COMPOSITIONAL VERIFICATION
AND SPECIFICATION OF
REFINEMENT FOR REACTIVE
SYSTEMS IN A DENSE TIME
TEMPORAL LOGIC

Dissertation
zur Erlangung des Doktorgrades
der Technischen Fakultät
der Christian-Albrechts-Universität
zu Kiel

vorgelegt von
Antonio Cau

Kiel,
im August 1995
Berichterstatter: Prof. Dr. Willem-Paul de Roever

---------------------------------------------

Tag der mündlichen Prüfung: 14 Dezember 1995  

Zum Druck genehmigt: Kiel, den 14 Dezember 1995  

---------------------------------------------

(Dekan)
Abstract

This thesis introduces a compositional dense time temporal logic for the composition and refinement of reactive systems. A reactive system is specified by a pair consisting of a machine and a condition on the computations of this machine. In order to compose reactive systems, each step in a computation has additionally composition information such as “this is a system step”, or “this is an environment step” or “this is a communication step”. By defining a merge operator that merges two steps into one step compositionality is achieved. Because a dense time temporal logic is used refinement can be expressed easily in this logic. Existing proof rules for refinement are reformulated in our formalism. The notion of relative refinement is introduced to handle refinement of systems that only under certain conditions are considered to be correct refinements. The proof rules for “normal” refinement are extended to handle relative refinement of systems. Relative refinement is used to formalize Dijkstra’s development strategy for the solution of the readers/writers problem and to formalize a development strategy for certain fault tolerant systems. This development strategy is applied to the development of a fault tolerant storage system.
Acknowledgements

I would like to thank my advisor Willem-Paul de Roever for his guidance and his scientific support especially during the first two years of writing this thesis. Without his support this thesis would be just “spaghetti code”. Special thanks go to Pierre Collette, the collaboration with whom was very agreeable and fruitful and led to many insights. I would like to thank all my colleagues of the group “Software Technology” at the Christian-Albrecht-University for their support, especially Yassine Lakhneche, who made many invaluable comments during the last stage of the thesis, Kai Engelhardt, who acted as a “finite variability condition” with respect to the changes I wanted to make in my thesis, and Qiwen Xu with whom I started to collaborate after he left the group. Also members of the other groups at the Christian Albrechts University are thanked for their comments, especially Thomas Wilke. I also want to thank Ruurd Kuiper for his help and comments during the first and last stage of the thesis. Finally I would like to thank the Personalrat and the Technical Faculty of the Christian Albrechts University in the persons of Reimer Hansen and Frank Paul for their efforts in extending my contract for 4 months, thus enabling me to finish this thesis. Last but not least I want to thank Anne Straßer for her (moral) support during the last stage of the thesis.
Contents

