( u'_1 \in \mathcal{S} \times \mathcal{H} \), then \( u_2 \leadsto u'_2 \) for some \( u'_2 \in \mathcal{S} \times \mathcal{H} \) such that \( (u'_1, u'_2, r') \in R \) for some \( r' \in \mathcal{V} \hookrightarrow \mathcal{V} \);
3. if \( u_1 \stackrel{a}{\rightarrow} u'_1 \) for some \( u'_1 \in \mathcal{S} \times \mathcal{H} \), then \( u_2 \stackrel{a}{\rightarrow} u'_2 \) for some \( u'_2 \in \mathcal{S} \times \mathcal{H} \) such that \( (u'_1, u'_2, r') \in R \) for some \( r' \in \mathcal{V} \hookrightarrow \mathcal{V} \); and
4. if \( u_1 \stackrel{W_1}{\rightarrow} u'_1 \) for some \( u'_1 = \langle u'_1', \eta'_1 \rangle \in \mathcal{S} \times \mathcal{H} \), then \( u_2 \stackrel{W_2}{\rightarrow} u'_2 \) for some \( u'_2 = \langle u'_2', \eta'_2 \rangle \in \mathcal{S} \times \mathcal{H} \) where \( r(W_1) = W_2 \), \( r(L_1) = L_2 \), and \( (u'_1, u'_2, r') \in R \) for some \( r' \in \mathcal{V} \hookrightarrow \mathcal{V} \) satisfying \( r'(X) = r(X) \) for \( X \in \mathcal{H}_{\eta_1}(L_1 \cap D(s'_1))\).

We say that two states \( u_1 \) and \( u_2 \) are stochastic bisimilar, notation \( u_1 \sim_s u_2 \), if there exists a stochastic bisimulation relation \( R \) such that \( (u_1, u_2, r) \in R \) for some \( r \in \mathcal{V} \hookrightarrow \mathcal{V} \).

The bisimulation relation is adapted for the stochastic setting by generalizing the environment from an age of the distribution to a racing history of expired winning delays. As the history also depends on the names of the delays, the bisimulation also must cater for the consistency of the complete history of the losers. This is expressed by the last condition \( r'(X) = r(X) \) for \( X \in \mathcal{H}_{\eta_1}(L_1 \cap D(s'_1))\). The extension is pretty straightforward and the proofs from TCP\textsubscript{dir} naturally extend to the new setting. The following theorem states without proof that stochastic bisimilarity is an equivalence relation.

65
Theorem 57. Stochastic bisimilarity is an equivalence relation.

We continue with the presentation of the operational semantics.

6.8 Structural Operational Semantics

The operational semantics of TCP_{rec} suffers from the same impediments as the one of TCP_{drst}. Again, for a closed term \( p \in \mathcal{C}(TCP_{rec}) \) to have proper semantics, the conflicting independent racing delay names have to be detected and renamed. We use the already established notions of dependent racing, independent racing, dependence binding, and newly enabled independent delay names to identify the conflicting names and set up \( \alpha \)-conversion. Because the environment holds the racing history of the racing delays in terms of stochastic delay names, the complete history has to be included in the detection of naming conflicts as well. We give a simple example to illustrate the situation.

Example 58. Let \([X],[Z].\delta + [Y].\delta\) be a term in an environment \( \eta \) with \( \eta(Y) = \{Z\} \) and \( \eta(Z) = \{U\} \). If \([X]\) wins the race, the resulting term is \([Z].\delta + [Y].\delta\) with \( \eta(Y) = \{Y,Z\} \) and \( \eta(Z) = \{U\} \). Now, the conflict arises because \([Z]\) is a newly enabled independent delay, but because of the racing history of \([Y]\) it has been wrongly made dependent on the sample of \( U \).

We denote the environment as a subscript and we extract the set of conflicting names \( \mathcal{C}_\eta(p) \) of a term \( p \in \mathcal{C}(TCP_{drst}) \) in an environment \( \eta \) as given in Table 13.

\[
\begin{align*}
\mathcal{C}_\eta(\epsilon) &= \mathcal{C}_\eta(\delta) = \mathcal{C}_\eta(a.p) = \emptyset, \\
\mathcal{C}_\eta([p]_W) &= \mathcal{C}_\eta(\partial_H(p)) = \mathcal{C}_\eta(\theta_H(p)) = \mathcal{C}_\eta(p) \\
\mathcal{C}_\eta(p_1 + p_2) &= \mathcal{C}_\eta(p_1 \parallel p_2) = \\
& (I(p_1) \cup \partial_H(R(p_2))) \cup (\partial_H(R(p_1))) \cup (\partial_H(R(p_1)) \cap (I(p_2) \cup \partial_H(R(p_2))) \cup \partial_H(R(p_2))) \cup C_\eta(p_1) \cup C_\eta(p_2).
\end{align*}
\]

