Fischer's Protocol in Timed Process Algebra

Jan Joris Vereijken

Department of Computing Science, Eindhoven University of Technology,
P.O. Box 513, NL-5600 MB Eindhoven, The Netherlands
email: janjoris@acm.org

August 18, 1994

Abstract

Timed algebraic process theories can be developed with quite different purposes in mind. One can aim for theoretical results about the theory itself (completeness, expressiveness, decidability), or one can aim for practical applicability to non-trivial protocols. Unfortunately, these aims do not go well together. In this paper we take two theories, which are probably of the first kind, and try to find out how well suited they are for practical verifications.

We verify Fischer's protocol for mutual exclusion in the settings of discrete-time process algebra (ACP_{dt}) and real-time process algebra (ACP_{ur}). We do this by transforming the recursive specification into an equivalent linear specification, and then dividing out the maximal bisimulation relation. The required mutual exclusion result can then be found by reasoning about the obtained process graph.

Finally, we consider the ease of the verification, and ways to adapt the theory to make it more practical. It will turn out that the theories investigated are quite unsatisfactory when verifying real-life protocols.

1991 Mathematics Subject Classification: 68Q10, 68Q22, 68Q60.
1991 CR Categories: D.1.3, D.2.4, D.3.1, F.1.2, F.3.1.
Keywords: Fischer's protocol, process algebra, real time, discrete time, ACP, mutual exclusion, verification.
Note: These investigations were supported by the Netherlands Computer Science Research Foundation (SION) with financial support from the Netherlands Organisation for Scientific Research (NWO).

Contents

1 Introduction **2**

2 Fischer's protocol **3**
2.1 History of the protocol 3
2.2 Proof requirements 3
2.3 First informal description 4
2.4 Second informal description 5

3 A verification using discrete-time ACP **6**
3.1 A short note on ACP_{dt} 6
3.2 Abstraction in ACP_{dt} 7
3.3 Fischer's protocol formalized in ACP_{dt} 7
3.4 The correctness of FP_{dt} 8

4 A verification using real-time ACP **10**
4.1 A short note on ACP_{ur} 11
4.2 Abstraction operators on ACP_{ur} 11
4.3 Expansion theorems for ACP_{ur} 12
4.4 Fischer's protocol formalized in ACP_{ur} 14
4.5 The correctness of FP_{ur} 14

5 Conclusions **16**

A The linearization of FP_{dt} **18**
A.1 Methodology 18
A.2 The obtained linear system 18

B The linearization of FP_{ur} **22**
B.1 Methodology 22
B.2 The obtained linear system 23

Acknowledgments **36**

Bibliography **36**

1 Introduction

In the past decade a lot of research has been done on algebraic process theories, the three most prominent ones being CCS [Mil89], CSP [Hoa85], and ACP [BW90]. Although these theories are well established in their untimed version, there is no consensus regarding versions of CCS, CSP, and ACP extended with time.

Admitted, there have been many proposals, some of them quite successful. But most of these timed theories were geared towards theoretical results. As a consequence, a lot of papers have been published regarding completeness results, expressivity results, decidability results, and so on, but almost no paper (one exception is [Hil94]) gives an actual verification of a real system, not even a toy system.

In this paper we will present such a more-or-less real system, namely Fischer's protocol for mutual exclusion [Fis85, Lam87]. We will try to prove the correctness of this protocol, which relies heavily on time aspects, using two timed ACP theories. First, we do a verification in a setting of *discrete time*, using the theory ACP_{dt} devised by Baeten and Bergstra [BB92a]. Secondly, a verification is done in a setting of *dense time*, using the theory ACP_{ur}. This theory was devised by Klusener [Klu93], and is closely related to ACP_{ρ}, the standard real-time extension of ACP by Baeten and Bergstra [BB91].

The whole point of this exercise lies in the following two questions. First, "How well suited are the current timed extensions of ACP to verify real-life systems?", and secondly, "With real-life verifications in mind, what modifications to the theory are desirable?".

2 Fischer's protocol

In this section we give a short history of Fischer's protocol and discuss the proof requirements. After that an informal description of the protocol is given, together with an informal correctness argument.

2.1 History of the protocol

The protocol we examine is a mutual exclusion protocol, first proposed by Fischer [Fis85], and later studied in [Lam87, SBM92, AL92a, JPXZ94]. None of these studies uses process algebra to prove correctness, they all rely on some form of temporal logic or Floyd-Hoare logic. Instead of using atomic test-and-set instructions or semaphores, as is nowadays often done to assure mutual exclusion, Fischer's protocol only assumes atomic reads and writes to a shared variable (when the first mutual exclusion protocols were developed in the late 1960s all exclusion protocols were of the "shared variable kind" [Dij65, Knu66, dB67, Lam74], later on researchers have more concentrated on the "semaphore kind" of protocol). Mutual exclusion in Fischer's Protocol is guaranteed by carefully placing bounds on the execution times of the instructions, leading to a protocol which is very simple, and relies heavily on time aspects. This makes it an ideal candidate for the purpose we have in mind, namely to try to verify (using process algebra) a not too difficult protocol which still has quite intricate timing aspects.

2.2 Proof requirements

What does one need to prove in the case of a mutual exclusion protocol? Strange as is it may seem, this is not all that clear. In the literature one sees requirements like:

- The actual property of mutual exclusion: only one component may be in its critical section at any time,
- Symmetry between the components,
- No assumptions about the execution times of statements (obviously not satisfied in our case!),
- Liveness: there should always be some process that is able to proceed,
- No starvation: it may not be that a component is permanently prohibited from entering its critical section,
- Various kinds of fairness: a component should get its fair share (in various senses) of being allowed to proceed into its critical section,
- Loosely connectedness: when one component deadlocks (outside its critical section), this should not affect the progress of the other components,

- Minimal overhead: the protocol should make a decision as soon it has enough information to do so,

and even more requirements. Some of these requirements are related (for example: symmetry guarantees most kinds of fairness), and each paper about mutual exclusion seems to have its own favorite subset of which ones to prove.

In the case of Fischer's protocol we choose, mostly following the earliest paper on mutual exclusion [Dij65], to prove the following three properties:

- Actual mutual exclusion between the two critical sections,
- Symmetry between the two components,
- No starvation.

We will not try to formalize these properties algebraically, as they do not lend themselves easily to this. This is more due to the shortcomings of the (current) algebraic approach than it is to unwillingness on our part; note for example that the above properties *can* indeed be very easily formalized using temporal logic. We will return to this subject in our conclusions.

2.3 First informal description

We will now describe the protocol in an informal way, giving an informal correctness argument. Assume the existence of a shared variable x, to which atomic reads and writes are possible. Initially x equals zero. In Figure 1 we give Fischer's protocol expressed in pseudo-PASCAL. There are two components, running in parallel. The angle brackets ("⟨", "⟩") denote atomicity.

Component 1:

```
repeat
    repeat
        await ⟨x = 0⟩;
        ⟨x := 1⟩;
        ⟨delay⟩;
    until ⟨x = 1⟩;
    critical section 1;
    ⟨x := 0⟩;
until false;
```

Component 2:

```
repeat
    repeat
        await ⟨x = 0⟩;
        ⟨x := 2⟩;
        ⟨delay⟩;
    until ⟨x = 2⟩;
    critical section 2;
    ⟨x := 0⟩;
until false;
```

Figure 1: Fischer's protocol, first informal version.

The protocol proceeds as follows. Initially, the value of the shared variable is 0. When component 1 observes that x is 0, it will write the value 1 to x. After that, it waits for some time, and if x then still has the value 1, it is safe to enter the critical section. Component 2 works in a similar way (using 2 instead of 1), and both components run in parallel.

The mutual exclusion property of the protocol is based on the following observation. The *delay* operation causes component 1 to wait sufficiently long so that, if component 2 had read the value of x in its **await** statement before the component 1 executed its $x \coloneqq 1$ assignment, then component 2 will have completed the following $x \coloneqq 2$ statement. Therefore, it can never happen that component 1 falls through its **until** statement, entering *critical section 1*, while component 2 is still about to execute its $x \coloneqq 2$ assignment. This guarantees mutual exclusion. By symmetry, the argument also holds the other way around.

2.4 Second informal description

Let us try to make the reasoning from the previous section a bit more solid by exactly indicating the possible durations of the statements. First of all, the **await** statement may take anywhere between 0 and ∞ time units after x becomes 0. The assignments $x \coloneqq i$ are supposed to take between a and a' time units, and the *delay* statements between d and d' time units, for fixed non-negative values $a \leq a'$ and $d \leq d'$ over some totally-ordered time domain. Furthermore, assume that $a' < d$, i.e. the delay always takes longer than an assignment. For simplicity sake, the read actions $x = i$ are supposed to take 0 time units, and the critical section may take any time, including 0 time units. Writing $\langle \mathit{action} \rangle_t^{t'}$ for an atomic action that happens between t and t' time units after it has been enabled, we arrive at the protocol of Figure 2.

Component 1:

repeat
  repeat
    await $\langle x = 0 \rangle_0^{\infty}$;
    $\langle x \coloneqq 1 \rangle_a^{a'}$;
    $\langle \mathit{delay} \rangle_d^{d'}$;
  until $\langle x = 1 \rangle_0^{0}$;
  critical section 1;
  $\langle x \coloneqq 0 \rangle_a^{a'}$;
until false;

Component 2:

repeat
  repeat
    await $\langle x = 0 \rangle_0^{\infty}$;
    $\langle x \coloneqq 2 \rangle_a^{a'}$;
    $\langle \mathit{delay} \rangle_d^{d'}$;
  until $\langle x = 2 \rangle_0^{0}$;
  critical section 2;
  $\langle x \coloneqq 0 \rangle_a^{a'}$;
until false;

Figure 2: Fischer's protocol, second informal version.

Remember we assumed that $0 \leq a \leq a' < d \leq d' < \infty$. Now we have that if component 2 falls through its **await** statement, it will complete its $x \coloneqq 2$ assignment within at most a' time units. If component 1 would have happened to complete its $x \coloneqq 1$ assignment just after component 2 fell through its **await**, it will take component 1 at least d time units to complete its *delay*. As $a' < d$, when component 1 reaches the **until** $\langle x = 1 \rangle$ statement, component 2 will have completed its $x \coloneqq 2$ assignment. Therefore, the value of x has stabilized, and component 2 can safely enter its critical section.

As a final remark: note that Fischer's protocol can be trivially generalized to any number $n > 2$ of components. This generalization, however, we will not examine.

3 A verification using discrete-time ACP

In this section we will prove the correctness of the protocol of Figure 2 on the preceding page for the special case where $a = a' = 0$ and $d = d' = 1$. We will use (a subset of) the discrete-time process algebra $\mathrm{ACP_{dt}}$ as described in [BB92a]. This is the simplest special case one can imagine. The resulting proof provides a clear illustration of all key issues, without becoming too cluttered with technicalities.

3.1 A short note on $\mathrm{ACP_{dt}}$

First, we briefly describe the essential extensions that $\mathrm{ACP_{dt}}$ has to model time. Intuitively, $\mathrm{ACP_{dt}}$ is very much like plain ACP [BW90], with the exception that a process may have the ability to "let time pass for one time unit". If in an $\mathrm{ACP_{dt}}$ process some component still wants to do an action, it can, and no time passes. If however all components agree to let time pass, one unit of time will pass (in other words: the current time slice ends, or the clock ticks).

This willingness to let time pass is expressed with the *discrete time unit delay* operator σ_d. The expression $\sigma_d(X)$ denotes the process that can do no action anymore in the current time slice, but has the ability to let time pass for one time unit. If that unit of time finally "happens" (time does not really pass in a discrete setting, it just happens!), the process $\sigma_d(X)$ turns into the process X. This is denoted by $\sigma_d(X) \xrightarrow{\sigma} X$, as if σ were a special action denoting a tick of the clock.

In Table 1 the operational semantics of $\mathrm{BPA_{dt}}$ is given. $\mathrm{BPA_{dt}}$ is the subset of $\mathrm{ACP_{dt}}$ that only has $\cdot$, $+$, and σ_d as operators. For the full axiomatization and semantics of $\mathrm{ACP_{dt}}$ see [BB92a]. Besides the σ_d operator, we will only use one more special operator

$$a \xrightarrow{a} \surd$$

$$\frac{x \xrightarrow{a} x'}{x \cdot y \xrightarrow{a} x' \cdot y} \qquad \frac{x \xrightarrow{a} \surd}{x \cdot y \xrightarrow{a} y}$$

$$\frac{x \xrightarrow{a} x'}{x + y \xrightarrow{a} x',\ y + x \xrightarrow{a} x'} \qquad \frac{x \xrightarrow{a} \surd}{x + y \xrightarrow{a} \surd,\ y + x \xrightarrow{a} \surd}$$

$$\sigma_d(x) \xrightarrow{\sigma} x \qquad \frac{x \xrightarrow{\sigma} x'}{x \cdot y \xrightarrow{\sigma} x' \cdot y}$$

$$\frac{x \xrightarrow{\sigma} x',\ y \xrightarrow{\sigma} y'}{x + y \xrightarrow{\sigma} x' + y'} \qquad \frac{x \xrightarrow{\sigma} x',\ y \not\xrightarrow{\sigma}}{x + y \xrightarrow{\sigma} x',\ y + x \xrightarrow{\sigma} x'}$$

Table 1: Operational semantics of $\mathrm{BPA_{dt}}$.

from $\mathrm{ACP_{dt}}$, namely the *unbounded start delay* of a process X, denoted by $\lfloor X \rfloor^{\omega}$. The expression $\lfloor X \rfloor^{\omega}$ is the process that can do exactly the same things that X can, and it is

also always willing to let time pass. This is shown in the operational semantics in Table 2.

$$\frac{x \xrightarrow{a} x'}{\lfloor x \rfloor^{\omega} \xrightarrow{a} x'} \qquad \frac{x \xrightarrow{a} \surd}{\lfloor x \rfloor^{\omega} \xrightarrow{a} \surd} \qquad \lfloor x \rfloor^{\omega} \xrightarrow{\sigma} \lfloor x \rfloor^{\omega}$$

Table 2: Operational semantics of the unbounded start delay.

3.2 Abstraction in $\mathrm{ACP_{dt}}$

Abstraction from internal actions in $\mathrm{ACP_{dt}}$ is defined as expected. It is axiomized as shown in Table 3. If A denotes the set of all possible actions, we have that $I \subseteq A$, $a \in A \cup \{\delta, \tau\}$, and $\sigma \notin A$. The variables x and y denote processes.

$\tau_I(a) = a$	if $a \notin I$	TI1
$\tau_I(a) = \tau$	if $a \in I$	TI2
$\tau_I(x + y) = \tau_I(x) + \tau_I(y)$		TI3
$\tau_I(x \cdot y) = \tau_I(x) \cdot \tau_I(y)$		TI4
$\tau_I(\sigma_d(x)) = \sigma_d(\tau_I(x))$		TI8

Table 3: Axioms for the abstraction operator τ_I in $\mathrm{ACP_{dt}}$

There exists an operator τ_σ that abstracts from time steps, but we do not describe it here because we will not need it. The non-consecutive numbering of the axioms is due to historical reasons.

3.3 Fischer's protocol formalized in $\mathrm{ACP_{dt}}$

Having introduced the relevant operators for $\mathrm{ACP_{dt}}$, we are now ready to give a formal specification of Fischer's protocol $\mathrm{FP_{dt}}$, using $\mathrm{ACP_{dt}}$ in Figure 3 on the following page. As said before, we specify the special case where $a = a' = 0$ and $d = d' = 1$.