Abstract iii
Acknowledgements v
1 Introduction 1
2 A Dense Model Formalism 3
  2.1 Introduction .................................................................. 3
  2.2 Specification of Reactive Systems ...................................... 6
    2.2.1 Semantic Specification of Reactive Systems .................. 6
    2.2.2 DTL Specification of Reactive Systems ....................... 16
  2.3 Refinement and Composition of Reactive System Specifications .......................................................... 29
    2.3.1 Semantic Refinement and Composition of Specifications .......... 29
    2.3.2 Refinement and Composition of DTL Specifications ............ 31
  2.4 Proving Refinement of Reactive System Specifications ............... 37
    2.4.1 Proving Semantic Refinement of Specifications ................. 37
    2.4.2 Proving Refinement of DTL Specifications ..................... 38
  2.5 Relative Refinement and Composition of Reactive System Specifications .......................................................... 43
    2.5.1 Semantic Relative Refinement and Composition of Specifications .......... 43
    2.5.2 Relative Refinement and Composition of DTL Specifications .... 44
    2.5.3 Proving Semantic Relative Refinement of Specifications ........ 45
    2.5.4 Proving Relative Refinement of DTL Specifications ............. 46
3 Readers/Writers Example 47
  3.1 Introduction .................................................................. 47
  3.2 The abstract specification ................................................. 48
    3.2.1 Specification $S_0$ ....................................................... 48
    3.2.2 Specification $S_{w_0}$ .................................................. 49
    3.2.3 Requirement $W_0$ ..................................................... 51
  3.3 The first development step .................................................. 51
    3.3.1 Specification $S_1$ ....................................................... 52
    3.3.2 Specification $S_{w_1}$ .................................................. 53
    3.3.3 Requirement $W_1$ ..................................................... 56
    3.3.4 $S_1$ relatively refines $S_0$ .......................................... 56
  3.4 The second development step .............................................. 59
    3.4.1 Specification $S_{w_2}$ .................................................. 60
    3.4.2 Specification $S_{w_2}$ .................................................. 62
A.11 Proof of Theorem 5 .................................................... 170
A.12 Proof of Theorem 6 .................................................... 171
A.13 Proof of Lemma 7 .................................................... 171
A.14 Proof of Lemma 8 .................................................... 171
A.15 Proof of Theorem 7 .................................................... 172
A.16 Proof of Lemma 9 .................................................... 173
A.17 Proof of Lemma 10 .................................................... 174
A.18 Proof of Lemma 11 .................................................... 175
A.19 Proof of Theorem 8 .................................................... 175
A.20 Proof of Theorem 9 .................................................... 176
A.21 Proof of Lemma 12 .................................................... 176
A.22 Proof of Lemma 13 .................................................... 177
List of Tables

2.1 Syntax of DTL ........................................ 16
2.2 Used abbreviations ..................................... 18
List of Figures

2.1 Computation of a machine. .............................................. 4
2.2 Concrete computation. .................................................. 5
2.3 Collapsed history. ....................................................... 11
2.4 Abstract machine ............................................................ 34
2.5 Concrete machine 1 ....................................................... 35
2.6 Concrete machine 2 ....................................................... 36
2.7 Transitions of $S_{c1}||S_{c2}$ ............................................. 40
3.1 Transitions of reader$^{0}$ .................................................. 49
3.2 Transitions of writer$^{0}$ .................................................. 50
3.3 Transitions of reader$^{1}$ .................................................. 54
3.4 Transitions of writer$^{1}$ .................................................. 55
3.5 Transitions of reader$^{2}$ .................................................. 63
3.6 Transitions of writer$^{2}$ .................................................. 65
3.7 Transitions of reader$^{3}$ .................................................. 75
3.8 Transitions of writer$^{3}$ .................................................. 77
3.9 Transitions of reader$^{4}$ .................................................. 86
3.10 Transitions of writer$^{4}$ ................................................ 89
4.1 Transitions of stable storage. .......................................... 98
4.2 Transitions of the physical disk. ..................................... 101
4.3 Transitions of the fail-stop detection layer. ......................... 108
4.4 Transitions of the relative composed system. ..................... 111
4.5 Transitions of the error recovery layer. ............................ 120
4.6 Transitions of the detection layer. ................................. 124
4.7 Transitions of the final implementation of stable storage. ...... 129
Chapter 1

Introduction

Current formal methods are far from solving the problems in software development. The simplest view of the formal paradigm is that one starts with a formal specification and subsequently decomposes this specification in sub Specifications which composed together form a correct refinement. These sub Specifications are decomposed into “finer” sub Specifications. This refinement process is continued until one gets sub Specifications for which an implementation can easily be given. This view is too idealistic in a number of respects. First of all, most Specifications of software are wrong (certainly most informal ones, unless they have been formally analyzed) and contain inconsistencies \[\text{PWT90}.\] Secondly, even if a formal Specification is produced, this is only after a number of approximation steps because writing a correct Specification is an even more difficult process than producing a correct implementation, and should therefore be structured, resulting in a number of increasingly less abstract layers with Specifications which tend to increase in detail (and therefore become less readable \[\text{LGdR79}\]). Thirdly, even an incorrect refinement step may be useful in the sense that from this incorrect refinement step one can sometimes easier derive the correct refinement step. This is especially the case with intricate algorithms such as those concerning specific strategies for solving the mutual exclusion problem. An interesting illustration of this third view is provided by E.W. Dijkstra’s “Tutorial on the split binary semaphore” \[\text{Dij79}\] in which he solves the readers/writers problem by subsequently improving incorrect refinement steps till they are correct. If this master of style prefers to approximate and finally arrive at his correct solution using formally “incorrect” intermediate stages, one certainly expects that a formally correct development process for that paradigm is difficult to find! The strategy described in \[\text{Dij79}\] is necessarily informal, reflecting the state of the art in 1979.