Table 13. Set of conflicting names in an environment \( \eta \)

For notational convenience we write \( \eta_0 \) for \( \eta_0(X) = \emptyset \) for \( X \in \mathcal{V} \). By \( \eta + W \) we denote \( (\eta + W)(X) = \eta(X) \cup \{Y\} \) for \( X \in \mathcal{V} \), a non-empty set \( W \subseteq \mathcal{V} \), and a randomly chosen \( Y \in W \). The notational conventions \( \Rightarrow \Rightarrow \Rightarrow \) and \( \leftarrow \leftarrow \leftarrow \) express that the term does not have any outgoing stochastic delay or passive delay transitions, respectively. Now that we have all prerequisites we give the structural operational semantics in Table 14 for the constant processes, the prefix operators, and the dependence scope, Table 15 for the alternative composition, Table 16 for the parallel composition, and Table 17 for the encapsulation and maximal progress operator, and the recursion. We comment some the rules, different from the ones for TCP_{drst}.
Rule 39 states that the passive passage of time constant has an outgoing passive delay transition. Rule 40 states that undelayable action prefixes perform only immediate action transitions. Rules 41 and 42 state that delayable action prefixes induce both an undelayable action and a passive delay transition. Rule 43 enables stochastic delay transitions. The environment is updated in two phases. First, the dependence sets of the losers are updated resulting in the new environment $\eta'$. Afterwards, only the relevant dependence history of the losers, given by $H_{\eta'}(L)$, is retained. The losers in resulting term $|p|_B$ are treated as dependent as their names must be protected. Again, the dependence scope operator does not affect any transitions as illustrated by the rules 44 – 47 and it is only used to specify dependent and independent racing delay names.

Rules 48–51 are as before. Rules 52 – 55 illustrates the default weak choice between action transitions and passage of time. Here, we have two transitions that denote passage of time, so both of them must be disabled in the other term. Rule 56 states that passive delay transitions merge. Rules 57 and 58 state that passive passage of time synchronizes with stochastic delays. Resolution of races is given by the rules 59 – 61. Unlike the timed delays that had predetermined racing context, stochastic delays resolve the races dynamically. The environment in rules 59 and 60 is again updated in two phases as for rule 43, but now with the joint set of losers obtained by resolving the race. When the delays have winners that exhibit the same sample, the environment is simply a merger of the resulting environments of the delays as given by rule 61. As in the timed setting, rules 62 and 63 express that a stochastic delay transition of one summand is in a resolved race if its racing delays are in a resolved race with the ones of every outgoing stochastic delays of the other summand.

Rules 64 – 69 give the standard behavior for the parallel composition for the termination and action transitions as for TCP$\text{rec}$. Rule 70 gives the synchronization of the passive delay transitions, whereas rules 71 and 72 give the synchronization of the passive and stochastic delay transitions as for the alternative composition. Rules 73 – 75 show the resolution of races analogous to the
they represent disjoint events that cannot occur simultaneously.