This specification can be understood intuitively in the following way. There are three processes running concurrently, namely A, B, and V. The processes A and B model "Component 1" and "Component 2" of Figure 1 respectively, and the process V models the variable x.

The process V can be in one of three states: V_0, V_1, or V_2, corresponding to the possible values of x. In any state V_i, V is capable of sending the message $x = i$, signaling that the value of x is currently i, after which V will continue in state V_i. Furthermore, V is in any state V_i capable of receiving the message $x := j$ for any j, indicating that x is being assigned with the value j. After such an assignment V will continue in state V_j. Finally, V is always capable of letting time pass. The process V constructed in this way behaves as a "variable server": if process A or B wants to assign a value i to x it performs the action

$$
\begin{aligned}
A &= A_0 & B &= B_0\\
A_0 &= \lfloor r(x=0) \rfloor^{\omega} \cdot A_1 & B_0 &= \lfloor r(x=0) \rfloor^{\omega} \cdot B_1\\
A_1 &= s(x \coloneqq 1) \cdot A_2 & B_1 &= s(x \coloneqq 2) \cdot B_2\\
A_2 &= \sigma_d(A_3) & B_2 &= \sigma_d(B_3)\\
A_3 &= (r(x=0) + r(x=2)) \cdot A_0 + r(x=1) \cdot A_4 & B_3 &= (r(x=0) + r(x=1) \cdot B_0 + r(x=2) \cdot B_4\\
A_4 &= \mathrm{EnterCS}_1 \cdot A_5 & B_4 &= \mathrm{EnterCS}_2 \cdot B_5\\
A_5 &= \mathrm{LeaveCS}_1 \cdot A_6 & B_5 &= \mathrm{LeaveCS}_2 \cdot B_6\\
A_6 &= s(x \coloneqq 0) \cdot A_0 & B_6 &= s(x \coloneqq 0) \cdot B_0
\end{aligned}
$$

$$
\begin{aligned}
V &= V_0\\
V_0 &= (r(x \coloneqq 0) + s(x=0)) \cdot V_0 + r(x \coloneqq 1) \cdot V_1 + r(x \coloneqq 2) \cdot V_2 + \sigma_d(V_0)\\
V_1 &= (r(x \coloneqq 1) + s(x=1)) \cdot V_1 + r(x \coloneqq 0) \cdot V_0 + r(x \coloneqq 2) \cdot V_2 + \sigma_d(V_1)\\
V_2 &= (r(x \coloneqq 2) + s(x=2)) \cdot V_2 + r(x \coloneqq 0) \cdot V_0 + r(x \coloneqq 1) \cdot V_1 + \sigma_d(V_2)
\end{aligned}
$$

$$\gamma(r(\alpha), s(\alpha)) = c(\alpha) \text{ for } \alpha \in \{ x = i, x \coloneqq i \mid i \in \{0,1,2\} \}$$

$$H = \{ r(\alpha), s(\alpha) \mid \alpha \in \{ x = i, x \coloneqq i \mid i \in \{0,1,2\} \} \}$$

$$\mathrm{FP}_{\mathrm{dt}} = \partial_H(A \parallel B \parallel V)$$

Figure 3: Fischer's protocol in discrete time ($\mathrm{ACP}_{\mathrm{dt}}$).

$s(x \coloneqq i)$. If it wants to check if x has the value i, it performs the action $r(x = i)$. (The idea to construct the variable server in this way was taken from [Nie90].)

The process A is constructed as follows. First (in state A_0) it waits for an undetermined amount of time till x is 0 ($\lfloor r(x=0) \rfloor^{\omega}$). Then (in state A_1) it sets x to 1 ($s(x \coloneqq 1)$). After that (in state A_2), it waits till the end of the time slice ($\sigma_d(A_3)$). When it has arrived in state A_3, it will examine the contents of x, and either jump back to A_0 (if $x = 0$ or $x = 2$), or continue with state A_4 (if $x = 1$). Thereafter, it enters its critical section, leaves it again, and resets x back to 0 in state A_6, after which it repeats the whole procedure over again. The process B is constructed in the same way as A is. The entire protocol, $\mathrm{FP}_{\mathrm{dt}}$, now consists of the processes A, B, and V running concurrently. Note that the assignment takes no time ($a = a' = 0$) and the delay takes one time unit ($d = d' = 1$).

3.4 The correctness of $\mathrm{FP}_{\mathrm{dt}}$

In order to prove the protocol $\mathrm{FP}_{\mathrm{dt}}$ correct, we first rewrite the recursive equations of Figure 3 into an equivalent *linear* system of equations (i.e. one that does not contain the operators $\parallel$, ∂_H, or τ_I anymore). We arrive at the system of 32 equations given in Appendix A (where also the details of the linearization are given). Using this linear system we then construct the process graph of $\mathrm{FP}_{\mathrm{dt}}$ given in Figure 4 on the next page (Note that all $c(\ldots)$ edges remain unlabelled, as labelling them would obscure the picture, and that $\mathrm{EnterCS}_1$ is abbreviated to E1. The other Enter and Leave actions are abbreviated in a

similar way).

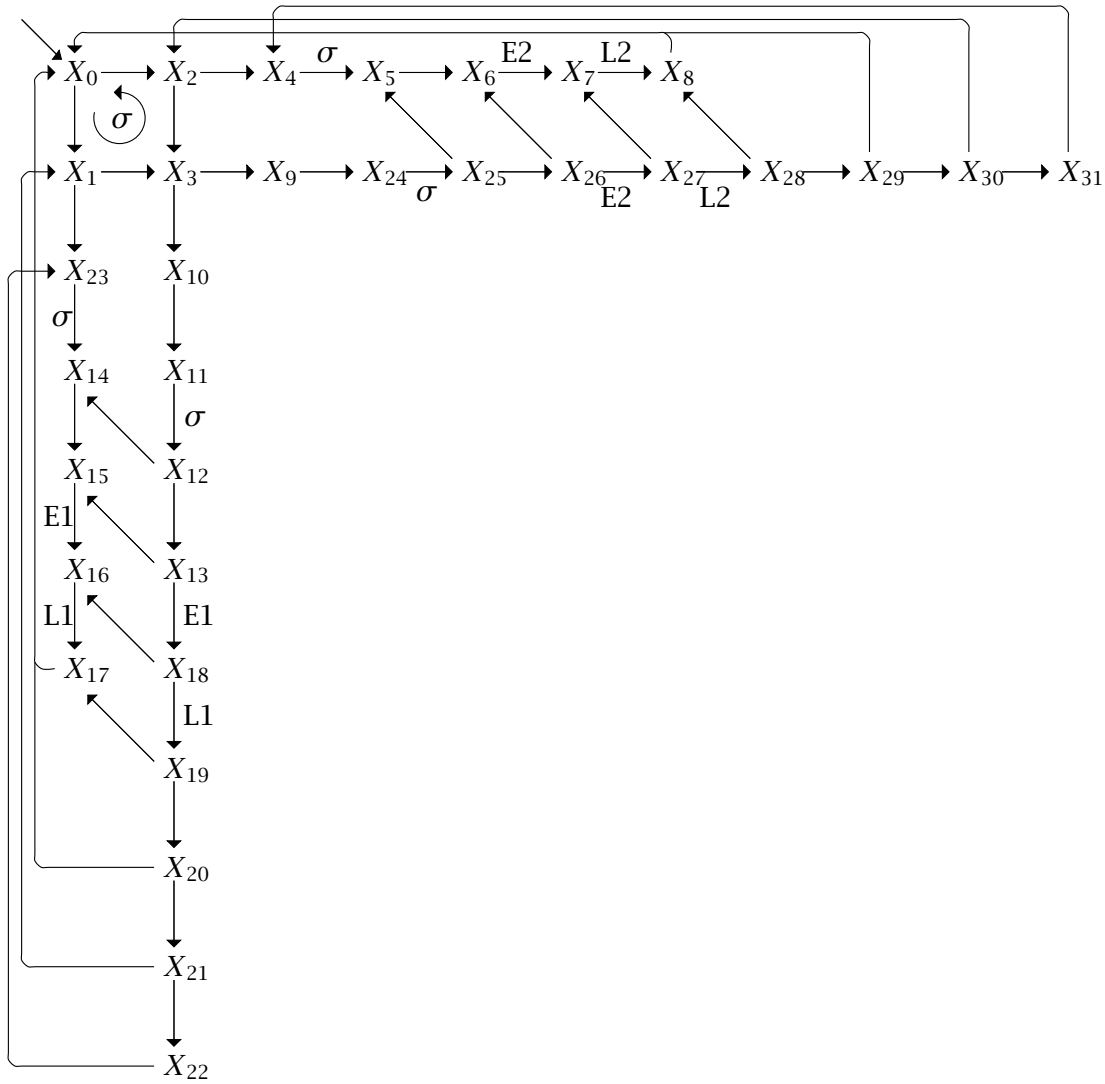

Figure 4: The process graph of $\mathrm{FP}_{\mathrm{dt}}$.

Now define the set I of internal actions as:

$$I = \{\, c(\alpha) \mid \alpha \in \{\, x = i, x \coloneqq i \mid i \in \{0,1,2\} \,\} \,\}$$

(i.e. all communication actions) and rename these actions into τ, yielding the process graph of $\tau_I(\mathrm{FP}_{\mathrm{dt}})$. On this graph we compute the maximal rooted branching auto bisimulation, which gives us the equivalence classes $X_A, \ldots, X_H$ given below:

$$\begin{aligned}
X_A &= \{X_0, X_8, X_{17}, X_{19}, X_{20}, X_{28}, X_{29}\} & X_E &= \{X_{16}, X_{18}\} \\
X_B &= \{X_1, X_2, X_3, X_{21}, X_{30}\} & X_F &= \{X_4, X_9, X_{24}, X_{31}\} \\
X_C &= \{X_{10}, X_{11}, X_{22}, X_{23}\} & X_G &= \{X_5, X_6, X_{25}, X_{26}\} \\
X_D &= \{X_{12}, X_{13}, X_{14}, X_{15}\} & X_H &= \{X_7, X_{27}\}
\end{aligned}$$

When we divide out this equivalence relation we arrive at the reduced process graph given in Figure 5. On this graph we will now be able to check all required properties very easily.

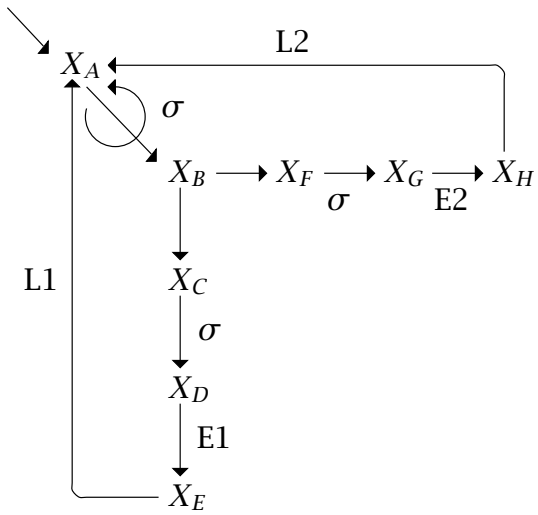

Figure 5: The reduced process graph of $\tau_I(\mathrm{FP}_{\mathrm{dt}})$.

The three requirements for mutual exclusion we chose in Section 2.2 are satisfied for the following reasons:

- **Actual mutual exclusion:** It can be easily seen from the graph of Figure 5 that an $\mathrm{EnterCS}_1$ action is always immediately followed by a $\mathrm{LeaveCS}_1$ action, and the same holds for $\mathrm{EnterCS}_2$ and $\mathrm{LeaveCS}_2$. Therefore, it cannot be the case that both components are in their critical section at the same time.

- **Symmetry:** As the graph is symmetrical with respect to the paths from the root, through the critical sections of the components, back to the root, it is clear that the the protocol is symmetrical with respect to the components.

- **No starvation:** Whichever state the protocol is in, there is always a path leading to each component's critical section. Therefore, using fairness, it cannot be the case that one component is permanently prohibited from entering its critical section.

This completes the proof of the correctness of $\mathrm{FP}_{\mathrm{dt}}$.

4 A verification using real-time ACP

In this section we will prove the correctness of the protocol of Figure 2, for the almost completely general case where $0 < a < a' < d < d' < 2a$ (this is only a very minor restriction of the general case, but it reduces the state-space significantly). We will use (a subset of) the real-time process algebra $\mathrm{ACP}_{\mathrm{ur}}$ ("ACP with **u**rgent actions and **r**elative time") as described in [Klu93].

4.1 A short note on $\mathrm{ACP_{ur}}$

Again, we briefly describe the essential extensions that $\mathrm{ACP_{ur}}$ has to model time. To begin with, every action is postfixed with a non-negative real number between square brackets. One could have, for example, $a[3]$, meaning "the process that performs the action a at exactly 3 time units from now". If the parameter is 0, e.g. $a[0]$, it means "execute the action immediately". Sequential composition and the choice operator behave as expected. So, for example, the process $a[2] \cdot b[3]$ started at time 4 does a at time 6 and b at time 9. The process $a[2] + b[3]$ started at time 2 can either do a at time 4 or b at time 5; if at any moment after 4 the action a has not been observed, one knows that the choice has been made for b at time 5. While this is all very intuitive, the exact semantics of $\mathrm{ACP_{ur}}$ is still quite intricate, and we will make no attempt at making it formal here. Please see [Klu93] and [BB91].

The single most important construct in $\mathrm{ACP_{ur}}$ is the *integral construct*. It is a generalized sum over an interval of time. For example, $\int_{v=1}^{2} a[v]$ denotes the process that can do the action a at any time between 1 and 2 time units from now, inclusively. The upper bound may be infinity: $\int_{v=0}^{\infty} a$ denotes the process that can do a once at some time, including right now, and all moments arbitrarily far into the future. (Please note that the integral construct as described here is a rather crude simplification of the one given by Klusener. This simplified one however is all we need.)

For the special cases where for an integral $\int_{v=p}^{q} P(v)$ we do not have $0 \leq p \leq q$, we will use the convention that:

$$\int_{v=p}^{q} P(v) = \begin{cases} \int_{v=0}^{q} P(v) & \text{if } p \leq 0 \text{ and } q \geq 0 \\ \delta[0] & \text{if } q < 0 \text{ or } p > q \end{cases}$$

This means that every integral can be written in a normal form in the following way.

$$\int_{v=p}^{q} P(v) = (0 \leq q \text{ and } p \leq q) :\rightarrow \int_{v=\max(0,p)}^{q} P(v)$$

Here $:\rightarrow$ is used to form a so called *guarded command*. For a boolean expression ϕ and a process P the guarded command is defined, dependent on the current valuation V of the free variables, as:

$$\phi :\rightarrow P = \begin{cases} P & \text{if } [\phi \equiv \text{true}]_V \\ \delta[0] & \text{otherwise} \end{cases}$$

This normal form for integrals will be called *guard-prefixed normal form*. The usefulness of this normal form lies in the fact that is immediate clear from the guard expression whether or not the integral is empty. This will be important when defining abstraction operators.

4.2 Abstraction operators on $\mathrm{ACP_{ur}}$

We define two abstraction operators. First, a *general abstraction operator*:

$$\tau_I : \mathrm{ACP_{ur}} \rightarrow \mathrm{ACP_{ur}}$$

that renames actions in I into τ-actions, and secondly, a *time abstraction operator*:

$$\tau_t : \mathrm{ACP_{ur}} \to \mathrm{ACP}$$

that transforms a (timed) $\mathrm{ACP_{ur}}$-term into an (untimed) ACP-term by "throwing away all timing information". The general abstraction operator is axiomized as shown in Table 4. We have that $I \subseteq A$, $a \in A \cup \{\delta, \tau\}$, $\nu \in \mathbb{R}^{\geq 0}$, and ϕ a boolean expression that may contain free variables. The variables x and y denote processes.