In Chapter 2 a dense time formalism is introduced for the specification and verification of refinement of systems based on \[\text{BKP86, DK90, KMP93, Sta84, Sta85, Sta88}\]. This formalism will be used to describe above strategy of incorrect intermediate stages. A dense time formalism is used because it allows one to deal with the stutter-problem (explained in section 2.1) and it enables one to express hiding of “internal” variables by existential quantification. Instead of using the assumption/commitment approach of \[\text{AL93a, AL93b, Jon83, MC81, PJ91, Pnu85, Sti88, St91, WD88, ZdBdR84, ZdRyEB84}\], unified in \[\text{XCC94, CC94}\], in order to achieve compositionality an event variable is used
that stores “compositionality information” like “this is a system step” or “this is an environment step” or “this is a communication step”. A merging operator, first version defined in [Ac93], based on the one defined in [CC94] is introduced to merge this “compositionality information” of the components into “compositionality information” of the composed system. The use of event variables has as second advantage that existing proof rules for refinement like those in [Lam91, KMP93] can easily be extended to our framework. The notion of relative refinement is introduced to handle “incorrect” development steps. The system specification is therefore extended by a requirement that extracts the “good” computations of the system. The refinement proof rules are extended to handle relative refinement so that the correct part of incorrect development steps can be proven correct.

In Chapter 3 we present Dijkstra’s development strategy of the readers/writers problem [Dij79] in our formalism. A preliminary version of this formalization, without proofs, appeared in [CKdR92] using the original formalism of [Sta84]. Our formalism preserves the flavour of the informal strategy in that it formalises Dijkstra’s argumentation in terms of incorrect approximations to a correct program and provides a formal criterion for recognising when a formally correct end product, the correct program, has finally been reached.

In Chapter 4 we present a formal development strategy for the development of certain fault tolerant systems using our notion of relative refinement. A preliminary version of this strategy appeared in [CdR93b, CdR93a] using the original formalism of [Sta84]. The formal strategy is as follows: one starts with an implementation for a specified fault tolerant system. This implementation contains some faults, i.e., the refinement step is incorrect because of these faults. It is however relative correct because when these faults don’t occur it is a correct implementation. In the next step we try to detect these faults, i.e., we construct a detection layer upon the previous implementation that stops that implementation when it detects an error caused by these faults. This is called a fail-stop implementation [LA90] and represents an improvement over the previous implementation because now at least the implementation stops on the occurrence of such a fault. The second implementation is also relatively correct because no occurrence of faults and the detection layer doesn’t detect any error due to a fault then the second implementation is correct. In the third approximation we recover these errors, i.e., we don’t stop anymore upon the detection of an error but merely recover the error by executing some special program that neutralizes that error. This third approximated refinement step is correct under the assumption that certain conditions are fulfilled, which exclude the occurrence of faults different from those whose errors are neutralized, i.e., it is again relative correct. This strategy is used for the development of a fault tolerant storage system, a so called stable storage.
Chapter 2
A Dense Model Formalism

2.1 Introduction

In this chapter we present a refinement method for reactive systems. A system is called reactive if it maintains some ongoing interaction with its environment, for example an operating system. This contrasts with transformational systems where from some input without further interaction output is produced. Because of this characteristic reactive systems should be described as sets of behaviours (histories). The underlying model for these behaviours is dense. The method which we present is based on the work of E.W. Stark [Sta84, Sta85, Sta88]. Here we present a framework which can model both CSP based and shared variable based concurrency, using the work of [BKP86, DK90, KMP93].