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>( (p_1, \eta) \overset{\alpha_1}{\rightarrow} (p_1', \eta_1) ) ( (p_1 + p_2, \eta) \overset{\alpha_1}{\rightarrow} (p_1', \eta_1) )</td>
</tr>
<tr>
<td>51</td>
<td>( (p_2, \eta) \overset{\alpha_2}{\rightarrow} (p_2', \eta_2) ) ( (p_1 + p_2, \eta) \overset{\alpha_2}{\rightarrow} (p_2', \eta_2) )</td>
</tr>
<tr>
<td>52</td>
<td>( (p_1, \eta) \overset{\omega_1}{\rightarrow} (p_1', \eta_1) ) ( (p_2, \eta) \overset{\omega_2}{\rightarrow} (p_2', \eta_2) )</td>
</tr>
<tr>
<td>53</td>
<td>( (p_1, \eta) \overset{\omega_1}{\rightarrow} (p_1', \eta_1) ) ( (p_2, \eta) \overset{\omega_2}{\rightarrow} (p_2', \eta_2) )</td>
</tr>
<tr>
<td>54</td>
<td>( (p_1 + p_2, \eta) \overset{\omega_1}{\rightarrow} (p_1', \eta_1) ) ( (p_2, \eta) \overset{\omega_2}{\rightarrow} (p_2', \eta_2) )</td>
</tr>
<tr>
<td>55</td>
<td>( (p_1 + p_2, \eta) \overset{\omega_1}{\rightarrow} (p_1', \eta_1) ) ( (p_2, \eta) \overset{\omega_2}{\rightarrow} (p_2', \eta_2) )</td>
</tr>
<tr>
<td>56</td>
<td>( (p_1 + p_2, \eta) \overset{\omega_2}{\rightarrow} (p_2', \eta_2) )</td>
</tr>
<tr>
<td>57</td>
<td>( (p_1, \eta) \overset{\omega_1}{\rightarrow} (p_1', \eta_1) ) ( (p_2, \eta) \overset{\omega_1}{\rightarrow} (p_1', \eta_1) )</td>
</tr>
<tr>
<td>58</td>
<td>( (p_1, \eta) \overset{\omega_1}{\rightarrow} (p_1', \eta_1) ) ( (p_2, \eta) \overset{\omega_2}{\rightarrow} (p_2', \eta_2) )</td>
</tr>
<tr>
<td>59</td>
<td>( (p_1, \eta) \overset{\omega_1}{\rightarrow} (p_1', \eta_1) ) ( (p_2, \eta) \overset{\omega_2}{\rightarrow} (p_2', \eta_2) ) ( W_1 \cap (W_2 \cup L_2) = 0 )</td>
</tr>
<tr>
<td>60</td>
<td>( (p_1 + p_2, \eta) \overset{\omega_1}{\rightarrow} (p_1', \eta_1) ) ( (p_2, \eta) \overset{\omega_2}{\rightarrow} (p_2', \eta_2) ) ( W_1 \cup (L_1 \cup L_2) = 0 )</td>
</tr>
<tr>
<td>61</td>
<td>( (p_1 + p_2, \eta) \overset{\omega_1}{\rightarrow} (p_1', \eta_1) ) ( (p_2, \eta) \overset{\omega_2}{\rightarrow} (p_2', \eta_2) ) ( W_1 \cap (W_2 \cup L_2) = \emptyset )</td>
</tr>
<tr>
<td>62</td>
<td>( (p_1, \eta) \overset{\omega_1}{\rightarrow} (p_1', \eta_1) ) ( \rho([W_1', L_1]) ) ( (p_2, \eta) \overset{\omega_2}{\rightarrow} (p_2', \eta_2) )</td>
</tr>
<tr>
<td>63</td>
<td>( (p_1, \eta) \overset{\omega_1}{\rightarrow} (p_1', \eta_1) ) ( (p_2, \eta) \overset{\omega_2}{\rightarrow} (p_2', \eta_2) )</td>
</tr>
</tbody>
</table>

Table 15. Structural operational semantics of TCP$_{alt}$ for the alternative composition

ones for the alternative composition. Again, resolved races are not possible as they represent disjoint events that cannot occur simultaneously.
by the specification.

guarded recursion, enabling the solution to have the same transitions as given
no prioritized outgoing action transitions. Finally, rules 84 – 87 are standard for
other transitions. Rules 80 – 83 show the behavior of the maximal progress
It suppresses only unwanted actions, whereas it simply propagates through the

Table 16. Structural operational semantics of TCP_{sc} for the parallel composition

Rules 76 – 79 express the standard behavior for the encapsulation operator. It
suppresses only unwanted actions, whereas it simply propagates through the
other transitions. Rules 80 – 83 show the behavior of the maximal progress
operator. Rules 82 and 83 state that passage of time is enabled only if there are
no prioritized outgoing action transitions. Finally, rules 84 – 87 are standard for
guarded recursion, enabling the solution to have the same transitions as given
by the specification.

$$\begin{align*}
(64) \quad & \langle p_1, \eta \rangle \xrightarrow{a} \langle p'_1, \eta_1 \rangle, \langle p_2, \eta \rangle \xrightarrow{a_2} \langle p'_2, \eta_2 \rangle \\
(65) \quad & \langle p'_1, \eta_1 \rangle, \langle p_2, \eta \rangle \xrightarrow{\eta_1} \langle p'_1, \eta_1 \rangle, \langle p_2, \eta \rangle \xrightarrow{\eta_2} \langle p'_2, \eta_2 \rangle \\
(66) \quad & \langle p_1, \eta \rangle \xrightarrow{\eta_1} \langle p'_1, \eta_1 \rangle, \langle p_2, \eta \rangle \xrightarrow{\eta_2} \langle p'_2, \eta_2 \rangle
\end{align*}$$

$$\begin{align*}
(67) \quad & \langle p_1, \eta \rangle \xrightarrow{a_1} \langle p'_1, \eta_1 \rangle, \langle p_2, \eta \rangle \xrightarrow{a_2} \langle p'_2, \eta_2 \rangle \\
(68) \quad & \langle p'_1, \eta_1 \rangle, \langle p_2, \eta \rangle \xrightarrow{\eta_1} \langle p'_1, \eta_1 \rangle, \langle p_2, \eta \rangle \xrightarrow{\eta_2} \langle p'_2, \eta_2 \rangle \\
(69) \quad & \langle p_1, \eta \rangle \xrightarrow{a_1} \langle p'_1, \eta_1 \rangle, \langle p_2, \eta \rangle \xrightarrow{a_2} \langle p'_2, \eta_2 \rangle, \gamma(a_1, a_2) = a_3 \\
(70) \quad & \langle p_1, \eta \rangle \xrightarrow{a_1} \langle p'_1, \eta_1 \rangle, \langle p_2, \eta \rangle \xrightarrow{a_2} \langle p'_2, \eta_2 \rangle \\
(71) \quad & \langle p_1, \eta \rangle \xrightarrow{\eta_1} \langle p'_1, \eta_1 \rangle, \langle p_2, \eta \rangle \xrightarrow{\eta_2} \langle p'_2, \eta_2 \rangle \\
(72) \quad & \langle p_1, \eta \rangle \xrightarrow{\eta_1} \langle p'_1, \eta_1 \rangle, \langle p_2, \eta \rangle \xrightarrow{\eta_2} \langle p'_2, \eta_2 \rangle
\end{align*}$$