$\tau_I(a[\nu])$	$= a[\nu]$	if $a \notin I$	TIT1
$\tau_I(a[\nu])$	$= \tau[\nu]$	if $a \in I$	TIT2
$\tau_I(x + y)$	$= \tau_I(x) + \tau_I(y)$		TIT3
$\tau_I(x \cdot y)$	$= \tau_I(x) \cdot \tau_I(y)$		TIT4
$\tau_I(\phi :\to x)$	$= \phi :\to \tau_I(x)$		TIT5

Table 4: Axioms for the general abstraction operator τ_I in $\mathrm{ACP_{ur}}$

The time abstraction operator is axiomized as shown in Table 5. We have that $a \in A \cup \{\delta, \tau\}$, ϕ a boolean expression that may contain free variables, and $p, q, \nu \in \mathbb{R}^{\geq 0}$ with $p \leq q$. The variables x and y denote processes, and $\mathsf{FV}(x)$ stands for the set of all free variables x contains.

$\tau_t(a[\nu])$	$= a$		TTT1
$\tau_t(x + y)$	$= \tau_t(x) + \tau_t(y)$		TTT2
$\tau_t(x \cdot y)$	$= \tau_t(x) \cdot \tau_t(y)$		TTT3
$\tau_t(\int_{\nu=p}^{q} a[\nu])$	$= a$		TTT4
$\tau_t(\int_{\nu=p}^{q} a[\nu] \cdot x)$	$= a \cdot \tau_t(x)$	if $\nu \notin \mathsf{FV}(x)$	TTT5
$\tau_t(\int_{\nu=p}^{q} a[\nu] \cdot x)$	$= a \cdot \tau_t(\int_{\nu=p}^{q} x)$	if $\nu \in \mathsf{FV}(x)$	TTT6
$\tau_t(\phi :\to x)$	$= \tau_t(x)$	if $\exists_V[\phi \equiv \mathrm{true}]_V$	TTT7

Table 5: Axioms for the time abstraction operator τ_t in $\mathrm{ACP_{ur}}$

Axiom TTT7, where V denotes a valuation of the free variables, is formulated this way because we do not want that $\tau_t(\phi :\to x)$ would equal $\tau_t(x)$ if there is no way to satisfy ϕ. On the other hand, we want that $\tau_t(\phi :\to x)$ does equal $\tau_t(x)$ if ϕ can be satisfied by some valuation of the free variables, even if there are other valuations that do not satisfy ϕ.

4.3 Expansion theorems for $\mathrm{ACP_{ur}}$

Before we are able to do the actual verification, we first need to have an expansion theorem, which becomes quite intricate as real time gets involved. We start by giving a version that assumes there is no communication at all:

The expansion theorem for $\mathrm{ACP}_{\mathrm{ur}}$ (without communication):

$$\mathop{\|}_{i=1,\ldots,n} \int_{\nu_i=p_i}^{q_i} a_i[\nu_i] \cdot P_i(\nu_i)$$
$$=$$
$$\sum_{i=1,\ldots,n} \int_{\nu_i=p_i}^{\min\{\, q_j \mid j=1,\ldots,n \,\}} a_i[\nu_i] \cdot \left(P_i(\nu_i) \parallel \mathop{\|}_{\substack{j=1,\ldots,n \\ j\neq i}} \int_{\nu_j=p_j-\nu_i}^{q_j-\nu_i} a_j[\nu_j] \cdot P_j(\nu_j) \right)$$

This theorem can be understood in the following way. We have n processes of the form $\int_{\nu_i=p_i}^{q_i} a_i[\nu_i] \cdot P_i(\nu_i)$ running in parallel, where a_i is an action, and $P_i(\nu_i)$ is an arbitrary process expression in ν_i. The action a_i is enabled in the interval $[p_i, q_i]$ from the start of the process. How does this big merge of n processes expand? Obviously, the first thing that is going to happen is some action a_i, so we sum over all those. At what time can this action happen? No earlier than the time when it gets enabled, but also not later than the minimum of the latest times of all other actions. This explains the bounds p_i and $\min\{\, q_j \mid j = 1, \ldots, n \,\}$ on the integral preceeding a_i. What remains is $P_i(\nu_i)$ running in parallel with the other components, albeit that ν_i time units have passed; hence the $p_j - \nu_i$ and $q_j - \nu_i$ bounds on the inner integrals. To make this somewhat clearer we give the special case where $n = 2$:

$$\left(\int_{\nu=p}^{q} a[\nu] \cdot P(\nu) \right) \parallel \left(\int_{w=r}^{s} b[w] \cdot Q(w) \right)$$
$$=$$
$$\int_{\nu=p}^{\min(q,s)} a[\nu] \cdot \left(P(\nu) \parallel \int_{w=r-\nu}^{s-\nu} b[w] \cdot Q(w) \right) + \int_{w=r}^{\min(q,s)} b[w] \cdot \left(\int_{\nu=p-w}^{q-w} a[\nu] \cdot P(\nu) \parallel Q(w) \right)$$

When we allow handshaking (communication with the restriction that $a \mid b \mid c$ is always δ for all actions a, b, and c) we arrive at the following theorem:

The expansion theorem for $\mathrm{ACP}_{\mathrm{ur}}$ (with handshaking):

$$\mathop{\|}_{i=1,\ldots,n} \int_{\nu_i=p_i}^{q_i} a_i[\nu_i] \cdot P_i(\nu_i)$$
$$=$$
$$\sum_{i=1,\ldots,n} \int_{\nu_i=p_i}^{\min\{\, q_j \mid j=1,\ldots,n \,\}} a_i[\nu_i] \cdot \left(P_i(\nu_i) \parallel \mathop{\|}_{\substack{j=1,\ldots,n \\ j\neq i}} \int_{\nu_j=p_j-\nu_i}^{q_j-\nu_i} a_j[\nu_j] \cdot P_j(\nu_j) \right) +$$
$$\sum_{1\leq i<j\leq n} \int_{\nu=\max(p_i,p_j)}^{\min\{\, q_j \mid j=1,\ldots,n \,\}} (a_i \mid a_j)[\nu] \cdot \left(P_i(\nu) \parallel P_j(\nu) \parallel \mathop{\|}_{\substack{k=1,\ldots,n \\ k\neq i, k\neq j}} \int_{\nu_k=p_k-\nu}^{q_k-\nu} a_k[\nu_k] \cdot P_k(\nu_k) \right)$$

This theorem can be understood in much the same way as the previous one, although this time there are extra summands, corresponding to all possible (necessarily two-way)

communications. At what time can such a communication $(a_i \mid a_j)$ take place? The first possible time is when both a_i and a_j are enabled, so we get $\max(p_i, p_j)$ as the lower bound of the integral. The upper bound is $\min\{\, q_j \mid j = 1, \ldots, n \,\}$, just as it was in the previous theorem. Here, the special case for $n = 2$ is as follows:

$$\left(\int_{v=p}^{q} a[v] \cdot P(v)\right) \parallel \left(\int_{w=r}^{s} b[w] \cdot Q(w)\right)$$
$$=$$
$$\int_{v=p}^{\min(q,s)} a[v] \cdot \left(P(v) \parallel \int_{w=r-v}^{s-v} b[w] \cdot Q(w)\right) +$$
$$\int_{w=r}^{\min(q,s)} b[w] \cdot \left(\int_{v=p-w}^{q-w} a[v] \cdot P(v) \parallel Q(w)\right) +$$
$$\int_{u=\max(p,r)}^{\min(q,s)} (a \mid b)[u] \cdot (P(u) \parallel Q(u))$$

We will not give proofs for the above theorems, as that would go beyond the scope of the paper. (Our goal is to find out which problems arise when *applying* existing theories, and consequently, which adaptions to these theories are desirable. It is not our goal to prove theorems in these theories.)

4.4 Fischer's protocol formalized in $\mathrm{ACP_{ur}}$

We are now ready to give a formally specified implementation of Fischer's protocol $\mathrm{FP_{ur}}$, using $\mathrm{ACP_{ur}}$ in Figure 6 on the next page. The protocol, $\mathrm{FP_{ur}}$, consists just like in the discrete-time case of three concurrent processes: A, B, and V. The process V models the variable x in such a way that assignments and reads can take place at any moment. The process A is also constructed much the same as in discrete time; the only difference being the execution time interval bounds on all actions. In state A_0 we do a $\int_{v=0}^{\infty} r(x = 0)[v]$, meaning that we will wait at least till x is 0, but possibly longer. After that, by performing $\int_{v=a}^{a'} s(x := 1)[v]$ we assign the value 1 to x somewhere within the interval $[a, a']$, and so on. The values a, a', d, and d' are positive constants such that $a < a' < d < d' < 2a$. Note that this time the "delay actions" are explicit modelled: i_1 and i_2.

4.5 The correctness of $\mathrm{FP_{ur}}$

Using the material from Sections 4.1 and 4.3 we are now able to rewrite the specification for $\mathrm{FP_{ur}}$ in Figure 6 into an equivalent linear system of equations. The way to do this, however, is far more complex than it was in the case of $\mathrm{FP_{dt}}$; see Appendix B for full details.

We ultimately arrive at the linear system of 51 equations given in Section B.2. Unfortunately, the process graph of $\mathrm{FP_{ur}}$ induced by this linearization is to complex to draw it without obscuring most of the crucial parts, so we will not give it here. Continuing with the verification, we define the set of internal actions as:

$$I = \{\, c(\alpha) \mid \alpha \in \{\, x = i, x := i \mid i \in \{0, 1, 2\} \,\} \,\} \cup \{i_1, i_2\}$$

$$
\begin{aligned}
A &= A_0 & B &= B_0 \\
A_0 &= \int_{\nu=0}^{\infty} r(x=0)[\nu] \cdot A_1 & B_0 &= \int_{\nu=0}^{\infty} r(x=0)[\nu] \cdot B_1 \\
A_1 &= \int_{\nu=a}^{a'} s(x \coloneqq 1)[\nu] \cdot A_2 & B_1 &= \int_{\nu=a}^{a'} s(x \coloneqq 2)[\nu] \cdot B_2 \\
A_2 &= \int_{\nu=d}^{d'} i_1[\nu] \cdot A_3 & B_2 &= \int_{\nu=d}^{d'} i_2[\nu] \cdot B_3 \\
A_3 &= (r(x=0)[0] + r(x=2)[0]) \cdot A_0 + & B_3 &= (r(x=0)[0] + r(x=1)[0]) \cdot B_0 + \\
&\quad r(x=1)[0] \cdot A_4 & &\quad r(x=2)[0] \cdot B_4 \\
A_4 &= \mathrm{EnterCS}_1[0] \cdot A_5 & B_4 &= \mathrm{EnterCS}_2[0] \cdot B_5 \\
A_5 &= \int_{\nu=0}^{\infty} \mathrm{LeaveCS}_1[\nu] \cdot A_6 & B_5 &= \int_{\nu=0}^{\infty} \mathrm{LeaveCS}_2[\nu] \cdot B_6 \\
A_6 &= \int_{\nu=a}^{a'} s(x \coloneqq 0)[\nu] \cdot A_0 & B_6 &= \int_{\nu=a}^{a'} s(x \coloneqq 0)[\nu] \cdot B_0
\end{aligned}
$$

$$
\begin{aligned}
V &= V_0 \\
V_0 &= \int_{\nu=0}^{\infty} (r(x \coloneqq 0)[\nu] + s(x=0)[\nu]) \cdot V_0 + r(x \coloneqq 1)[\nu] \cdot V_1 + r(x \coloneqq 2)[\nu] \cdot V_2 \\
V_1 &= \int_{\nu=0}^{\infty} (r(x \coloneqq 1)[\nu] + s(x=1)[\nu]) \cdot V_1 + r(x \coloneqq 0)[\nu] \cdot V_0 + r(x \coloneqq 2)[\nu] \cdot V_2 \\
V_2 &= \int_{\nu=0}^{\infty} (r(x \coloneqq 2)[\nu] + s(x=2)[\nu]) \cdot V_2 + r(x \coloneqq 0)[\nu] \cdot V_0 + r(x \coloneqq 1)[\nu] \cdot V_1
\end{aligned}
$$

$$\gamma(r(\alpha), s(\alpha)) = c(\alpha) \text{ for } \alpha \in \{\, x = i, x \coloneqq i \mid i \in \{0,1,2\} \,\}$$

$$H = \{\, r(\alpha), s(\alpha) \mid \alpha \in \{\, x = i, x \coloneqq i \mid i \in \{0,1,2\} \,\} \,\}$$

$$\mathrm{FP}_{\mathrm{ur}} = \partial_H(A \parallel B \parallel V)$$

Figure 6: Fischer's protocol in real time ($\mathrm{ACP}_{\mathrm{ur}}$).

i.e. all communication actions, and i_1 and i_2 (the "*delay*" actions). As we have made sure that all guarded commands in the linearization can be satisfied, it is very easy to compute $\tau_t \circ \tau_I(\mathrm{FP}_{\mathrm{ur}})$. On the process graph of $\tau_t \circ \tau_I(\mathrm{FP}_{\mathrm{ur}})$ we then compute the maximal rooted branching auto bisimulation, which gives us the following equivalence classes:

$$
\begin{aligned}
X_A &= \{X_0, X_1, X_2, X_3, X_8, X_{19}, X_{22}, X_{25}, X_{26}, X_{27}, X_{36}, X_{37}, X_{38}, X_{39}, X_{42}\} \\
X_B &= \{X_{10}, X_{11}, X_{12}, X_{13}, X_{14}, X_{15}, X_{16}, X_{17}, X_{20}, X_{23}, X_{41}, X_{44}, X_{45}, X_{46}, X_{47}\} \\
X_C &= \{X_{18}, X_{21}, X_{24}\} \\
X_D &= \{X_4, X_5, X_6, X_9, X_{28}, X_{29}, X_{30}, X_{31}, X_{32}, X_{33}, X_{40}, X_{43}, X_{48}, X_{49}, X_{50}\} \\
X_E &= \{X_7, X_{34}, X_{35}\}
\end{aligned}
$$

When we divide out this equivalence relation, we arrive at the reduced process graph given in Figure 7. On this graph we can now check all required properties, in almost

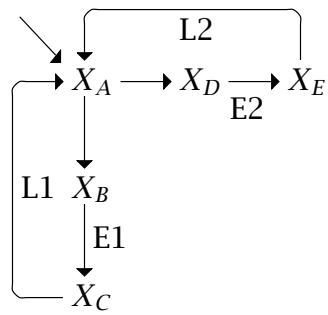

Figure 7: The reduced process graph of $\tau_t \circ \tau_I(\mathrm{FP}_{\mathrm{ur}})$.

exactly the same way as we did earlier for $\tau_I(\mathrm{FP}_{\mathrm{dt}})$:

- **Actual mutual exclusion:** It can be easily seen from the graph of Figure 7 that an $\mathrm{EnterCS}_1$ action is always immediately followed by a $\mathrm{LeaveCS}_1$ action, and the same holds for $\mathrm{EnterCS}_2$ and $\mathrm{LeaveCS}_2$. Therefore, it cannot be the case that both components are in their critical section at the same time.
- **Symmetry:** As the graph is symmetrical with respect to the paths from the root, through the critical sections of the components, back to the root, it is clear that the the protocol is symmetrical with respect to the components.
- **No starvation:** Whichever state the protocol is in, there is always a path leading to each component's critical section. Therefore, using fairness, it cannot be the case that one component is permanently prohibited from entering its critical section.