In section 2.2 reactive systems are specified by sets of histories together with a basis. A history is pair consisting of an event and a state function. The domains of these functions are the non-negative real numbers (the underlying dense model). The event function maps each non-negative real number to an event (an action occurring during the operation of the system and its environment) and the state function maps each real number to a state of the system and its environment. The intuition is that an occurrence of an action causes (potentially) a state change as illustrated in Figure 2.1. The basis is a pair consisting of an action basis and a process basis, where the action basis specifies the input and output channels over which the system communicates with its environment and the process basis specifies the local (only accessible by the system) and shared (accessible by both system and its environment) variables. Due to this basis composition of reactive systems corresponds to conjunction. Note that in for instance Lamport’s work on TLA [Lam91, Lam94, Lam] this is not always the case: $x := 1 \parallel x := 1$ must be modeled as disjunction because conjunction leads to a “one process” specification $x := 1$. In our model however, it can be modeled as conjunction because the specification of one component also contains environmental information, especially about the other component. With a “conjoining” operator the histories of both components are merged into a history of the composite one. This conjoining operator based on [CC94] corresponds in our model almost to conjunction and is actually an extended version of Aczel’s one [Acz83] because it also can handle CSP based concurrency whereas Aczel’s one can only handle shared variable
based concurrency.

States: \((s, x)\)

\[ (2, 2) \]
\[ (0, 1) \]
\[ (1, 2) \]
\[ (0, 0) \]

\( \mathbb{R} \geq 0 \)

: a state function

Events: \(\varepsilon\)

\[ i \]
\[ a^2 \]
\[ e \]
\[ \lambda \]

\( \mathbb{R} \geq 0 \)

: an event function

Figure 2.1: This picture illustrates the notion of state and event function, which together characterize the notion of computation of a machine. It illustrates the following computation: initially \((s, x) = (0, 0)\), the event \(a^2\) changes \(x\) into 1, i.e., \(s\) doesn’t change. In the interval \([0, t_1]\) there are only \(\lambda\) events. The event \(i\) at point \(t_2\) changes \((s, x)\) into \((1, 2)\), at point \(t_3\) the event \(e\) changes \(s\) into 2 and at point \(t_4\) the event \(i\) doesn’t change \(s\) or \(x\).

A notion of a machine is introduced for generating these histories, i.e., a history is a computation generated by a machine. With this machine notion only safety properties, i.e., sets of histories generated by a machine, of a system can be specified, so an extra condition on the computations of this machine is introduced for specifying liveness properties of the system.

The use of real numbers as domain for the event and state function handles the stutter problem. This problem, first observed by Lamport [Lam83, Lam89], is as follows. Given two behaviours of a system, let the first behaviour contain only consecutive snap-shots of the system that differ from each other whereas the second behaviour contains the same snap-shots but also some consecutive ones that are identical. This is called stuttering. From the viewpoint of an observer these behaviours are considered as equal. Consequently, any formalism that allows to distinguish between these behaviours is not abstract enough and has a power of discrimination which is too strong. An example of such a formalism is linear temporal logic with a next operator \(\bigcirc\). In the present formalism this excessive expressive power is avoided as follows: state changes caused by events happen only now and then, so that in between each two consecutive changes there are uncountably many instants of time at which \textit{nothing} happens. Consequently, it is impossible to count, or express, stutter steps. Furthermore the use of real numbers for defining the event and state function enables us to express hiding of variables as existential quantification and consider refinement as implication, even if there are more “states” on the abstract level than on the concrete level: let the history illustrated in Figure 2.1 be a history at the
### 2.1 Introduction

abstract level where \( x \) is the variable that should be hidden and let the history illustrated in Figure 2.2 be a history at the concrete level. The history of Figure 2.2 is a refinement of the history of Figure 2.1.

![States and Events Diagram](image)

Figure 2.2: This picture illustrates the following concrete computation: initially \( s = (0) \), the event \( a? \) doesn’t change \( s \), the event \( i \) changes \( s \) into 1, and the event \( e \) changes \( s \) into 2.