\begin{align*}
(73) \quad & \langle p_1, \eta \rangle \xrightarrow{w_1} \langle p'_1, \eta_1 \rangle, \langle p_2, \eta \rangle \xrightarrow{w_2} \langle p'_2, \eta_2 \rangle, W_1 \cap (W_2 \cup L_2) = \emptyset, \\
& \langle p_1, \eta \rangle \xrightarrow{w_1} \langle p'_1, \eta_1 \rangle, \langle p_2, \eta \rangle \xrightarrow{w_2} \langle p'_2, \eta_2 \rangle, W_1 \cup W_2 \cup L_2 = \emptyset
\end{align*}
The renaming of conflicting independent delay names is again performed by means of α-conversion, which is defined as for DTCP\textsuperscript{dst}. The bisimilarity relation is given in the same vein as for TCP\textsuperscript{rec} requiring that dependent delay names are respected.

**Definition 59.** Two terms \( p_1, p_2 \in C(TCP\textsuperscript{rec}) \) are stochastic bisimilar if there exists a bisimulation relation \( R \) such that \( (\langle p_1, \eta_0 \rangle, \langle p_2, \eta_0 \rangle, r) \in R \) for some \( r \in V \leftrightarrow V \) satisfying \( r(X) = X \) for \( X \in D(p_1) \).

As before, the definition does not impose a restriction on the use of environments because of the analogous result of Lemma 15 holds.

It should come as no surprise that stochastic bisimilarity is a congruence for TCP\textsuperscript{rec}. The proof is along the same lines as the one for Theorem 60 and, therefore, it is omitted.

**Theorem 60.** The bisimilarity relation \( \equiv_s \) is a congruence on \( C(TCP\textsuperscript{rec}) \). \(\square\)

Supported by Theorem 60 we give a term model modulo stochastic bisimilarity.

**Definition 61.** The term model of TCP\textsuperscript{rec} is the quotient algebra \( \mathcal{P}(TCP\textsuperscript{rec})/\equiv_s \), where \( \mathcal{P}(TCP\textsuperscript{rec}) = (C(TCP\textsuperscript{rec}), \delta, \epsilon, \sigma, \mu A.S) \) for \( S \in G \) and \( A \in R(S), a^- \) for \( a \in A, \pi^- \) for \( A \in \mathcal{A}, [\_]^- \) for \( W, L \subseteq V \), satisfying \( W \neq \emptyset \) and \( W \cap L = \emptyset \) for \( D \subseteq V, \partial H(.) \) for \( H \subseteq A, \partial H(.) \) for \( H \subseteq A, \_ \).
The equational theory of TCP\textsuperscript{st}\textsubscript{rec} coincides with the one of DTCP\textsuperscript{det}\textsubscript{rec}. Moreover, the main results carry over to the new setting and the theory is ground-complete in TCP\textsuperscript{st}\textsubscript{rec} as well.

Next, we compare treatment of the passage of time with the other stochastic formalisms by discussing the expansion of the parallel composition.

### 6.9 Expansion of the Parallel Composition

First, we give an abstract description of the expansion of the parallel composition in clock-based approaches [3, 4, 7, 8] that employ start-termination semantics. There, the stochastic delay \([X]\) is split on a starting \(X^+\) and an ending \(X^-\) activity, which are then treated as normal undelayable action transitions. Intuitively \([X].p = X^+.X^-q\), and the expansion of \([X].p \parallel [Y].q\) is given by

\[
\]