We find that all properties hold, which completes the proof of the correctness of $\mathrm{FP}_{\mathrm{ur}}$.

5 Conclusions

As we have seen Fischer's protocol can be proven correct quite satisfactorily using algebraic techniques. This however does not mean too much; it is an almost trivial protocol,

that has been solved time and time again using all kinds of formalisms. But it is not all that bad either; see for example [SBM92] where an (incomplete) proof is given of Fischer's protocol. When it would be written out in full detail, that proof would be about as long and tedious as ours is [Sch94], and the same probably holds for a detailed temporal logic proof [Aba94].

Looking at our proof one observes that, although conceptually very clear and easy, the inner workings required a lot of bothersome and failure-prone computations. It seems valid to doubt whether all these calculations were really necessary. Much of their complexity results from the fact that the theories we used are based on bisimulation semantics, which preserves very many moments of choice, while that was not at all required for the proof we were constructing. Furthermore, the so-called "algebraic advantage", i.e. the ability to calculate with processes without having to write out the entire state space, is almost absent. This is disappointing; although the algebraic advantage manifests itself very clearly when verifying protocols that do not exhibit much internal parallelism, such as the Alternating Bit Protocol (see for example [BW90]), it seems to be lost in the verification of Fischer's protocol, which has very much internal parallelism. As a result, we get the worst of both worlds: the state explosion from naive model checking, and the complicated term rewriting from real-time process algebra ...

Let us however not become too pessimistic. It has become clear that algebraic theories can indeed be applied to protocols with intricate real-time aspects. It would be unrealistic to expect that these theories, which were designed without much regard for their practical application, would in their unmodified form be splendidly suited for real-life verifications. There is a lot of room for tuning these theories towards more practicality. We see at least three directions in which we would like to proceed, preferably in all three at the same time.

- First of all, it would be nice to have a real-time process algebra that does not lean on bisimulation semantics only. It is simply not always a good idea to preserve all internal moments of choice. If we had a theory based on, say, ready semantics or failure semantics, we would probably gain some of our algebraic advantage back.

  But maybe abandoning bisimulation semantics altogether would be too crude. A more sophisticated and subtle approach could be to abstract only from those internal moments of choice we really want to abstract from. This could be implemented by introducing a special choice operator next to the ordinary $+$. For example, the delayed choice of [BM94] or a τ-angelic choice.

- Secondly, it might be profitable to augment process algebra with a (limited) form of temporal logic. Looking at the linearization process of FP_{ur} in Appendix B, it is clear that much of the calculations involved are needed to manage the exact ranges of the parameters of the variables. Or, on a conceptually higher level: the calculations get complicated because we do not have adequate tools for denoting the precise flow of time. When working with a hybrid process algebra-temporal logic theory (still predominantly algebraic!), these complications would probably not have arisen. See for example [BBB93], where this approach is investigated.

- Thirdly, we might just as well admit that real-time verifications are difficult, and probably will remain so for some years to come. Therefore, it may be advisable

to have computer tools at our disposal. One could for example imagine an "process algebra calculator" which, like an ordinary arithmetic calculator, could assist in performing complex calculations. This way we could put the human back into the driver's seat of the verification process, instead of having him stumble in the dark while juggling complex process terms. Assuming we can indeed exploit our algebraic advantage, this approach could be very interesting. A combination of easy process term manipulation and powerful algebraic techniques could lead to verifications that are both short and easy to construct.

Concluding, we can say that the verification of Fischer's protocol by algebraic means as given in this paper is not very straightforward. But nonetheless, there are several promising directions for future research that could lead to new, algebraically oriented theories that may perform much better for real-life verifications.

A The linearization of $\mathrm{FP}_{\mathrm{dt}}$

A.1 Methodology

Finding a linear specification for $\mathrm{FP}_{\mathrm{dt}}$ is very easy (at least in principle): one just repeatedly applies the expansion theorem for $\mathrm{ACP}_{\mathrm{dt}}$. All new states reached are numbered consecutively using process variables X_i. Linearizing in a depth-first fashion we arrive at the 32 processes $X_0, \ldots, X_{31}$ given below in Section A.2. The protocol $\mathrm{FP}_{\mathrm{dt}}$ itself corresponds to the variable X_0.

Please note that the equations in the following section were produced by hand, and therefore initially had several mistakes in them. To attempt to find these, we have performed several computerized sanity checks on the hand-generated data file. For example, it is obvious that the entire system is symmetrical with respect to the A and B component, and this can be quite easily checked. This means that if there are any mistakes left, it must be that we have made exactly the same mistake two times over, in a symmetrical way. Some more checks were made, exploiting similar intuitively known facts about the process graph, and by old-fashioned human proofreading.

Finally, the data file was automatically converted to "TEX-format" by means of a computer, in order to minimize the chances of mistakes in that step of the process.

A.2 The obtained linear system

$$\begin{aligned}
\mathrm{FP}_{\mathrm{dt}} &= \partial_H(A \parallel B \parallel V) \\
&= \partial_H(A_0 \parallel B_0 \parallel V_0) \\
&= X_0
\end{aligned}$$

$$\begin{aligned}
X_0 &= \partial_H(A_0 \parallel B_0 \parallel V_0) \\
&= c(x=0) \cdot \partial_H(A_1 \parallel B_0 \parallel V_0) + c(x=0) \cdot \partial_H(A_0 \parallel B_1 \parallel V_0) + \\
&\quad \sigma_d(\partial_H(A_0 \parallel B_0 \parallel V_0)) \\
&= c(x=0) \cdot X_1 + c(x=0) \cdot X_2 + \sigma_d(X_0)
\end{aligned}$$

$$
\begin{aligned}
X_1 &= \partial_H(A_1 \parallel B_0 \parallel V_0) \\
&= c(x \coloneqq 1) \cdot \partial_H(A_2 \parallel B_0 \parallel V_1) + c(x = 0) \cdot \partial_H(A_1 \parallel B_1 \parallel V_0) \\
&= c(x \coloneqq 1) \cdot X_{23} + c(x = 0) \cdot X_3
\end{aligned}
$$

$$
\begin{aligned}
X_2 &= \partial_H(A_0 \parallel B_1 \parallel V_0) \\
&= c(x = 0) \cdot \partial_H(A_1 \parallel B_1 \parallel V_0) + c(x \coloneqq 2) \cdot \partial_H(A_0 \parallel B_2 \parallel V_2) \\
&= c(x = 0) \cdot X_3 + c(x \coloneqq 2) \cdot X_4
\end{aligned}
$$

$$
\begin{aligned}
X_3 &= \partial_H(A_1 \parallel B_1 \parallel V_0) \\
&= c(x \coloneqq 1) \cdot \partial_H(A_2 \parallel B_1 \parallel V_1) + c(x \coloneqq 2) \cdot \partial_H(A_1 \parallel B_2 \parallel V_2) \\
&= c(x \coloneqq 1) \cdot X_9 + c(x \coloneqq 2) \cdot X_{10}
\end{aligned}
$$

$$
\begin{aligned}
X_4 &= \partial_H(A_0 \parallel B_2 \parallel V_2) \\
&= \sigma_d(\partial_H(A_0 \parallel B_3 \parallel V_2)) \\
&= \sigma_d(X_5)
\end{aligned}
$$

$$
\begin{aligned}
X_5 &= \partial_H(A_0 \parallel B_3 \parallel V_2) \\
&= c(x = 2) \cdot \partial_H(A_0 \parallel B_4 \parallel V_2) \\
&= c(x = 2) \cdot X_6
\end{aligned}
$$

$$
\begin{aligned}
X_6 &= \partial_H(A_0 \parallel B_4 \parallel V_2) \\
&= \mathrm{EnterCS}_2 \cdot \partial_H(A_0 \parallel B_5 \parallel V_2) \\
&= \mathrm{EnterCS}_2 \cdot X_7
\end{aligned}
$$

$$
\begin{aligned}
X_7 &= \partial_H(A_0 \parallel B_5 \parallel V_2) \\
&= \mathrm{LeaveCS}_2 \cdot \partial_H(A_0 \parallel B_6 \parallel V_2) \\
&= \mathrm{LeaveCS}_2 \cdot X_8
\end{aligned}
$$

$$
\begin{aligned}
X_8 &= \partial_H(A_0 \parallel B_6 \parallel V_2) \\
&= c(x \coloneqq 0) \cdot \partial_H(A_0 \parallel B_0 \parallel V_0) \\
&= c(x \coloneqq 0) \cdot X_0
\end{aligned}
$$

$$
\begin{aligned}
X_9 &= \partial_H(A_2 \parallel B_1 \parallel V_1) \\
&= c(x \coloneqq 2) \cdot \partial_H(A_2 \parallel B_2 \parallel V_2) \\
&= c(x \coloneqq 2) \cdot X_{24}
\end{aligned}
$$

$$
\begin{aligned}
X_{10} &= \partial_H(A_1 \parallel B_2 \parallel V_2) \\
&= c(x \coloneqq 1) \cdot \partial_H(A_2 \parallel B_2 \parallel V_1) \\
&= c(x \coloneqq 1) \cdot X_{11}
\end{aligned}
$$

$$
\begin{aligned}
X_{11} &= \partial_H(A_2 \parallel B_2 \parallel V_1) \\
&= \sigma_d(\partial_H(A_3 \parallel B_3 \parallel V_1)) \\
&= \sigma_d(X_{12})
\end{aligned}
$$

$$
\begin{aligned}
X_{12} &= \partial_H(A_3 \parallel B_3 \parallel V_1) \\
&= c(x = 1) \cdot \partial_H(A_4 \parallel B_3 \parallel V_1) + c(x = 1) \cdot \partial_H(A_3 \parallel B_0 \parallel V_1) \\
&= c(x = 1) \cdot X_{13} + c(x = 1) \cdot X_{14}
\end{aligned}
$$

$$
\begin{aligned}
X_{13} &= \partial_H(A_4 \parallel B_3 \parallel V_1) \\
&= \mathrm{EnterCS}_1 \cdot \partial_H(A_5 \parallel B_3 \parallel V_1) + c(x = 1) \cdot \partial_H(A_4 \parallel B_0 \parallel V_1) \\
&= \mathrm{EnterCS}_1 \cdot X_{18} + c(x = 1) \cdot X_{15}
\end{aligned}
$$

$$
\begin{aligned}
X_{14} &= \partial_H(A_3 \parallel B_0 \parallel V_1) \\
&= c(x = 0) \cdot \partial_H(A_4 \parallel B_0 \parallel V_1) \\
&= c(x = 0) \cdot X_{15}
\end{aligned}
$$

$$
\begin{aligned}
X_{15} &= \partial_H(A_4 \parallel B_0 \parallel V_1) \\
&= \mathrm{EnterCS}_1 \cdot \partial_H(A_5 \parallel B_0 \parallel V_1) \\
&= \mathrm{EnterCS}_1 \cdot X_{16}
\end{aligned}
$$

$$
\begin{aligned}
X_{16} &= \partial_H(A_5 \parallel B_0 \parallel V_1) \\
&= \mathrm{LeaveCS}_1 \cdot \partial_H(A_6 \parallel B_0 \parallel V_1) \\
&= \mathrm{LeaveCS}_1 \cdot X_{17}
\end{aligned}
$$

$$
\begin{aligned}
X_{17} &= \partial_H(A_6 \parallel B_0 \parallel V_1) \\
&= c(x \coloneqq 0) \cdot \partial_H(A_0 \parallel B_0 \parallel V_0) \\
&= c(x \coloneqq 0) \cdot X_0
\end{aligned}
$$

$$
\begin{aligned}
X_{18} &= \partial_H(A_5 \parallel B_3 \parallel V_1) \\
&= \mathrm{LeaveCS}_1 \cdot \partial_H(A_6 \parallel B_3 \parallel V_1) + c(x = 1) \cdot \partial_H(A_5 \parallel B_0 \parallel V_1) \\
&= \mathrm{LeaveCS}_1 \cdot X_{19} + c(x = 1) \cdot X_{16}
\end{aligned}
$$

$$
\begin{aligned}
X_{19} &= \partial_H(A_6 \parallel B_3 \parallel V_1) \\
&= c(x \coloneqq 0) \cdot \partial_H(A_0 \parallel B_3 \parallel V_0) + c(x = 1) \cdot \partial_H(A_6 \parallel B_0 \parallel V_1) \\
&= c(x \coloneqq 0) \cdot X_{20} + c(x = 1) \cdot X_{17}
\end{aligned}
$$

$$
\begin{aligned}
X_{20} &= \partial_H(A_0 \parallel B_3 \parallel V_0) \\
&= c(x = 0) \cdot \partial_H(A_1 \parallel B_3 \parallel V_0) + c(x = 0) \cdot \partial_H(A_0 \parallel B_0 \parallel V_0) \\
&= c(x = 0) \cdot X_{21} + c(x = 0) \cdot X_0
\end{aligned}
$$

$$
\begin{aligned}
X_{21} &= \partial_H(A_1 \parallel B_3 \parallel V_0) \\
&= c(x \coloneqq 1) \cdot \partial_H(A_2 \parallel B_3 \parallel V_1) + c(x = 0) \cdot \partial_H(A_1 \parallel B_0 \parallel V_0) \\
&= c(x \coloneqq 1) \cdot X_{22} + c(x = 0) \cdot X_1
\end{aligned}
$$

$$
\begin{aligned}
X_{22} &= \partial_H(A_2 \parallel B_3 \parallel V_1) \\
&= c(x = 1) \cdot \partial_H(A_2 \parallel B_0 \parallel V_1) \\
&= c(x = 1) \cdot X_{23}
\end{aligned}
$$

$$
\begin{aligned}
X_{23} &= \partial_H(A_2 \parallel B_0 \parallel V_1) \\
&= \sigma_d(\partial_H(A_3 \parallel B_0 \parallel V_1)) \\
&= \sigma_d(X_{14})
\end{aligned}
$$

$$
\begin{aligned}
X_{24} &= \partial_H(A_2 \parallel B_2 \parallel V_2) \\
&= \sigma_d(\partial_H(A_3 \parallel B_3 \parallel V_2)) \\
&= \sigma_d(X_{25})
\end{aligned}
$$

$$
\begin{aligned}
X_{25} &= \partial_H(A_3 \parallel B_3 \parallel V_2) \\
&= c(x = 2) \cdot \partial_H(A_0 \parallel B_3 \parallel V_2) + c(x = 2) \cdot \partial_H(A_3 \parallel B_4 \parallel V_2) \\
&= c(x = 2) \cdot X_5 + c(x = 2) \cdot X_{26}
\end{aligned}
$$

$$
\begin{aligned}
X_{26} &= \partial_H(A_3 \parallel B_4 \parallel V_2) \\
&= c(x = 2) \cdot \partial_H(A_0 \parallel B_4 \parallel V_2) + \mathrm{EnterCS}_2 \cdot \partial_H(A_3 \parallel B_5 \parallel V_2) \\
&= c(x = 2) \cdot X_6 + \mathrm{EnterCS}_2 \cdot X_{27}
\end{aligned}
$$

$$
\begin{aligned}
X_{27} &= \partial_H(A_3 \parallel B_5 \parallel V_2) \\
&= c(x = 2) \cdot \partial_H(A_0 \parallel B_5 \parallel V_2) + \mathrm{LeaveCS}_2 \cdot \partial_H(A_3 \parallel B_6 \parallel V_2) \\
&= c(x = 2) \cdot X_7 + \mathrm{LeaveCS}_2 \cdot X_{28}
\end{aligned}
$$

$$
\begin{aligned}
X_{28} &= \partial_H(A_3 \parallel B_6 \parallel V_2) \\
&= c(x = 2) \cdot \partial_H(A_0 \parallel B_6 \parallel V_2) + c(x \coloneqq 0) \cdot \partial_H(A_3 \parallel B_0 \parallel V_0) \\
&= c(x = 2) \cdot X_8 + c(x \coloneqq 0) \cdot X_{29}
\end{aligned}
$$

$$
\begin{aligned}
X_{29} &= \partial_H(A_3 \parallel B_0 \parallel V_0) \\
&= c(x = 0) \cdot \partial_H(A_0 \parallel B_0 \parallel V_0) + c(x = 0) \cdot \partial_H(A_3 \parallel B_1 \parallel V_0) \\
&= c(x = 0) \cdot X_0 + c(x = 0) \cdot X_{30}
\end{aligned}
$$

$$
\begin{aligned}
X_{30} &= \partial_H(A_3 \parallel B_1 \parallel V_0) \\
&= c(x = 0) \cdot \partial_H(A_0 \parallel B_1 \parallel V_0) + c(x \coloneqq 2) \cdot \partial_H(A_3 \parallel B_2 \parallel V_2) \\
&= c(x = 0) \cdot X_2 + c(x \coloneqq 2) \cdot X_{31}
\end{aligned}
$$

$$
\begin{aligned}
X_{31} &= \partial_H(A_3 \parallel B_2 \parallel V_2) \\
&= c(x = 2) \cdot \partial_H(A_0 \parallel B_2 \parallel V_2) \\
&= c(x = 2) \cdot X_4
\end{aligned}
$$