A dense time temporal logic DTL based on histories is introduced in section 2.2.2. This logic is based on [Sta84, Sta85, BKP86, DK90, KMP93]. A salient feature of the dense time temporal logic is the “immediately after” operator \( ' \), in a version which Lamport [Lam83] approves of, i.e., it is stutter insensitive. In this logic the notion of a machine and the condition on the computations of that machine will be expressed. It is also possible to express in this logic whether a system refines another system, i.e., the set of histories of the first system is a subset of the histories of the second one and the “observable” part of the abstract basis (i.e., observable from outside of the component) is equal to the “observable” part of the concrete basis. In our model initial stuttering is incorporated by default (cf. [DK90]) and refinement can be expressed using implication and existential quantification.

In section 2.3 the notions of composition and refinement of systems are defined. Firstly in terms of histories (semantically) and secondly in the dense time temporal logic DTL (syntactically). It is also investigated how composition relates to refinement, i.e., the notion of compositional refinement [ZCdR92] is given. Compositional refinement means intuitively that if the components of an abstract composed system are refined by the components of a concrete composed system then the abstract composed system is refined by the concrete composed system, i.e., refinement is preserved under composition.

Section 2.4 gives proof rules for refinement based on those given in [Lam91, KMP93]. These proof rules split the proof of refinement of systems into (1) a proof of refinement of the safety parts of the systems and (2) a proof of refinement of the liveness part of the systems.
Section 2.5 explains how the formalism can be used to describe relative (incorrect) refinement steps as discussed in Chapter 1. Also the notion of relative composition is introduced which intuitively means that only restricted parts of the components are composed together. The notion of compositional refinement of section 2.3 is extended to compositional relative refinement. The proof rules for refinement of section 2.4 are extended to handle relative refinement. These proof rules are used extensively in the readers/writers example of Chapter 3 and the stable storage example of Chapter 4.

2.2 Specification of Reactive Systems

This section explains how reactive systems can be specified. Firstly they will be specified at the semantical level, i.e., by sets of histories. A history intuitively specifies which event occurs at a particular point and in what state the system is at that particular point. Secondly reactive systems are specified using the dense time temporal logic DTL.

2.2.1 Semantic Specification of Reactive Systems

In [Sta84] a method for specifying reactive systems is introduced. Such systems are characterized by sets of histories. A history is a pair consisting of an event function and a state function. An event function records at each point (i.e., element of the positive reals, including zero) which event occurs. An event is an instantaneous occurrence of an action during the operation of a system, that can be generated by that system or its environment and that is of interest at the given level of abstraction. Four kinds of actions are distinguished:

1. communication actions $a$, $b$, i.e., actions that transmit information over a channel. A channel is a connection between the system and its environment.

2. system actions $i$, i.e., non-communication actions of the system.

3. environment actions $e$, i.e., non-communication actions of the environment.

4. silent actions $\lambda$, i.e., actions that don't influence the status of the system.

Event states are introduced in order to record which event occurs during the operation of the system. An event state is like the usual notion of state with the exception that instead of normal program variables event variables are used. An event state is defined formally in the following definition.

Definition 1 (Event variable and event state)

Let Chan denote the set of all channels. Let $\mathcal{E}$ denote the set of event variables with typical elements $\epsilon, \epsilon_0, \epsilon_1, \ldots$. Event variable $\epsilon$ will record which action occurs during the operation of the system. And the event variables $\epsilon_0, \epsilon_1, \ldots$ are auxiliary event variables recording which
2.2 Specification of Reactive Systems

actions occur in components of the system. Let $\mathcal{A}$ denote the set of actions, with typical elements $i$ (denoting system actions), $e$ (denoting environment actions), $a?, b!$ ... denoting respectively an input communication action over channel $a$ and an output communication action over channel $b$, and $\lambda$ denoting the silent action. An event state is a mapping $\delta$ from $\mathcal{E}$ to $\mathcal{A}$. Let $\Delta$ denote the set of all event states.