This allows for much more elegant expansion law, then our Theorem 47. For comparison purposes, we present the expansion of the parallel composition in SPADES, which employs clocks with residual lifetime semantics [3]. The treatment of the expansion for clocks with spent lifetimes and start-termination semantics is similar [4, 8]. In SPADES the normal form of the processes is given by \(x = \text{set } C \text{ in } x'\) and \(y = \text{set } D \text{ in } y'\), for \(x' = \sum_{i=1}^{m} (\text{when } C_i \mapsto a_i; p_i)\) and \(y' = \sum_{j=1}^{n} (\text{when } D_j \mapsto b_j; q_j)\). The operator \(\text{set}\) sets the clocks, \(\text{action prefix operator}, \text{guard that enables the process} \) when all clocks in \(C\) have expired. The expansion of \(x \parallel A y = ,\) where \(A\) is the synchronization set, is given by

\[
\text{set } (C \cup D) (\sum_{a_i \notin A \text{ when } C_i \mapsto a_i; (p_i \parallel A y')} + \sum_{b_j \notin A \text{ when } D_j \mapsto b_j; (x' \parallel A q_j)} + \sum_{a_i = b_j \in A \text{ when } (C_i \cup D_j) \mapsto a_i; (p_i \parallel A q_j)}).
\]

However, such treatment only involves the setting of the joint sets of clocks, i.e., the enabling of the starting activities. There is no relation between the passage of time of the components as in standard real-time semantics, where the expansion of \(t.p \parallel s.q\) is given by

\[
t.p \parallel s.q = \min(t, s).((t - \min(t, s)).p \parallel (s - \min(t, s)).q).
\]

As a consequence, the maximal progress operator cannot be handled explicitly as there is no knowledge about the relationship between the winners and the losers. This leads to more complicated definitions of the bisimulation relations, which must account for the priority of the internal actions [3–6].

Finally, the explicit treatment of the race condition in the stochastic transition schemes corresponds to the regional trees that are used in model checking of stochastic automata (albeit in residual lifetime semantics) [14]. Originally, the regional trees were obtained from stochastic automata [22] by explicitly ordering clock samples by their duration as symbolically represented by the stochastic delay prefix.

Next, we discuss the embedding of real time in a stochastic setting by means of Dirac stochastic delays.
6.10 Embedding Real Time as Dirac Stochastic Time

A natural embedding of real time in a stochastic setting is by means of Dirac (or degenerated) stochastic delays. These delays are guided by Dirac random variables $X_n$, where $P(X_n = n) = 1$. The Dirac delays can be included in the theory as separate stochastic delay prefixes. The duration of the Dirac delay is stated in the subscript.

Such direct inclusion of real time in the stochastic setting has a side effect, viz. the stochastic transition schemes may contain non-accessible transitions. For example, the transition

$$\langle [X_m]p + [X_{m+n}]q, \eta \rangle \xrightarrow{X_{m+n}} \langle [X_m]p + q, \eta \{X_{m+n}/\{X_m\}\} \rangle$$

will never be observed in the probabilistic timed transition system for $m, n > 0$. Similarly, the only transition with non-zero probability of $|[X_n]p|q + |[Y_n]q|q$ is the joint stochastic delay transition with winners $\{X_n, Y_n\}$. Moreover, there is need to distinguish between independent and dependent Dirac delays because of the resolution of the race condition. For example, the age of the Dirac delay $[Y_n]$ in $[X_n],[Y_n],\delta$ is dependent on the sample of the winner $[X]$. In this case, the aged distribution of $[Y_n]$ in the subterm $[Y_n],\delta$ is no longer Dirac.

Also, it should be clear that the concept of time additivity does not apply to the Dirac delays because of the race condition semantics. However, the treatment of timed delays as stochastic Dirac delays actually leads back to the notion of context-sensitive interpolation. Thus, the embedding of real time as Dirac stochastic time can be done by restricting the standard notion of time additivity by the new notion of context-sensitive interpolation. In such a setting Dirac delays support time determinism, and moreover, the side-effects from above do not occur.

We will not develop the complete embedding of real-time delays as Dirac stochastic delays, but we only give and discuss the fingerprint axioms for race-complete processes. The additional axioms for Dirac delays that enable context-sensitive interpolation are given in Table 18.

| $|\langle[X_n].p_1|q + |[X_n].p_2|q = |\langle[X_n].(\langle[p_1|q + |p_2|q]\rangle|q)$ | A44 |
| $|\langle[X_n].p_1|q + |[X_{n+m}].p_2|q = |\langle[X_n].(\langle[p_1|q + |[X_{n+m}].p_2|q]\rangle|q$ | A45 |

Table 18. Axioms for the context-sensitive interpolation of Dirac delays in race-complete process specifications

They are very similar to their real-time counterparts given by the axioms A42 and A43. However, there is an extra condition that the Dirac delays must be independent. This condition plays an important role because it ensures that the age of the Dirac delays is zero. On the contrary, it is possible that the Dirac delay is dependent on a stochastic delay as in the term $[X_n],[Y_n],\delta$ in the example above.
Next, we illustrate the features of the process theories developed so far by specifying and analyzing the CABP with real timeouts and stochastically distributed lossy channels.