B The linearization of $\mathrm{FP}_{\mathrm{ur}}$

B.1 Methodology

Finding a linear specification for $\mathrm{FP}_{\mathrm{ur}}$ is a lot more difficult than it was in the discrete-time case. This is caused by the fact that most of the variables used in the linear recursive system are parameterized with a time stamp $v \in \mathbb{R}^{\geq 0}$. So in order to perform calculations on these parameterized variables, a lot of bookkeeping has to be done. There are three things that have to be watched:

- First, when introducing new parameterized variables, one has to compute the range the parameter might take. The range will always take the form of a non-empty closed interval $[t, t'] \subseteq \mathbb{R}^{\geq 0}$. It is important that all points of this interval can actually be reached by the protocol.

  For example, when expanding $X_3(v)$ (where $0 \leq v \leq a'$) on page 24, we need to introduce new parameterized variables X_9 and X_{10}. The summand for X_9 is:

  $$(v \leq a' - a) :\rightarrow \int_{u=a}^{a'-v} c(x \coloneqq 1)[u] \cdot X_9(u + v)$$

  To determine the range the parameter of X_9 can take, we first observe that although $0 \leq v \leq a'$, the guarded command $(v \leq a' - a) :\rightarrow \ldots$ effectively limits this range to $0 \leq v \leq a' - a$. As u can range from a to $a' - v$, this means that $u + v$ ranges from a (when $v = 0$ and $u = a$) to a' (when $u = a' - v$, and v is anything between 0 and $a' - a$). So, if we substitute the fresh variable v' for $u + v$, the parameter range for $X_9(v')$ is $a \leq v' \leq a'$. This range is indicated just above the definition of X_9, on page 25.

- Secondly, two parameterized variables that both represent the same subprocess, but differ in the range which their parameter may take, have to be considered different variables, and get different names.

  This happens for example between variables $X_{15}(v)$ and and $X_{47}(v)$. Both are defined as:

  $$\partial_H \left(\int_{w=d-v}^{d'-v} i_1[w] \cdot A_3 \parallel B_0 \parallel V_1 \right)$$

  but for $X_{15}(v)$ we have $d - a' \leq v \leq d'$, while for $X_{47}(v)$ we have $d - a' + a \leq v \leq d'$.

- Thirdly, at some points in the calculation it is crucial that the information provided by the range of a parameter be used to simplify the equations involved. Often this manifests itself as a guarded command evaluating to false. *These steps are the essence of the verification:* the paths that are cut off because the guarded command never evaluates to true represent paths that would violate the safety properties, such as actual mutual exclusion, we are trying to prove.

  This can be observed for example in the expansion of $X_9(v)$, where $a \leq v \leq a'$. There we arrive at a summand:

  $$(v \leq a' - d) :\rightarrow \int_{u=d}^{a'-v} i_1[u] \cdot \partial_H \left(A_2 \parallel \int_{w=a-v-u}^{a'-v-u} s(x \coloneqq 2)[w] \cdot B_2 \parallel V_1 \right)$$

  Because $a' < d$ (intuitively: a *delay* always takes longer than an assignment), $a' - d < 0$. As $0 < a \leq v$ we have $v > a' - d$, so the condition is always false. Therefore, the whole summand vanishes into δ. This is just what we need, because choosing that summand would lead us onto a path where both components could enter the critical section simultaneously.

  At this point, it becomes clear why we put in the extra restriction $d' < 2a$ (see page 10). Although this restriction is irrelevant for the correctness of the protocol, it does ensure that $d' - d < a$. This greatly reduces the size of the linear specification, as it intuitively cuts off all paths where the one component is delaying "forever" while the other component is repeatedly entering and leaving its critical section. See the expansion of $X_{26}(v)$ for an example of this.

After linearizing in a depth-first fashion we arrive at the 51 processes $X_0, \ldots, X_{50}$ given in Section B.2. The protocol $\mathrm{FP}_{\mathrm{ur}}$ itself corresponds to the variable X_0. We have made sure to indicate exactly the range the parameters may take. Furthermore, we ultimately put all integrals in guard-prefixed normal form.

As in the discrete-time case, several computerized sanity checks were made on the hand-generated data. This time we also had the opportunity to exploit the information provided by the parameter ranges. Translating the data file into "TEX-format" was again done automatically, although some final manual editing was necessary to achieve a satisfactory layout.

B.2 The obtained linear system

$\mathrm{FP}_{\mathrm{ur}} = \partial_H(A \parallel B \parallel V)$

$$
\begin{aligned}
&= \partial_H(A_0 \parallel B_0 \parallel V_0) \\
&= X_0
\end{aligned}
$$

$$
\begin{aligned}
X_0 &= \partial_H(A_0 \parallel B_0 \parallel V_0) \\
&= \int_{v=0}^{\infty} c(x=0)[v] \cdot \partial_H(A_1 \parallel B_0 \parallel V_0) + \int_{v=0}^{\infty} c(x=0)[v] \cdot \partial_H(A_0 \parallel B_1 \parallel V_0) \\
&= \int_{v=0}^{\infty} c(x=0)[v] \cdot X_1 + \int_{v=0}^{\infty} c(x=0)[v] \cdot X_2 \\
&= (\text{true}) :\rightarrow \int_{v=0}^{\infty} c(x=0)[v] \cdot X_1 + (\text{true}) :\rightarrow \int_{v=0}^{\infty} c(x=0)[v] \cdot X_2
\end{aligned}
$$

$$
\begin{aligned}
X_1 &= \partial_H(A_1 \parallel B_0 \parallel V_0) \\
&= \int_{v=a}^{a'} c(x \coloneqq 1)[v] \cdot \partial_H(A_2 \parallel B_0 \parallel V_1) + \\
&\quad \int_{v=0}^{a'} c(x=0)[v] \cdot \partial_H\left(\int_{w=a-v}^{a'-v} s(x \coloneqq 1)[w] \cdot A_2 \parallel B_1 \parallel V_0\right) \\
&= \int_{v=a}^{a'} c(x \coloneqq 1)[v] \cdot X_{41} + \int_{v=0}^{a'} c(x=0)[v] \cdot X_{42}(v) \\
&= \int_{v=a}^{a'} c(x \coloneqq 1)[v] \cdot X_{41} + \int_{v=0}^{a'} c(x=0)[v] \cdot X_{42}(v) \\
&= (\text{true}) :\rightarrow \int_{v=a}^{a'} c(x \coloneqq 1)[v] \cdot X_{41} + (\text{true}) :\rightarrow \int_{v=0}^{a'} c(x=0)[v] \cdot X_{42}(v)
\end{aligned}
$$

$$
\begin{aligned}
X_2 &= \partial_H(A_0 \parallel B_1 \parallel V_0) \\
&= \int_{v=0}^{a'} c(x=0)[v] \cdot \partial_H\left(A_1 \parallel \int_{w=a-v}^{a'-v} s(x \coloneqq 2)[w] \cdot B_2 \parallel V_0\right) + \\
&\quad \int_{v=a}^{a'} c(x \coloneqq 2)[v] \cdot \partial_H(A_0 \parallel B_2 \parallel V_2) \\
&= \int_{v=0}^{a'} c(x=0)[v] \cdot X_3(v) + \int_{v=a}^{a'} c(x \coloneqq 2)[v] \cdot X_4 \\
&= (\text{true}) :\rightarrow \int_{v=0}^{a'} c(x=0)[v] \cdot X_3(v) + (\text{true}) :\rightarrow \int_{v=a}^{a'} c(x \coloneqq 2)[v] \cdot X_4
\end{aligned}
$$

For all v, $0 \leq v \leq a'$:

$$
\begin{aligned}
X_3(v) &= \partial_H\left(A_1 \parallel \int_{w=a-v}^{a'-v} s(x \coloneqq 2)[w] \cdot B_2 \parallel V_0\right) \\
&= \int_{u=a}^{\min(a',a'-v)} c(x \coloneqq 1)[u] \cdot \partial_H\left(A_2 \parallel \int_{w=a-v-u}^{a'-v-u} s(x \coloneqq 2)[w] \cdot B_2 \parallel V_1\right) + \\
&\quad \int_{w=a-v}^{\min(a',a'-v)} c(x \coloneqq 2)[w] \cdot \partial_H\left(\int_{u=a-w}^{a'-w} s(x \coloneqq 1)[u] \cdot A_2 \parallel B_2 \parallel V_2\right) \\
&= \int_{u=a}^{a'-v} c(x \coloneqq 1)[u] \cdot \partial_H\left(A_2 \parallel \int_{w=a-v-u}^{a'-v-u} s(x \coloneqq 2)[w] \cdot B_2 \parallel V_1\right) + \\
&\quad \int_{w=a-v}^{a'-v} c(x \coloneqq 2)[w] \cdot \partial_H\left(\int_{u=a-w}^{a'-w} s(x \coloneqq 1)[u] \cdot A_2 \parallel B_2 \parallel V_2\right)
\end{aligned}
$$

$$
\begin{aligned}
&= \int_{u=a}^{a'-v} c(x \coloneqq 1)[u] \cdot X_9(u+v) + \int_{w=a-v}^{a'-v} c(x \coloneqq 2)[w] \cdot X_{10}(w) \\
&= (v \leq a'-a) :\rightarrow \int_{u=a}^{a'-v} c(x \coloneqq 1)[u] \cdot X_9(u+v) + \\
&\quad (\text{true}) :\rightarrow \int_{w=\max(0,a-v)}^{a'-v} c(x \coloneqq 2)[w] \cdot X_{10}(w)
\end{aligned}
$$

$$
\begin{aligned}
X_4 &= \partial_H(A_0 \parallel B_2 \parallel V_2) \\
&= \int_{v=d}^{d'} i_2[v] \cdot \partial_H(A_0 \parallel B_3 \parallel V_2) \\
&= \int_{v=d}^{d'} i_2[v] \cdot X_5 \\
&= (\text{true}) :\rightarrow \int_{v=d}^{d'} i_2[v] \cdot X_5
\end{aligned}
$$

$$
\begin{aligned}
X_5 &= \partial_H(A_0 \parallel B_3 \parallel V_2) \\
&= c(x=2)[0] \cdot \partial_H(A_0 \parallel B_4 \parallel V_2) \\
&= c(x=2)[0] \cdot X_6
\end{aligned}
$$

$$
\begin{aligned}
X_6 &= \partial_H(A_0 \parallel B_4 \parallel V_2) \\
&= \text{EnterCS}_2[0] \cdot \partial_H(A_0 \parallel B_5 \parallel V_2) \\
&= \text{EnterCS}_2[0] \cdot X_7
\end{aligned}
$$

$$
\begin{aligned}
X_7 &= \partial_H(A_0 \parallel B_5 \parallel V_2) \\
&= \int_{v=0}^{\infty} \text{LeaveCS}_2[v] \cdot \partial_H(A_0 \parallel B_6 \parallel V_2) \\
&= \int_{v=0}^{\infty} \text{LeaveCS}_2[v] \cdot X_8 \\
&= (\text{true}) :\rightarrow \int_{v=0}^{\infty} \text{LeaveCS}_2[v] \cdot X_8
\end{aligned}
$$

$$
\begin{aligned}
X_8 &= \partial_H(A_0 \parallel B_6 \parallel V_2) \\
&= \int_{v=a}^{a'} c(x \coloneqq 0)[v] \cdot \partial_H(A_0 \parallel B_0 \parallel V_0) \\
&= \int_{v=a}^{a'} c(x \coloneqq 0)[v] \cdot X_0 \\
&= (\text{true}) :\rightarrow \int_{v=a}^{a'} c(x \coloneqq 0)[v] \cdot X_0
\end{aligned}
$$

For all v, $a \leq v \leq a'$:

$$
\begin{aligned}
X_9(v) &= \partial_H\left(A_2 \parallel \int_{w=a-v}^{a'-v} s(x \coloneqq 2)[w] \cdot B_2 \parallel V_1\right) \\
&= \int_{u=d}^{\min(d',a'-v)} i_1[u] \cdot \partial_H\left(A_2 \parallel \int_{w=a-v-u}^{a'-v-u} s(x \coloneqq 2)[w] \cdot B_2 \parallel V_1\right) +
\end{aligned}
$$

$$
\begin{aligned}
&\int_{w=a-v}^{\min(d',a'-v)} c(x := 2)[w] \cdot \partial_H \left(\int_{u=d-w}^{d'-w} i_1[u] \cdot A_3 \parallel B_2 \parallel V_2 \right) \\
= &\int_{u=d}^{a'-v} i_1[u] \cdot \partial_H \left(A_2 \parallel \int_{w=a-v-u}^{a'-v-u} s(x := 2)[w] \cdot B_2 \parallel V_1 \right) + \\
&\int_{w=a-v}^{a'-v} c(x := 2)[w] \cdot \partial_H \left(\int_{u=d-w}^{d'-w} i_1[u] \cdot A_3 \parallel B_2 \parallel V_2 \right) \\
= &\int_{u=d}^{a'-v} i_1[u] \cdot \partial_H(\dots) + \int_{w=a-v}^{a'-v} c(x := 2)[w] \cdot X_{28}(w) \\
= &(v \leq a' - d) :\rightarrow \int_{u=d}^{a'-v} i_1[u] \cdot \partial_H(\dots) + \\
&(\text{true}) :\rightarrow \int_{w=0}^{a'-v} c(x := 2)[w] \cdot X_{28}(w) \\
= &(\text{false}) :\rightarrow \int_{u=d}^{a'-v} i_1[u] \cdot \partial_H(\dots) + (\text{true}) :\rightarrow \int_{w=0}^{a'-v} c(x := 2)[w] \cdot X_{28}(w) \\
= &(\text{true}) :\rightarrow \int_{w=0}^{a'-v} c(x := 2)[w] \cdot X_{28}(w)
\end{aligned}
$$