An state function records at each point (a non-negative real number) the process state, i.e. the usual notion of state of a system and its environment. In order to distinguish the normal variables from the event variables the normal variables are called here process variables. Three kind of process variables are distinguished:

1. **shared** process variables which are “shared” between a system and its environment, and
2. **local** process variables which are only accessible by a system.
3. **rigid** variables which are not changed by the system and its environment, i.e., which are used for specification purposes.

The process state is defined formally in the following definition.

**Definition 2 (Process variable and process state)**

A process state is a mapping from variables to values. Let $\mathcal{V}$ denote the set of shared variables with typical elements $u, \ldots$, and $\mathcal{X}$ the set of local variables ($\mathcal{V} \cap \mathcal{X} = \emptyset$) with typical elements $x, \ldots$, and $\mathcal{R}$ the set of rigid variables with typical elements $n, \ldots$. A state is a mapping $\sigma$ from $\mathcal{V} \cup \mathcal{X} \cup \mathcal{R}$ to the set of values $\text{Val}$. Let $\Sigma$ denote the set of all process states.

As already said above, event and state functions are mappings from the non-negative reals to, respectively event and process states. Because of this some requirements are needed in order to specify “reasonable” histories. Here reasonable is used in the sense that in a bounded interval only a finite number of non-silent actions and process state changes can occur. This requirement is called the finite variability condition [BKP86]. Next several notions for functions from $\mathbb{R}^{\geq 0}$ (the positive reals including 0) to some domain $D$ are introduced in order to define this requirement and to formally define the event and state functions.

**Definition 3 (Left and right constant, limit)**

Given function $f: \mathbb{R}^{\geq 0} \to D$.

- $f$ is called **left constant** at $t \in \mathbb{R}^{\geq 0}$, if there exists a real number $t_0$, $0 < t_0 < t$, such that $f(t_1) = d$ for all $t_1 \in (t_0, t)$. $d$ is called the left limit of $f$ at $t$, and is denoted by $\lim_{t_1 \to t} f(t_1)$.
- $f$ is called **right constant** at $t \in \mathbb{R}^{\geq 0}$, if there exists a real number $t_0$, $t_0 > t$, such that $f(t_1) = d$ for all $t_1 \in (t, t_0)$. $d$ is called the right limit of $f$ at $t$, and is denoted by $\lim_{t_1 \to t} f(t_1)$.

\(^1\)In this chapter we omit the value part of the communication, i.e., which value is transmitted, in order to ease the formalism a little bit. In the example of the stable storage we will use this value part although it is not formally introduce in this chapter.
Definition 4 (Left and right continuous, discontinuous)
Given function $f : \mathbb{R}^+ \to D$.

$f$ is called **left continuous**, if $f(t) = \lim_{t_i \to t} f(t_i)$ for every $t > 0$.

$f$ is called **right continuous**, if $f(t) = \lim_{t_i \to t} f(t_i)$ for every $t \geq 0$.

$f$ is called **discontinuous at** $t$, if $f(t) \neq \lim_{t_i \to t} f(t_i)$ or $f(t) \neq \lim_{t_i \to t} f(t_i)$.

$f$ is called **strongly discontinuous at** $t$, if $f(t) \neq \lim_{t_i \to t} f(t_i)$ and $f(t) \neq \lim_{t_i \to t} f(t_i)$.

Definition 5 (Finite variability)
Given function $f : \mathbb{R}^+ \to D$.

$f$ has the **finite variability** property iff $f$ has only finitely many points of discontinuity in any interval $[a, b]$, $0 \leq a \leq b$, $a, b \in \mathbb{R}^+$.

Now **event** and **state** functions can be defined. [DK90] states that initial stuttering is needed in order to express refinement in a logic with the help of existential quantification and implication. We must first define what stuttering, in the sense of [DK90], is in our setting. In our setting a stutter step is a step in which a non-communication action doesn’t change the state. So here this initial stuttering can included by requiring that in the first interval the event function has the constant value $\lambda$ and the state function remains constant there. Furthermore a state should remain constant for an interval of points in order to be observable. Also non-$\lambda$ events are considered to be single points. Another possibility would be the events to remain constant during an interval of points. The intuitive meaning of a history is that the points of non-$\lambda$ event occurrence mark the state changes. For the non-$\lambda$ events the question to be answered is: at which point of the interval should the state change take place? Answer: at the last point of the interval of the event. So for events only the last point of the interval is interesting because it marks the state change. So why consider an interval if only its last point is interesting? This is the explanation of the choice made here that the non-$\lambda$ events occur only at single points. This is captured by the following definitions.