7 Case-Study: CABP

In this section we specify CABP both in the process theory TCP_{drst} and in the specification language $\chi$. By restricting to deterministic timed delays, we show how to analytically obtain transient performance measures out of a $\chi$-specification based on the proposal for long-run analysis in [18]. In the general case, we exploit discrete-event simulation in $\chi$. For comparison, we perform Markovian analysis using an extension of the $\chi$ toolset\(^1\) by turning all delays into exponential ones with mean values equal to the duration of the timed delays.

7.1 Specification in TCP_{drst}_{rec}

CABP is used for communicating data along an unreliable channel with a guarantee that no information is lost. The protocol relies on retransmission of data. An overview of the CABP is depicted in Fig. 4.

Fig. 4. Scheme of the CABP

The arrival process sends the data at port 1 to the sender process $S$. The sender adds an alternating bit to the data and sends the package to receiver $R$ via the channel $K$ using port 3. It keeps re-sending the same package with a fixed time-out, waiting for the correct acknowledgement that the data has been correctly received. The channel $K$ has some probability of failure and it transfers the data with a generally distributed delay to the port 4. If the data is successfully received by $R$, then it is unpacked and the data is sent to the exit process via port 2. The alternating bit is sent as an acknowledgement back to the sender using the acknowledgement sender $AS$. The receiver $R$ communicates with $AS$ using port 5. The acknowledgement is sent via the unreliable channel $L$ using port 6. Similarly to $S$ the acknowledgement process re-sends data after a fixed time-out. The acknowledgement is communicated to the acknowledgement receiver process $AR$. If the received acknowledgement is the one expected,

\(^1\) Provided by Nikola Trčka.
then $AR$ informs the sender $S$ that it can start with the transmission of the next data package.

We can specify the CABP as below for a data set $D$. We note that the process theory does not contain an explicit probabilistic choice operator. To specify probabilistic behavior of the channel, we introduce time-outs to the channels $K$ and $L$ with duration $t_k$ and $t_l$, respectively, along the lines of Example 41. Thus, the messages are sent via the channels $K$ and $L$ before the time-out expires with a delay distributed according to the conditional random variables $(X \mid X < t_k)$ and $(X \mid X < t_k)$, respectively, or they get lost with probability $1 - F_X(t_k)$, and $1 - F_Y(t_l)$, respectively. We note that to eliminate the nondeterministic choice between $s_4$ and $r_3$ it must be that $P(X = t_k) = 0$ and $P(Y = t_l) = 0$. The CABP is specified as $\theta_1(\partial_H(S \parallel K \parallel R \parallel AS \parallel L \parallel AR))$ with

$$S = S_0$$
$$S_b = \sum_{d \in D} \tau_1(d).\sigma_i^p.\tau_5(d,b).T_{d,b}$$
$$T_{d,b} = \sigma_i^p.\tau_3(d,b).T_{d,b} + \tau_b(ack).S_{I-b}$$
$$K = \sum_{e \in D \times \{0,1\}} \tau_3(e).\theta_i([X].\tau_i(e).K + \sigma_i^L.\tau_i K)$$
$$R = R_0$$
$$R_b = \sum_{d \in D} \tau_4(d,b).\sigma_i^p.\tau_5(ack).\tau_2(d).R_{1-b} + \sum_{d \in D} \tau_4(d,1-b).R_b$$
$$AS = AS_1$$
$$AS_b = \tau_5(ack).S_0(1-b).AS_{I-b} + \sigma_i^p.\tau_6(b).AS_b$$
$$L = \sum_{b \in \{0,1\}} \tau_5(b).\theta_i([Y].\tau_i b(b).L + \sigma_i^L.\tau_i L)$$
$$AR = AR_0$$
$$AR_b = \tau_7(b).\tau_8(ack).AR_{I-b} + \tau_7(1-b).AR_b$$

where the recursive variables are parameterized by $d \in D$ and $b \in \{0,1\}$, $I = \{r_1(d), r_2(d) \mid d \in D\} \cup \{c_3(d,b), c_4(d,b) \mid b \in \{0,1\}, d \in D\} \cup \{c_6(d), c_7(d) \mid b \in \{0,1\}\} \cup \{c_8(ack), c_9(ack)\}$, and $H = \{s_3(d,b), s_4(d,b), r_3(d,b), s_4(d,b) \mid b \in \{0,1\}, d \in D\} \cup \{r_5(ack), r_6(ack), s_5(ack), s_8(ack)\}$. The deterministic timed delays with duration $t_p$, $t_s$, $t_k$, $t_r$, $t_s$, and $t_l$ represent the processing time of the sender, the time-out of the sender, the time-out of the data channel, the processing time of the receiver, the time-out of the acknowledgement sender, and the time-out of the acknowledgement channel. The internal action $i$ enables the probabilistic choices induced by the time-outs as discussed in Example 41.

### 7.2 Specification in $\chi$

We illustrate some features of the language $\chi$ by presenting the $\chi$ specification of the sender process in Fig. 5. For a detailed introduction to $\chi$ we refer the reader to [17].