For all v, $0 \leq v \leq a'$:

$$
\begin{aligned}
X_{10}(v) = &\partial_H \left(\int_{w=a-v}^{a'-v} s(x := 1)[w] \cdot A_2 \parallel B_2 \parallel V_2 \right) \\
= &\int_{w=a-v}^{\min(a'-v,d')} c(x := 1)[w] \cdot \partial_H \left(A_2 \parallel \int_{u=d-w}^{d'-w} i_2[u] \cdot B_3 \parallel V_1 \right) + \\
&\int_{u=d}^{\min(a'-v,d')} i_2[u] \cdot \partial_H \left(\int_{w=a-v-u}^{a'-v-u} s(x := 1)[w] \cdot A_2 \parallel B_3 \parallel V_2 \right) + \\
= &\int_{w=a-v}^{a'-v} c(x := 1)[w] \cdot \partial_H \left(A_2 \parallel \int_{u=d-w}^{d'-w} i_2[u] \cdot B_3 \parallel V_1 \right) + \\
&\int_{u=d}^{a'-v} i_2[u] \cdot \partial_H \left(\int_{w=a-v-u}^{a'-v-u} s(x := 1)[w] \cdot A_2 \parallel B_3 \parallel V_2 \right) \\
= &\int_{w=a-v}^{a'-v} c(x := 1)[w] \cdot X_{11}(w) + \int_{u=d}^{a'-v} i_2[u] \cdot \partial_H(\dots) \\
= &(\text{true}) :\rightarrow \int_{w=\max(0,a-v)}^{a'-v} c(x := 1)[w] \cdot X_{11}(w) + \\
&(v \leq a' - d) :\rightarrow \int_{u=d}^{a'-v} i_2[u] \cdot \partial_H(\dots) \\
= &(\text{true}) :\rightarrow \int_{w=\max(0,a-v)}^{a'-v} c(x := 1)[w] \cdot X_{11}(w) + (\text{false}) :\rightarrow \int_{u=d}^{a'-v} i_2[u] \cdot \partial_H(\dots) \\
= &(\text{true}) :\rightarrow \int_{w=\max(0,a-v)}^{a'-v} c(x := 1)[w] \cdot X_{11}(w)
\end{aligned}
$$

For all v, $0 \leq v \leq a'$:

$$
\begin{aligned}
X_{11}(v) = &\partial_H \left(A_2 \parallel \int_{w=d-v}^{d'-v} i_2[w] \cdot B_3 \parallel V_1 \right) \\
= &\int_{u=d}^{\min(d',d'-v)} i_1[u] \cdot \partial_H \left(A_3 \parallel \int_{w=d-v-u}^{d'-v-u} i_2[w] \cdot B_3 \parallel V_1 \right) +
\end{aligned}
$$

$$
\begin{aligned}
&\int_{w=d-v}^{\min(d',d'-v)} i_2[w] \cdot \partial_H \left(\int_{u=d-w}^{d'-w} i_1[u] \cdot A_3 \parallel B_3 \parallel V_1 \right) \\
= &\int_{u=d}^{d'-v} i_1[u] \cdot \partial_H \left(A_3 \parallel \int_{w=d-v-u}^{d'-v-u} i_2[w] \cdot B_3 \parallel V_1 \right) + \\
&\int_{w=d-v}^{d'-v} i_2[w] \cdot \partial_H \left(\int_{u=d-w}^{d'-w} i_1[u] \cdot A_3 \parallel B_3 \parallel V_1 \right) \\
= &\int_{u=d}^{d'-v} i_1[u] \cdot X_{12}(u+v) + \int_{w=d-v}^{d'-v} i_2[w] \cdot X_{13}(w) \\
= &(v \le d' - d) :\rightarrow \int_{u=d}^{d'-v} i_1[u] \cdot X_{12}(u+v) + (\text{true}) :\rightarrow \int_{w=d-v}^{d'-v} i_2[w] \cdot X_{13}(w)
\end{aligned}
$$

For all v, $d \le v \le d'$:

$$
\begin{aligned}
X_{12}(v) &= \partial_H \left(A_3 \parallel \int_{w=d-v}^{d'-v} i_2[w] \cdot B_3 \parallel V_1 \right) \\
&= c(x=1)[0] \cdot \partial_H \left(A_4 \parallel \int_{w=d-v}^{d'-v} i_2[w] \cdot B_3 \parallel V_1 \right) + \\
&\quad (v \ge d) :\rightarrow i_2[0] \cdot \partial_H(A_3 \parallel B_3 \parallel V_1) \\
&= c(x=1)[0] \cdot X_{23}(v) + (v \ge d) :\rightarrow i_2[0] \cdot X_{14} \\
&= c(x=1)[0] \cdot X_{23}(v) + (\text{true}) :\rightarrow i_2[0] \cdot X_{14} \\
&= c(x=1)[0] \cdot X_{23}(v) + i_2[0] \cdot X_{14}
\end{aligned}
$$

For all v, $d - a' \le v \le d'$:

$$
\begin{aligned}
X_{13}(v) &= \partial_H \left(\int_{w=d-v}^{d'-v} i_1[w] \cdot A_3 \parallel B_3 \parallel V_1 \right) \\
&= (v \ge d) :\rightarrow i_1[0] \cdot \partial_H(A_3 \parallel B_3 \parallel V_1) + c(x=1)[0] \cdot \partial_H \left(\int_{w=d-v}^{d'-v} i_1[w] \cdot A_3 \parallel B_0 \parallel V_1 \right) \\
&= (v \ge d) :\rightarrow i_1[0] \cdot X_{14} + c(x=1)[0] \cdot X_{15}(v)
\end{aligned}
$$

$$
\begin{aligned}
X_{14} &= \partial_H(A_3 \parallel B_3 \parallel V_1) \\
&= c(x=1)[0] \cdot \partial_H(A_4 \parallel B_3 \parallel V_1) + c(x=1)[0] \cdot \partial_H(A_3 \parallel B_0 \parallel V_1) \\
&= c(x=1)[0] \cdot X_{20} + c(x=1)[0] \cdot X_{16}
\end{aligned}
$$

For all v, $d - a' \le v \le d'$:

$$
\begin{aligned}
X_{15}(v) &= \partial_H \left(\int_{w=d-v}^{d'-v} i_1[w] \cdot A_3 \parallel B_0 \parallel V_1 \right) \\
&= \int_{w=d-v}^{d'-v} i_1[w] \cdot \partial_H(A_3 \parallel B_0 \parallel V_1) \\
&= \int_{w=d-v}^{d'-v} i_1[w] \cdot X_{16} \\
&= (\text{true}) :\rightarrow \int_{w=\max(0,d-v)}^{d'-v} i_1[w] \cdot X_{16}
\end{aligned}
$$

$$
X_{16} = \partial_H(A_3 \parallel B_0 \parallel V_1)
$$

$$
\begin{aligned}
&= c(x = 1)[0] \cdot \partial_H(A_4 \parallel B_0 \parallel V_1) \\
&= c(x = 1)[0] \cdot X_{17}
\end{aligned}
$$

$$
\begin{aligned}
X_{17} &= \partial_H(A_4 \parallel B_0 \parallel V_1) \\
&= \mathrm{EnterCS}_1[0] \cdot \partial_H(A_5 \parallel B_0 \parallel V_1) \\
&= \mathrm{EnterCS}_1[0] \cdot X_{18}
\end{aligned}
$$

$$
\begin{aligned}
X_{18} &= \partial_H(A_5 \parallel B_0 \parallel V_1) \\
&= \int_{v=0}^{\infty} \mathrm{LeaveCS}_1[v] \cdot \partial_H(A_6 \parallel B_0 \parallel V_1) \\
&= \int_{v=0}^{\infty} \mathrm{LeaveCS}_1[v] \cdot X_{19} \\
&= (\mathrm{true}) :\rightarrow \int_{v=0}^{\infty} \mathrm{LeaveCS}_1[v] \cdot X_{19}
\end{aligned}
$$

$$
\begin{aligned}
X_{19} &= \partial_H(A_6 \parallel B_0 \parallel V_1) \\
&= \int_{v=a}^{a'} c(x \coloneqq 0) \cdot \partial_H(A_0 \parallel B_0 \parallel V_0) \\
&= \int_{v=a}^{a'} c(x \coloneqq 0)[v] \cdot X_0 \\
&= (\mathrm{true}) :\rightarrow \int_{v=a}^{a'} c(x \coloneqq 0)[v] \cdot X_0
\end{aligned}
$$

$$
\begin{aligned}
X_{20} &= \partial_H(A_4 \parallel B_3 \parallel V_1) \\
&= \mathrm{EnterCS}_1[0] \cdot \partial_H(A_5 \parallel B_3 \parallel V_1) + c(x = 1)[0] \cdot \partial_H(A_4 \parallel B_0 \parallel V_1) \\
&= \mathrm{EnterCS}_1[0] \cdot X_{21} + c(x = 1)[0] \cdot X_{17}
\end{aligned}
$$

$$
\begin{aligned}
X_{21} &= \partial_H(A_5 \parallel B_3 \parallel V_1) \\
&= \mathrm{LeaveCS}_1[0] \cdot \partial_H(A_6 \parallel B_3 \parallel V_1) + c(x = 1)[0] \cdot \partial_H(A_5 \parallel B_0 \parallel V_1) \\
&= \mathrm{LeaveCS}_1[0] \cdot X_{22} + c(x = 1)[0] \cdot X_{18}
\end{aligned}
$$

$$
\begin{aligned}
X_{22} &= \partial_H(A_6 \parallel B_3 \parallel V_1) \\
&= c(x = 1)[0] \cdot \partial_H(A_6 \parallel B_0 \parallel V_1) \\
&= c(x = 1)[0] \cdot X_{19}
\end{aligned}
$$

For all v, $d \leq v \leq d'$:

$$
\begin{aligned}
X_{23}(v) &= \partial_H\left(A_4 \parallel \int_{w=d-v}^{d'-v} i_2[w] \cdot B_3 \parallel V_1\right) \\
&= \mathrm{EnterCS}_1[0] \cdot \partial_H\left(A_5 \parallel \int_{w=d-v}^{d'-v} i_2[w] \cdot B_3 \parallel V_1\right) + \\
&\quad (v \geq d) :\rightarrow i_2[0] \cdot \partial_H(A_4 \parallel B_3 \parallel V_1) \\
&= \mathrm{EnterCS}_1[0] \cdot X_{24}(v) + (v \geq d) :\rightarrow i_2[0] \cdot X_{20} \\
&= \mathrm{EnterCS}_1[0] \cdot X_{24}(v) + (\mathrm{true}) :\rightarrow i_2[0] \cdot X_{20} \\
&= \mathrm{EnterCS}_1[0] \cdot X_{24}(v) + i_2[0] \cdot X_{20}
\end{aligned}
$$

For all v, $d \leq v \leq d'$:

$$
\begin{aligned}
X_{24}(v) &= \partial_H\left(A_5 \parallel \int_{w=d-v}^{d'-v} i_2[w] \cdot B_3 \parallel V_1\right) \\
&= \int_{u=0}^{\min(\infty,d'-v)} \mathrm{LeaveCS}_1[u] \cdot \partial_H\left(A_6 \parallel \int_{w=d-v-u}^{d'-v-u} i_2[w] \cdot B_3 \parallel V_1\right) + \\
&\quad \int_{w=d-v}^{\min(\infty,d'-v)} i_2[w] \cdot \partial_H(A_5 \parallel B_3 \parallel V_1) \\
&= \int_{u=0}^{d'-v} \mathrm{LeaveCS}_1[u] \cdot \partial_H\left(A_6 \parallel \int_{w=d-v-u}^{d'-v-u} i_2[w] \cdot B_3 \parallel V_1\right) + \\
&\quad \int_{w=d-v}^{d'-v} i_2[w] \cdot \partial_H(A_5 \parallel B_3 \parallel V_1) \\
&= \int_{u=0}^{d'-v} \mathrm{LeaveCS}_1[u] \cdot X_{25}(u+v) + \int_{w=d-v}^{d'-v} i_2[w] \cdot X_{21} \\
&= (\mathrm{true}) :\rightarrow \int_{u=0}^{d'-v} \mathrm{LeaveCS}_1[u] \cdot X_{25}(u+v) + (\mathrm{true}) :\rightarrow \int_{w=0}^{d'-v} i_2[w] \cdot X_{21}
\end{aligned}
$$

For all v, $d \leq v \leq d'$:

$$
\begin{aligned}
X_{25}(v) &= \partial_H\left(A_6 \parallel \int_{w=d-v}^{d'-v} i_2[w] \cdot B_3 \parallel V_1\right) \\
&= \int_{u=a}^{\min(a',d'-v)} c(x \coloneqq 0)[u] \cdot \partial_H(\ldots) + \\
&\quad \int_{w=d-v}^{\min(a',d'-v)} i_2[w] \cdot \partial_H\left(\int_{u=a-w}^{a'-w} s(x \coloneqq 0)[u] \cdot A_0 \parallel B_3 \parallel V_1\right) \\
&= \int_{u=a}^{d'-v} c(x \coloneqq 0)[u] \cdot \partial_H(\ldots) + \\
&\quad \int_{w=d-v}^{d'-v} i_2[w] \cdot \partial_H\left(\int_{u=a-w}^{a'-w} s(x \coloneqq 0)[u] \cdot A_0 \parallel B_3 \parallel V_1\right) \\
&= \int_{u=a}^{d'-v} c(x \coloneqq 0)[u] \cdot \partial_H(\ldots) + \int_{w=d-v}^{d'-v} i_2[w] \cdot X_{26}(w) \\
&= (v \leq d'-a) :\rightarrow \int_{u=a}^{d'-v} c(x \coloneqq 0)[u] \cdot \partial_H(\ldots) + \\
&\quad (\mathrm{true}) :\rightarrow \int_{w=0}^{d'-v} i_2[w] \cdot X_{26}(w) \\
&= (\mathrm{false}) :\rightarrow \int_{u=a}^{d'-v} c(x \coloneqq 0)[u] \cdot \partial_H(\ldots) + (\mathrm{true}) :\rightarrow \int_{w=0}^{d'-v} i_2[w] \cdot X_{26}(w) \\
&= (\mathrm{true}) :\rightarrow \int_{w=0}^{d'-v} i_2[w] \cdot X_{26}(w)
\end{aligned}
$$

For all v, $0 \leq v \leq d'-d$:

$$
\begin{aligned}
X_{26}(v) &= \partial_H\left(\int_{w=a-v}^{a'-v} s(x \coloneqq 0)[w] \cdot A_0 \parallel B_3 \parallel V_1\right) \\
&= (v \geq a) :\rightarrow c(x \coloneqq 0)[0] \cdot \partial_H(A_0 \parallel B_3 \parallel V_1) +
\end{aligned}
$$

$$
\begin{aligned}
& c(x=1)[0] \cdot \partial_H \left(\int_{w=a-v}^{a'-v} s(x \coloneqq 0) \cdot A_0 \parallel B_0 \parallel V_1 \right) \\
&= (v \geq a) :\rightarrow c(x \coloneqq 0)[0] \cdot \partial_H(\dots) + c(x=1)[0] \cdot X_{27}(v) \\
&= (\text{false}) :\rightarrow c(x \coloneqq 0)[0] \cdot \partial_H(\dots) + c(x=1)[0] \cdot X_{27}(v) \\
&= c(x=1)[0] \cdot X_{27}(v)
\end{aligned}
$$