Definition 6 (Restriction)
For $g : A_1 \to A_2$, $A_0 \subseteq A_1$ define $g|_{A_0} : A_0 \to A_2$ as $g|_{A_0}(x) = g(x)$ for $x \in A_0$. If $A_0$ is a set containing only one element $x$ then we will write $g|_{x}$ instead of $g|_{[x]}$.

For $g : A_1 \to (A_2 \to A_3)$, $A_0 \subseteq A_2$ define $g|_{A_0} : A_1 \to (A_0 \to A_3)$ as $g|_{A_0}(t)(x) = g(t)(x)$ for $x \in A_0$. Again if $A_0$ is a set containing only one element $x$ then we will write $g|_{x}$ instead of $g|_{[x]}$.

Definition 7 (Event function)
An **event function** is a function from $\mathbb{R}^+ \to \Sigma$, such that $\psi|_{t}^{e}$ has the finite variability condition, $\psi(0)(e) = \lambda$ (i.e. initial stuttering) and for all points $t$, $\psi$ is strongly discontinuous at $t$ iff $\psi(t)(e) \neq \lambda$ (i.e. an event function is almost constant $\lambda$). Let $\Psi$ denote the set of all event functions.

Figure 2.1 illustrates the notion of event function. At point $t_1$ event $a$ occurs, at point $t_2$ event $i$ occurs, at point $t_3$ event $e$ occurs, and at all other points event $\lambda$ occurs. Points $t_1$, $t_2$, and $t_3$ are here the strongly discontinuous points.
2.2 Specification of Reactive Systems

Definition 8 (State function)
A state function $\theta$ is a left continuous function from $\mathbb{R}^{\geq 0}$ to $\Sigma$ such that for all $n \in \mathbb{N}$ and $t \in \mathbb{R}^{\geq 0}$, $\theta(t)(n) = \theta(0)(n)$ (i.e., the rigid variables don’t change at all), and for all $x \in \mathfrak{A} \cup \mathfrak{X}$, $\theta[x]$ satisfies the finite variability property and $\theta[x](0)(x) = \lim_{t \to 0+} \theta[x](t)(x)$ (i.e., initial stuttering). Let $\Theta$ denote the set of all state functions.

Figure 2.1 illustrates the notion of state function. In interval $[0, t_1]$ the system is in state $(s, x) = (0, 0)$, in interval $(t_1, t_2]$ in state $(s, x) = (0, 1)$, in interval $(t_2, t_3]$ in state $(s, x) = (1, 2)$ and in interval $(t_3, \infty]$ in state $(s, x) = (2, 2)$. The event $i$ at $t_4$ is an illustration of a non-$\lambda$ stutter step.

The following definition combines the notions of state function and event function into the notion of history. Two requirements are imposed on the combination of event and state function in order to be a history. The first requirement is that silent actions don’t give rise to process state changes. The second requirement is that communication actions don’t change the shared variables; this requirement is imposed in order to model CSP [Hoa84] like processes.

Definition 9 (History)
A history $h$ is a pair $\langle \psi, \theta \rangle$, where $\psi$ is an event function and $\theta$ is a state function s.t. a $\lambda$ action doesn’t change the values of variables from $\mathfrak{A} \cup \mathfrak{X}$, i.e.:

$$\forall t : \psi(t)(e) = \lambda \rightarrow \theta(t) = \lim_{t \to t_1} \theta(t_1)$$

and a communication action doesn’t change the values of shared variables, i.e.:

$$\forall t : \psi(t)(e) = a? \rightarrow \theta(t) |_{\mathfrak{A}} = \lim_{t \to t_1} \theta(t_1) |_{\mathfrak{A}}$$
$$\forall t : \psi(t)(e) = a! \rightarrow \theta(t) |_{\mathfrak{B}} = \lim_{t \to t_1} \theta(t_1) |_{\mathfrak{B}}$$

Let $\mathcal{H}$ denote the set of all histories.