74
The process `sender` communicates with the other processes via three channels: `c1, c3, c8` (see Fig. 4). The alternating bit is defined as a boolean variable and the data set is assumed to be the set of natural numbers. The sender waits for an arrival of a new data, which it packs in `tp` time units. Afterwards, a frame with the data and the alternating bit is send via channel `c3`. Here, the process enters the iterative construct represented by `*(...)` and it either resubmits the data every `ts` time units or it waits for an acknowledgement at channel `c8` from the acknowledgement receiver process. If the acknowledgement is received before the time-out expires, the process flips the alternating bit, packs the new data in `tp` time units, and sends it again via channel `c3`. Note that in the example, the processing time `tp = 1` and the time-out `ts = 10` time units.

### 7.3 Analyzing Timed Systems in the Long-Run

A proposal for analysis of timed $\chi$ specifications in the long-run was made in [18]. One considers a so-called discrete-time probabilistic reward graph, DTPRG for short, that has deterministic timed transitions and instantaneous probabilistic transitions and only one type of transitions per state allowed. The DTPRG is yielded from the timed branching bisimulation reduced $\chi$-specification and abstracting from internal transitions.

The prototype extension of $\chi$ that produces the DTPRGs uses special action transitions and self loops to simulate probabilistic choices and rewards. The pipeline for obtaining the DTPRG starts with the $\chi$ specification in which all transitions, except the ones that are meant for the probabilistic choices and the rewards, are hidden. Afterwards, rewards are added as self loops during state space generation. The timed transition system that contains internal actions is then reduced using a timed branching bisimulation to a timed system without internal actions. The special actions are then replaced by probabilistic choices to obtain the DTPRG. We note that the method is not generally sound as there may be internal transitions and actions denoted probabilistic choices originating from the same state in the original specification in which case we cannot measure.
the performance. However, here it serves its purpose in illustrating the approach. For more details we refer to [18].

Long-run performance analysis of a DTPRG is performed by translating it to a discrete-time Markov reward chain (DTMRC) with equivalent behavior. The translation from a DTPRG to a DTMRC is done in two steps. First, all timed transitions are unfolded to obtain the ‘unfolded’ DTMRC depicted in Fig. 6b. Afterwards, the (formerly immediate) probabilistic transitions are aggregated, i.e., they are eliminated and their probabilities are propagated to the original timed transitions. We illustrate the translation in Fig. 6 by an example from [18].

Fig. 6. From DTPRG to DTMRC

The DTPRG depicted in Fig. 6a has two types of states: timed (1, 2, and 3) and probabilistic (4 and 5). The translation from a DTPRG to a DTMRC is done in two steps. First, all timed transitions are unfolded to obtain the ‘unfolded’ DTMRC, as depicted in Fig. 6b. Afterwards, the (formerly immediate) probabilistic transitions are aggregated, i.e., they are eliminated and their probabilities are propagated to the timed transitions as depicted in Fig. 6. The unfolding in Fig. 6b is obtained after every timed transition of the DTPRG in Fig. 6a is represented as a sequence of probabilistic transitions with probability 1 in the DTMRC while preserving the state rewards. In the next step, an aggregation is done on the originally probabilistic transitions in the DTPRG, since in the DTPRG they do not take time, whereas in the DTMRC each transition takes one time unit. The aggregation basically splits the last transition of the unfolding of every timed transition that precedes a probabilistic transition according to the accumulative probability of reaching another timed transition as depicted in Fig. 6c. For more details and the matrix theoretic considerations underlying the problem the reader may consult [18].

The idea behind the long-run performance analysis of the DTPRG is to compute the long-run expected reward of the unfolded & aggregated DTMRC, and then to ‘fold’ it back using a collector matrix that folds back the unfolded timed transitions. The folding collector matrix sums up the states which were
obtained by the unfolding of the timed transitions. For example, the folding collector matrix for the DTMRC in Fig. 6 is \( V = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ \end{pmatrix} \) (where states 6 and 7 have been renumbered to 4 and 5 in the matrix representation).

7.4 Transient Analysis of DTPRGs

We do transient analysis at time step \( n \) of the DTPRG by using the probability transition matrix of the unfolded & aggregated DTMRC at time step \( n \). The idea is to first fold the transition matrix of the DTMRC and afterwards to compute the joint transient probability at time step \( n \) for the unfolded states of each original timed state. To fold the transition matrix we need a folding distributor matrix in addition to the folding collector matrix. The folding distributor matrix has 1s for the first states in the unfolding of the timed delays, and 0s otherwise. It supports the intuition that the process remains in a timed state until the timed delay expires by computing the joint transient probability at time step \( n \). In the example above, the folding distributor matrix is

\[
U = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ \end{pmatrix}
\]

as the starting states for unfolding of the timed delays are 1, 2, and 3.