For all v, $0 \leq v \leq d' - d$:

$$
\begin{aligned}
X_{27}(v) &= \partial_H \left(\int_{w=a-v}^{a'-v} s(x \coloneqq 0) \cdot A_0 \parallel B_0 \parallel V_1 \right) \\
&= \int_{w=a-v}^{a'-v} c(x \coloneqq 0)[w] \cdot \partial_H(A_0 \parallel B_0 \parallel V_0) \\
&= \int_{w=a-v}^{a'-v} c(x \coloneqq 0)[w] \cdot X_0 \\
&= (\text{true}) :\rightarrow \int_{w=a-v}^{a'-v} c(x \coloneqq 0)[w] \cdot X_0
\end{aligned}
$$

For all v, $0 \leq v \leq a' - a$:

$$
\begin{aligned}
X_{28}(v) &= \partial_H \left(\int_{w=d-v}^{d'-v} i_1[w] \cdot A_3 \parallel B_2 \parallel V_2 \right) \\
&= \int_{w=d-v}^{\min(d'-v,'d')} i_1[w] \cdot \partial_H \left(A_3 \parallel \int_{u=d-w}^{d'-w} i_2[u] \cdot B_3 \parallel V_2 \right) + \\
&\quad \int_{u=d}^{\min(d'-v,d')} i_2[u] \cdot \partial_H \left(\int_{w=d-v-u}^{d'-v-u} i_1[w] \cdot A_3 \parallel B_3 \parallel V_2 \right) \\
&= \int_{w=d-v}^{d'-v} i_1[w] \cdot \partial_H \left(A_3 \parallel \int_{u=d-w}^{d'-w} i_2[u] \cdot B_3 \parallel V_2 \right) + \\
&\quad \int_{u=d}^{d'-v} i_2[u] \cdot \partial_H \left(\int_{w=d-v-u}^{d'-v-u} i_1[w] \cdot A_3 \parallel B_3 \parallel V_2 \right) \\
&= \int_{w=d-v}^{d'-v} i_1[w] \cdot X_{29}(w) + \int_{u=d}^{d'-v} i_2[u] \cdot X_{30}(w) \\
&= (\text{true}) :\rightarrow \int_{w=d-v}^{d'-v} i_1[w] \cdot X_{29}(w) + \\
&\quad (v \leq d' - d) :\rightarrow \int_{u=d}^{d'-v} i_2[u] \cdot X_{30}(u+v)
\end{aligned}
$$

For all v, $d - a' + a \leq v \leq d'$:

$$
\begin{aligned}
X_{29}(v) &= \partial_H \left(A_3 \parallel \int_{w=d-v}^{d'-v} i_2[w] \cdot B_3 \parallel V_2 \right) \\
&= c(x=1)[0] \cdot \partial_H \left(A_0 \parallel \int_{w=d-v}^{d'-v} i_2[w] \cdot B_3 \parallel V_2 \right) + \\
&\quad (v \geq d) :\rightarrow i_2[0] \cdot \partial_H(A_3 \parallel B_3 \parallel V_2) \\
&= c(x=1)[0] \cdot X_{40}(v) + (v \geq d) :\rightarrow i_2[0] \cdot X_{31}
\end{aligned}
$$

For all v, $d \leq v \leq d'$:

$$\begin{aligned}
X_{30}(v) &= \partial_H\left(\int_{w=d-v}^{d'-v} i_1[w] \cdot A_3 \parallel B_3 \parallel V_2\right) \\
&= (v \geq d) :\rightarrow i_1[0] \cdot \partial_H(A_3 \parallel B_3 \parallel V_2) + \\
&\quad c(x=2)[0] \cdot \partial_H\left(\int_{w=d-v}^{d'-v} i_1[w] \cdot A_3 \parallel B_4 \parallel V_2\right) \\
&= i_1[0] \cdot \partial_H(A_3 \parallel B_3 \parallel V_2) + c(x=2)[0] \cdot \partial_H\left(\int_{w=d-v}^{d'-v} i_1[w] \cdot A_3 \parallel B_4 \parallel V_2\right) \\
&= i_1[0] \cdot X_{31} + c(x=2)[0] \cdot X_{32}(v)
\end{aligned}$$

$$\begin{aligned}
X_{31} &= \partial_H(A_3 \parallel B_3 \parallel V_2) \\
&= c(x=2)[0] \cdot \partial_H(A_0 \parallel B_3 \parallel V_2) + c(x=2)[0] \cdot \partial_H(A_3 \parallel B_4 \parallel V_2) \\
&= c(x=2)[0] \cdot X_5 + c(x=2)[0] \cdot X_{33}
\end{aligned}$$

For all v, $d \leq v \leq d'$:

$$\begin{aligned}
X_{32}(v) &= \partial_H\left(\int_{w=d-v}^{d'-v} i_1[w] \cdot A_3 \parallel B_4 \parallel V_2\right) \\
&= (v \geq d) :\rightarrow i_1[0] \cdot \partial_H(A_3 \parallel B_4 \parallel V_2) + \\
&\quad \mathrm{EnterCS}_2[0] \cdot \partial_H\left(\int_{w=d-v}^{d'-v} i_1[w] \cdot A_3 \parallel B_5 \parallel V_2\right) \\
&= (v \geq d) :\rightarrow i_1[0] \cdot X_{33} + \mathrm{EnterCS}_2[0] \cdot X_{34}(v) \\
&= (\mathrm{true}) :\rightarrow i_1[0] \cdot X_{33} + \mathrm{EnterCS}_2[0] \cdot X_{34}(v) \\
&= i_1[0] \cdot X_{33} + \mathrm{EnterCS}_2[0] \cdot X_{34}(v)
\end{aligned}$$

$$\begin{aligned}
X_{33} &= \partial_H(A_3 \parallel B_4 \parallel V_2) \\
&= c(x=2)[0] \cdot \partial_H(A_0 \parallel B_4 \parallel V_2) + \mathrm{EnterCS}_2[0] \cdot \partial_H(A_3 \parallel B_5 \parallel V_2) \\
&= c(x=2)[0] \cdot X_6 + \mathrm{EnterCS}_2[0] \cdot X_{35}
\end{aligned}$$

For all v, $d \leq v \leq d'$:

$$\begin{aligned}
X_{34}(v) &= \partial_H\left(\int_{w=d-v}^{d'-v} i_1[w] \cdot A_3 \parallel B_5 \parallel V_2\right) \\
&= \int_{w=d-v}^{\min(d'-v,\infty)} i_1[w] \cdot \partial_H(A_3 \parallel B_5 \parallel V_2) + \\
&\quad \int_{u=0}^{\min(d'-v,\infty)} \mathrm{LeaveCS}_2[u] \cdot \partial_H\left(\int_{w=d-v-u}^{d'-v-u} i_1[w] \cdot A_3 \parallel B_6 \parallel V_2\right) \\
&= \int_{w=d-v}^{d'-v} i_1[w] \cdot \partial_H(A_3 \parallel B_5 \parallel V_2) + \\
&\quad \int_{u=0}^{d'-v} \mathrm{LeaveCS}_2[u] \cdot \partial_H\left(\int_{w=d-v-u}^{d'-v-u} i_1[w] \cdot A_3 \parallel B_6 \parallel V_2\right) \\
&= \int_{w=d-v}^{d'-v} i_1[w] \cdot X_{35} + \int_{u=0}^{d'-v} \mathrm{LeaveCS}_2[u] \cdot X_{36}(u+v) \\
&= (\mathrm{true}) :\rightarrow \int_{w=0}^{d'-v} i_1[w] \cdot X_{35} + (\mathrm{true}) :\rightarrow \int_{u=0}^{d'-v} \mathrm{LeaveCS}_2[u] \cdot X_{36}(u+v)
\end{aligned}$$

$$
\begin{aligned}
X_{35} &= \partial_H(A_3 \parallel B_5 \parallel V_2) \\
&= c(x=2)[0] \cdot \partial_H(A_0 \parallel B_5 \parallel V_2) + \mathrm{LeaveCS}_2[0] \cdot \partial_H(A_3 \parallel B_6 \parallel V_2) \\
&= c(x=2)[0] \cdot X_7 + \mathrm{LeaveCS}_2[0] \cdot X_{39}
\end{aligned}
$$

For all v, $d \leq v \leq d'$:

$$
\begin{aligned}
X_{36}(v) &= \partial_H\left(\int_{w=d-v}^{d'-v} i_1[w] \cdot A_3 \parallel B_6 \parallel V_2\right) \\
&= \int_{w=d-v}^{\min(d'-v,a')} i_1[w] \cdot \partial_H\left(A_3 \parallel \int_{u=a-w}^{a'-w} s(x \coloneqq 0)[u] \cdot B_0 \parallel V_2\right) + \\
&\quad \int_{u=a}^{\min(d'-v,a')} c(x \coloneqq 0)[u] \cdot \partial_H(\ldots) \\
&= \int_{w=d-v}^{d'-v} i_1[w] \cdot \partial_H\left(A_3 \parallel \int_{u=a-w}^{a'-w} s(x \coloneqq 0)[u] \cdot B_0 \parallel V_2\right) + \\
&\quad \int_{u=a}^{d'-v} c(x \coloneqq 0)[u] \cdot \partial_H(\ldots) \\
&= \int_{w=d-v}^{d'-v} i_1[w] \cdot X_{37}(w) + \int_{u=a}^{d'-v} c(x \coloneqq 0)[u] \cdot \partial_H(\ldots) \\
&= (\mathrm{true}) :\rightarrow \int_{w=0}^{d'-v} i_1[w] \cdot X_{37}(w) + \\
&\quad (v \leq d'-a) :\rightarrow \int_{u=a}^{d'-v} c(x \coloneqq 0)[u] \cdot \partial_H(\ldots) \\
&= (\mathrm{true}) :\rightarrow \int_{w=0}^{d'-v} i_1[w] \cdot X_{37}(w) + (\mathrm{false}) :\rightarrow \int_{u=a}^{d'-v} c(x \coloneqq 0)[u] \cdot \partial_H(\ldots) \\
&= (\mathrm{true}) :\rightarrow \int_{w=0}^{d'-v} i_1[w] \cdot X_{37}(w)
\end{aligned}
$$

For all v, $0 \leq v \leq d'-d$:

$$
\begin{aligned}
X_{37}(v) &= \partial_H\left(A_3 \parallel \int_{w=a-v}^{a'-v} s(x \coloneqq 0)[w] \cdot B_0 \parallel V_2\right) \\
&= c(x=2)[0] \cdot \partial_H\left(A_0 \parallel \int_{w=a-v}^{a'-v} s(x \coloneqq 0)[w] \cdot B_0 \parallel V_2\right) + \\
&\quad (v \geq a) :\rightarrow c(x \coloneqq 0)[0] \cdot \partial_H(A_3 \parallel B_0 \parallel V_0) \\
&= c(x=2)[0] \cdot X_{38}(v) + (v \geq a) :\rightarrow c(x \coloneqq 0)[0] \cdot \partial_H(\ldots) \\
&= c(x=2)[0] \cdot X_{38}(v) + (\mathrm{false}) :\rightarrow c(x \coloneqq 0)[0] \cdot \partial_H(\ldots) \\
&= c(x=2)[0] \cdot X_{38}(v)
\end{aligned}
$$

For all v, $0 \leq v \leq d'-d$:

$$
\begin{aligned}
X_{38}(v) &= \partial_H\left(A_0 \parallel \int_{w=a-v}^{a'-v} s(x \coloneqq 0)[w] \cdot B_0 \parallel V_2\right) \\
&= \int_{w=a-v}^{a'-v} c(x \coloneqq 0)[w] \cdot \partial_H(A_0 \parallel B_0 \parallel V_0) \\
&= \int_{w=a-v}^{a'-v} c(x \coloneqq 0)[w] \cdot X_0
\end{aligned}
$$

$$= (\text{true}) :\rightarrow \int_{w=a-v}^{a'-v} c(x := 0)[w] \cdot X_0$$

$$\begin{aligned} X_{39} &= \partial_H(A_3 \parallel B_6 \parallel V_2) \\ &= c(x = 2)[0] \cdot \partial_H(A_0 \parallel B_6 \parallel V_2) \\ &= c(x = 2)[0] \cdot X_8 \end{aligned}$$

For all v, $d - a' + a \leq v \leq d'$:

$$\begin{aligned} X_{40}(v) &= \partial_H\left(A_0 \parallel \int_{w=d-v}^{d'-v} i_2[w] \cdot B_3 \parallel V_2\right) \\ &= \int_{w=d-v}^{d'-v} i_2[w] \cdot \partial_H(A_0 \parallel B_3 \parallel V_2) \\ &= \int_{w=d-v}^{d'-v} i_2[w] \cdot X_5 \\ &= (\text{true}) :\rightarrow \int_{w=d-v}^{d'-v} i_2[w] \cdot X_5 \end{aligned}$$

$$\begin{aligned} X_{41} &= \partial_H(A_2 \parallel B_0 \parallel V_1) \\ &= \int_{v=d}^{d'} i_1[v] \cdot \partial_H(A_3 \parallel B_0 \parallel V_1) \\ &= \int_{v=d}^{d'} i_1[v] \cdot X_{16} \\ &= (\text{true}) :\rightarrow \int_{v=d}^{d'} i_1[v] \cdot X_{16} \end{aligned}$$

For all v, $0 \leq v \leq a'$:

$$\begin{aligned} X_{42}(v) &= \partial_H\left(\int_{w=a-v}^{a'-v} s(x := 1)[w] \cdot A_2 \parallel B_1 \parallel V_0\right) \\ &= \int_{w=a-v}^{\min(a'-v,a')} c(x := 1)[w] \cdot \partial_H\left(A_2 \parallel \int_{u=a-w}^{a'-w} s(x := 2)[u] \cdot B_2 \parallel V_1\right) + \\ &\quad \int_{u=a}^{\min(a'-v,a')} c(x := 2)[u] \cdot \partial_H\left(\int_{w=a-v-u}^{a'-v-u} s(x := 1)[w] \cdot A_2 \parallel B_2 \parallel V_2\right) \\ &= \int_{w=a-v}^{a'-v} c(x := 1)[w] \cdot \partial_H\left(A_2 \parallel \int_{u=a-w}^{a'-w} s(x := 2)[u] \cdot B_2 \parallel V_1\right) + \\ &\quad \int_{u=a}^{a'-v} c(x := 2)[u] \cdot \partial_H\left(\int_{w=a-v-u}^{a'-v-u} s(x := 1)[w] \cdot A_2 \parallel B_2 \parallel V_2\right) \\ &= \int_{w=a-v}^{a'-v} c(x := 1)[w] \cdot X_{43}(w) + \int_{u=a}^{a'-v} c(x := 2)[u] \cdot X_{44}(u + v) \\ &= (\text{true}) :\rightarrow \int_{w=\max(0,a-v)}^{a'-v} c(x := 1)[w] \cdot X_{43}(w) + \\ &\quad (v \leq a' - a) :\rightarrow \int_{u=a}^{a'-v} c(x := 2)[u] \cdot X_{44}(u + v) \end{aligned}$$