To summarize, if the probability transition matrix at time step \( n \) of the unfolded & aggregated DTMRC is \( P(n) \), then the probability transition matrix of the DTPRG \( T(n) \) would be \( T(n) = U P(n) V \), where \( U \) and \( V \) are the folding distributor and collector matrix. We note that the lumping condition does not hold for the folding distributor and collector matrix, i.e., \( V U P(n) V \neq P(n) V \).

Therefore, the folding must be computed separately for every time step, as we did not see an obvious way of obtaining \( T(n+1) \) from \( T(n) \).

7.5 Performance Measures in a Deterministic Setting

If we assume that the distributions of the channels in the CABP are deterministic, then we can obtain its DTPRG representation and subsequently calculate its performance measures. First, we give in Fig. 7, the long-run utilization of the data channel \( K \). We assume that \( t_p = t_r = 1, t_s = t_a = 10, t_k = 6, t_l = 2 \), that the distribution of the delay of the channel \( L \) is deterministic at 6, i.e., \( P(X=6) = 1 \), and that the distribution of the delay of the channel \( K \) is deterministic at 2, i.e., \( P(Y=2) = 1 \). To obtain the utilization of the data channel, we place reward 1 for every state in the unfolding of the timed delays with duration 6, which is the delay od the data channel \( K \).

We note that, although the surface is smooth in the long-run analysis, if we observe the utilization at time step 200 we see that transient measure is not at all stable as depicted in Fig. 8.

7.6 Discrete-Event Simulation

When the channels are generally distributed we resort to discrete-event simulation in \( \chi \) for performance analysis. Fig. 9 gives the utilization of the data
channel $K$, when the distribution of the delay of the data channel is uniform between 2 and 10 and the distribution of the delay of the acknowledgement channel is uniform between 1 and 4. Thus, the uniform distributions of the data and the acknowledgement channels have the mean values of delay 6 and 2, respectively, as in the deterministic case.
Fig. 9. Utilization at 200 of channel K with uniformly distributed delays

For comparison, we also performed Markovian analysis, again by using discrete event simulation, and the result is depicted in Fig. 10. The exponential delays were chosen of the same mean values as the corresponding delays in the deterministic case.

Fig. 10. Utilization at 200 of channel K with exponentially distributed delays

Finally, to give a flavor of the results we show the dependence of the utilization of the channel K on the unreliability of the channel K at time step 200.
in Fig. 11 for each approach. Here, the unreliability of the acknowledgement channel \( L \) is fixed to 0.5. One sees that the long-run analysis using DTPRGs is close to the simulation results for the uniformly distributed channels. This is expected because they have the same mean value. As noted in [18], the Markovian analysis always underestimates the performance because the expected value of the maximum of two exponential delays is greater than the expected values of both delays, which increases the average cycle length of the system.

![Fig. 11. Utilization of the channel \( K \) at time 200 for unreliability 0.5 of the channel \( L \)](image)

8 Conclusion

We presented a process theory that enables specification of distributed systems with discrete timed and stochastic delays. The process theory axiomatizes sequential processes comprising termination, immediate actions, and timed delays in a racing context. By construction, the theory conservatively extends standard timed process algebras of [21]. We provided expansion laws for the parallel composition and the maximal progress operator. We derived delayable action and stochastic delay using timed delay prefixes and guarded recursive specifications. Using the formalism, the \( G/G/1/\infty \) queue could be handled quite conveniently.

Afterwards, we investigated the phenomenon of ‘stochastifying’ timed process theories, i.e., generalizing real-time process specifications to stochastic time. We approached the problem by adapting the real-time semantics to conform to the race condition. This lead to the new notion of context-sensitive interpolation, a restriction of time additivity when interpreted in race condition semantics. We built a stochastic process algebra from scratch that has the same equational
theory as the derived one using guarded recursive equations and timed delay prefixes.

At the end, we modeled the concurrent alternating bit protocol in the process theory and, subsequently, in the specification language $\chi$. We analyzed the protocol in the $\chi$ toolset by using discrete-event simulation for the generally distributed delays. By restricting to deterministic delays, we were able to analyze the protocol analytically with the discrete-time probabilistic reward graphs of [18]. We extended the $\chi$-environment to also deal with transient performance analysis in addition to the existing methods for long-run analysis. Finally, we performed Markovian analysis by restricting to exponential delay and we compared the results from all methods of analysis.

As future work, we plan to introduce the hiding operator that produces internal transitions and to develop a notion of branching or weak bisimulation in that setting. This should pave the way for bigger case studies Internet protocols verification and analysis as detailed performance specification is viable by using both generally distributed stochastic delays and standard timeouts. We can also exploit existing real-time specification as the theory is sufficiently flexible to allow extension of real-time with stochastic time while retaining any imposed ordering of the original delays.

References