For all v, $0 \le v \le a'$:

$$
\begin{aligned}
X_{43}(v) &= \partial_H\left(A_2 \parallel \int_{w=a-v}^{a'-v} s(x \coloneqq 2)[w] \cdot B_2 \parallel V_1\right) \\
&= \int_{u=d}^{\min(d',a'-v)} i_1[u] \cdot \partial_H\left(A_3 \parallel \int_{w=a-v-u}^{a'-v-u} s(x \coloneqq 2)[w] \cdot B_2 \parallel V_1\right) + \\
&\quad \int_{w=a-v}^{\min(d',a'-v)} c(x \coloneqq 2)[w] \cdot \partial_H\left(\int_{u=d-w}^{d'-w} i_1[u] \cdot A_3 \parallel B_2 \parallel V_1\right) \\
&= \int_{u=d}^{a'-v} i_1[u] \cdot \partial_H\left(A_3 \parallel \int_{w=a-v-u}^{a'-v-u} s(x \coloneqq 2)[w] \cdot B_2 \parallel V_1\right) + \\
&\quad \int_{w=a-v}^{a'-v} c(x \coloneqq 2)[w] \cdot \partial_H\left(\int_{u=d-w}^{d'-w} i_1[u] \cdot A_3 \parallel B_2 \parallel V_2\right) \\
&= \int_{u=d}^{a'-v} i_1[u] \cdot \partial_H(\dots) + \int_{w=a-v}^{a'-v} c(x \coloneqq 2)[w] \cdot X_{48}(w) \\
&= (v \le a' - d) :\rightarrow \int_{u=d}^{a'-v} i_1[u] \cdot \partial_H(\dots) + \\
&\quad (\text{true}) :\rightarrow \int_{w=a-v}^{a'-v} c(x \coloneqq 2)[w] \cdot X_{48}(w) \\
&= (\text{false}) :\rightarrow \int_{u=d}^{a'-v} i_1[u] \cdot \partial_H(\dots) + (\text{true}) :\rightarrow \int_{w=a-v}^{a'-v} c(x \coloneqq 2)[w] \cdot X_{48}(w) \\
&= (\text{true}) :\rightarrow \int_{w=a-v}^{a'-v} c(x \coloneqq 2)[w] \cdot X_{48}(w)
\end{aligned}
$$

For all v, $a \le v \le a'$:

$$
\begin{aligned}
X_{44}(v) &= \partial_H\left(\int_{w=a-v}^{a'-v} s(x \coloneqq 1)[w] \cdot A_2 \parallel B_2 \parallel V_2\right) \\
&= \int_{w=a-v}^{\min(a'-v,d')} c(x \coloneqq 1)[w] \cdot \partial_H\left(A_2 \parallel \int_{u=d-w}^{d'-w} i_2[u] \cdot B_3 \parallel V_1\right) + \\
&\quad \int_{u=d}^{\min(a'-v,d')} i_2[u] \cdot \partial_H\left(\int_{w=a-v-u}^{a'-v-u} s(x \coloneqq 1)[u] \cdot A_2 \parallel B_3 \parallel V_2\right) \\
&= \int_{w=a-v}^{a'-v} c(x \coloneqq 1)[w] \cdot \partial_H\left(A_2 \parallel \int_{u=d-w}^{d'-w} i_2[u] \cdot B_3 \parallel V_1\right) + \\
&\quad \int_{u=d}^{a'-v} i_2[u] \cdot \partial_H\left(\int_{w=a-v-u}^{a'-v-u} s(x \coloneqq 1)[u] \cdot A_2 \parallel B_3 \parallel V_2\right) \\
&= \int_{w=a-v}^{a'-v} c(x \coloneqq 1)[w] \cdot X_{45}(w) + \int_{u=d}^{a'-v} i_2[u] \cdot \partial_H(\dots) \\
&= (\text{true}) :\rightarrow \int_{w=0}^{a'-v} c(x \coloneqq 1)[w] \cdot X_{45}(w) + \\
&\quad (v \le a' - d) :\rightarrow \int_{u=d}^{a'-v} i_2[u] \cdot \partial_H(\dots) \\
&= (\text{true}) :\rightarrow \int_{w=0}^{a'-v} c(x \coloneqq 1)[w] \cdot X_{45}(w) + (\text{false}) :\rightarrow \int_{u=d}^{a'-v} i_2[u] \cdot \partial_H(\dots) \\
&= (\text{true}) :\rightarrow \int_{w=0}^{a'-v} c(x \coloneqq 1)[w] \cdot X_{45}(w)
\end{aligned}
$$

For all v, $0 \leq v \leq a' - a$:

$$\begin{aligned}
X_{45}(v) &= \partial_H\left(A_2 \parallel \int_{w=d-v}^{d'-v} i_2[w] \cdot B_3 \parallel V_1\right) \\
&= \int_{u=d}^{\min(d',d'-v)} i_1[u] \cdot \partial_H\left(A_3 \parallel \int_{w=d-v-u}^{d'-v-u} i_2[w] \cdot B_3 \parallel V_1\right) + \\
&\quad \int_{w=d-v}^{\min(d',d'-v)} i_2[w] \cdot \partial_H\left(\int_{u=d-w}^{d'-w} i_1[u] \cdot A_3 \parallel B_3 \parallel V_1\right) \\
&= \int_{u=d}^{d'-v} i_1[u] \cdot \partial_H\left(A_3 \parallel \int_{w=d-v-u}^{d'-v-u} i_2[w] \cdot B_3 \parallel V_1\right) + \\
&\quad \int_{w=d-v}^{d'-v} i_2[w] \cdot \partial_H\left(\int_{u=d-w}^{d'-w} i_1[u] \cdot A_3 \parallel B_3 \parallel V_1\right) \\
&= \int_{u=d}^{d'-v} i_1[u] \cdot X_{12}(u) + \int_{w=d-v}^{d'-v} i_2[w] \cdot X_{46}(w) \\
&= (v \leq d' - d) :\rightarrow \int_{u=d}^{d'-v} i_1[u] \cdot X_{12}(v+u) + (\text{true}) :\rightarrow \int_{w=d-v}^{d'-v} i_2[w] \cdot X_{46}(w)
\end{aligned}$$

For all v, $d - a' + a \leq v \leq d'$:

$$\begin{aligned}
X_{46}(v) &= \partial_H\left(\int_{w=d-v}^{d'-v} i_1[w] \cdot A_3 \parallel B_3 \parallel V_1\right) \\
&= c(x=1)[0] \cdot \partial_H\left(\int_{w=d-v}^{d'-v} i_1[w] \cdot A_3 \parallel B_0 \parallel V_1\right) + \\
&\quad (v \geq d) :\rightarrow i_1[0] \cdot \partial_H(A_3 \parallel B_3 \parallel V_1) \\
&= c(x=1)[0] \cdot X_{47}(v) + (v \geq d) :\rightarrow i_1[0] \cdot X_{14}
\end{aligned}$$

For all v, $d - a' + a \leq v \leq d'$:

$$\begin{aligned}
X_{47}(v) &= \partial_H\left(\int_{w=d-v}^{d'-v} i_1[w] \cdot A_3 \parallel B_0 \parallel V_1\right) \\
&= \int_{w=d-v}^{d'-v} i_1[w] \cdot \partial_H(A_3 \parallel B_0 \parallel V_1) \\
&= (\text{true}) :\rightarrow \int_{w=\max(0,d-v)}^{d'-v} i_1[w] \cdot X_{16}
\end{aligned}$$

For all v, $0 \leq v \leq a'$:

$$\begin{aligned}
X_{48}(v) &= \partial_H\left(\int_{w=d-v}^{d'-v} i_1[w] \cdot A_3 \parallel B_2 \parallel V_2\right) \\
&= \int_{w=d-v}^{\min(d'-v,d')} i_1[w] \cdot \partial_H\left(A_3 \parallel \int_{u=d-w}^{d'-w} i_2[u] \cdot B_3 \parallel V_2\right) + \\
&\quad \int_{u=d}^{\min(d'-v,d')} i_2[u] \cdot \partial_H\left(\int_{w=d-v-u}^{d'-v-u} i_1[w] A_3 \parallel B_3 \parallel V_2\right) \\
&= \int_{w=d-v}^{d'-v} i_1[w] \cdot \partial_H\left(A_3 \parallel \int_{u=d-w}^{d'-w} i_2[u] \cdot B_3 \parallel V_2\right) +
\end{aligned}$$

$$
\begin{aligned}
& \int_{u=d}^{d'-v} i_2[u] \cdot \partial_H \left(\int_{w=d-v-u}^{d'-v-u} i_1[w] A_3 \parallel B_3 \parallel V_2 \right) \\
= & \int_{w=d-v}^{d'-v} i_1[w] \cdot X_{49}(w) + \int_{u=d}^{d'-v} i_2[u] \cdot X_{30}(u+v) \\
= & \ (\text{true}) :\rightarrow \int_{w=d-v}^{d'-v} i_1[w] \cdot X_{49}(w) + (v \leq d' - d) :\rightarrow \int_{u=d}^{d'-v} i_2[u] \cdot X_{30}(u+v)
\end{aligned}
$$

For all v, $d - a' \leq v \leq d'$:

$$
\begin{aligned}
X_{49}(v) = & \ \partial_H \left(A_3 \parallel \int_{w=d-v}^{d'-v} i_2[w] \cdot B_3 \parallel V_2 \right) \\
= & \ c(x=2)[0] \cdot \partial_H \left(A_0 \parallel \int_{w=d-v}^{d'-v} i_2[w] \cdot B_3 \parallel V_2 \right) + \\
& (v \geq d) :\rightarrow i_2[0] \cdot \partial_H (A_3 \parallel B_3 \parallel V_2) \\
= & \ c(x=2)[0] \cdot X_{50}(v) + (v \geq d) :\rightarrow i_2[0] \cdot X_{31}
\end{aligned}
$$

For all v, $d' - a \leq v \leq d'$:

$$
\begin{aligned}
X_{50}(v) = & \ \partial_H \left(A_0 \parallel \int_{w=d-v}^{d'-v} i_2[w] \cdot B_3 \parallel V_2 \right) \\
= & \int_{w=d-v}^{d'-v} i_2[w] \cdot \partial_H (A_0 \parallel B_3 \parallel V_2) \\
= & \int_{w=d-v}^{d'-v} i_2[w] \cdot X_5 \\
= & \ (\text{true}) :\rightarrow \int_{w=\max(0,d-v)}^{d'-v} i_2[w] \cdot X_5
\end{aligned}
$$

Acknowledgments

We would like to thank Martín Abadi, Jos Baeten, Roland Bol, Jozef Hooman, Hans Mulder, Michel Reniers, and Fred Schneider for their various comments and fruitful discussions, and Jan Friso Groote for kindly providing the PASCAL source of his bisimulation-checking tool (which is described in [Gro91, §3]).

Bibliography

[Aba94] M. Abadi, 1994. Personal communication.

[AL92a] M. Abadi and L. Lamport. An old-fashioned recipe for real time. Technical Report Systems Research Center 91, Digital Equipment Corporation, October 1992. To appear in *ACM Transactions on Programming Languages and Systems*. An earlier version appeared as [AL92b].

[AL92b] M. Abadi and L. Lamport. An old-fashioned recipe for real time, 1992. In [dBHdRR92], pages 1–27.

[Bae90] J.C.M. Baeten, editor. *Applications of Process Algebra*. Number 18 in Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 1990.

[BB91] J.C.M. Baeten and J.A. Bergstra. Real time process algebra. *Journal of Formal Aspects of Computing*, 3(2):142–188, 1991.

[BB92a] J.C.M. Baeten and J.A. Bergstra. Discrete time process algebra. Technical Report CSN 92/06, Eindhoven University of Technology, Computing Science Department, 1992. An extended abstract appeared as [BB92b].

[BB92b] J.C.M. Baeten and J.A. Bergstra. Discrete time process algebra. In [Cle92], pages 401–420, 1992. Extended abstract.

[BBB93] J.C.M. Baeten, J.A. Bergstra, and R.N. Bol. A real-time process logic. Technical Report CSN 93/15, Eindhoven University of Technology, Computing Science Department, 1993. An extended abstract appeared as [BBB94].

[BBB94] J.C.M. Baeten, J.A. Bergstra, and R.N. Bol. A real-time process logic. In [GO94], pages 30–47, 1994.

[BM94] J.C.M. Baeten and S. Mauw. Delayed choice: an operator for joining message sequence charts, 1994. To appear in the proceedings of FORTE '94, Berne, Switzerland, October 1994.

[BW90] J.C.M. Baeten and W.P. Weijland. *Process Algebra*. Number 18 in Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 1990.

[Cle92] W.R. Cleaveland, editor. *CONCUR '92, Third International Conference on Concurrency Theory*. Number 630 in Lecture Notes in Computer Science. Springer-Verlag, 1992. Proceedings of CONCUR '92, Stony Brook, NY, USA, August 1992.

[dB67] N.G. de Bruijn. Additional comments on a problem in concurrent programming control. *Communications of the ACM*, 10(3):137–138, 1967.

[dBHdRR92] J.W. de Bakker, C. Huizing, W.P. de Roever, and G. Rozenberg, editors. *Real-Time: Theory in Practice*. Number 600 in Lecture Notes in Computer Science. Springer-Verlag, 1992. Proceedings of the REX Workshop, Mook, The Netherlands, June 1991.

[Dij65] E.W. Dijkstra. Solution of a problem in concurrent programming control. *Communications of the ACM*, 8(9):569, 1965.

[Fis85] M. Fischer. Re: where are you? Electronic mail message from Michael Fischer to Leslie Lamport. Arpanet message sent on June 25, 1985 18:56:29 EDT, number `8506252257.AA07636@yale-bulldog.yale.arpa` (47 lines), 1985.

[GO94] D.M. Gabbay and H.J. Ohlbach, editors. *Temporal Logic, First International Conference*. Number 827 in Lecture Notes in Artificial Intelligence (Subseries of Lecture Notes in Computer Science). Springer-Verlag, 1994. Proceedings of ICTL' 94, Bonn, Germany, July 1994.

[Gro91] J.F. Groote. *Process Algebra and Structured Operational Semantics*. PhD thesis, University of Amsterdam, 1991.

[Hil94] J. Hillebrand. The ABP and the CABP—a comparison of performances in real time process algebra. In [PVvV94], pages 123–157, 1994.

[Hoa85] C.A.R. Hoare. *Communicating Sequential Processes*. International Series in Computer Science. Prentice Hall, 1985.

[JPXZ94] W. Janssen, M. Poel, Q. Xu, and J. Zwiers. Layering of real-time distributed processes, 1994. To appear in the proceedings of the *Third International School and Symposium on Formal Techniques in Real Time and Fault Tolerant Systems*, Lübeck, Germany, September 1994.

[Klu93] A.S. Klusener. *Models and Axioms for a Fragment of Real Time Process Algebra*. PhD thesis, Eindhoven University of Technology, Computing Science Department, 1993.

[Knu66] D.E. Knuth. Additional comments on a problem in concurrent programming control. *Communications of the ACM*, 9(5):321–322, 1966.

[Lam74] L. Lamport. A new solution of Dijkstra's concurrent programming problem. *Communications of the ACM*, 17(8):453–455, 1974.

[Lam87] L. Lamport. A fast mutual exclusion algorithm. *ACM Transactions on Computer Systems*, 5(1):1–11, 1987.

[Mil89] R. Milner. *Communication and Concurrency*. International Series in Computer Science. Prentice Hall, 1989.

[Nie90] E.R. Nieuwland. Proving mutual exclusion with process algebra, 1990. In [Bae90], pages 45–51.

[PVvV94] A. Ponse, C. Verhoef, and S.F.M. van Vlijmen, editors. ACP94, workshop on algebra of communicating processes. Technical Report P9413, University of Amsterdam, Programming Research Group, May 1994.

[SBM92] F. Schneider, B. Bloom, and K. Marzullo. Putting time into proof outlines. In [dBHdRR92], pages 618–639, 1992.

[Sch94] F. Schneider, 1994. Personal communication.

Typeset by LaTeX 2_ε in Lucida® Bright.