The following definition defines when a history is stutter equivalent to another history. A history collapse function is introduced that takes a history and collapses it in such a way that the non-stutter steps only occur at discrete points (elements of $\mathbb{N}$) and at all remaining points stutter steps occur. Also a restricted version of the history stutter equivalence relation is defined, namely, restricted to the process state information. The last one will be used to define a “process state history stutter insensitive” logic DTL. This logic will be restricted to a special kind of formulæ in order to obtain the “history stutter insensitive” logic.

Definition 10 (History collapse, stutter equivalent)
Given history $h \in \mathcal{H}$, the history collapse denoted $\tau_k(h)$ is a function from $\mathcal{H}$ to $\mathcal{H}$ defined as $\tau_k(h) \triangleq h \circ \text{di}(h)$ where $\text{di}(h)$ is the discretization bijection for $h$ from $\mathbb{R}^{\geq 0}$ to $\mathbb{R}^{\geq 0}$ and is defined as follows:

Let $\text{tt}(h, k)$ be the function from $\mathcal{H} \times \mathbb{N}$ to $\mathbb{R}^{\geq 0}$ that gives the point in $\mathbb{R}^{\geq 0}$ of the $k$-th change in $h$, formally:

$$\text{tt}(h, 0) \triangleq 0$$
for $k > 0$,

$$\text{tt}(h, k) \triangleq \min(t : t > \text{tt}(h, k - 1) \wedge \psi(t)(e) \notin \{\lambda, i, e\} \vee \theta(t) \neq \lim_{t \to t_{k-1}} \theta(t_{k-1}))$$

9
Let \( nn(h) \) denote the number of non-stutter points of \( h \). Then the discretization bijection \( \text{di}(h) \) for \( h \) is defined as follows:

\[
\text{di}(h)(t) \overset{\Delta}{=} \begin{cases} 
\text{tt}(h, k) + (t - k) \ast (\text{tt}(h, k + 1) - \text{tt}(h, k)) & nn(h) < \infty \land 0 \leq k < nn(h) \\
\text{tt}(h, k) + (t - k) & nn(h) < \infty \land k = nn(h) \land k \leq t \\
\text{tt}(h, k) + (t - k) \ast (\text{tt}(h, k + 1) - \text{tt}(h, k)) & nn(h) = \infty \land 0 \leq k \land k \leq t \leq k + 1 
\end{cases}
\]

The inverse discretization of \( h \) is denoted \( \text{di}^{-1}(h) \).

Given histories \( h_0, h_1 \in \mathcal{H}, h_0 \) is history stutter equivalent to \( h_1 \) denoted \( h_0 \simeq_h h_1 \) iff

\[
nn(h_0) = nn(h_1), \quad \text{and} \quad \theta_{in}(h_0) = \theta_{in}(h_1), \quad \text{and} \quad \psi_{in}(h_0)(k) = \psi_{in}(h_1)(k), \quad k \leq \text{nn}(h_0)
\]

i.e., the number of non-stutter steps should be equal, the state information should be equal in both collapsed histories and the event information should be equal in the points of non-stuttering. A restricted version of the history stutter equivalence relation is the one that considers only the process state information, i.e., \( h_0 \) is history process state stutter equivalent to \( h_1 \) denoted \( h_0 \simeq_{h_p} h_1 \) iff

\[
nn(h_0) = nn(h_1), \quad \text{and} \quad \theta_{in}(h_0) = \theta_{in}(h_1),
\]

Application of above definition to the history of Figure 2.1 results in: \( \text{tt}(h, 0) = 0, \text{tt}(h, 1) = t_1, \text{tt}(h, 2) = t_2, \text{tt}(h, 3) = t_3, \text{and tt}(h, k) = \infty \) for \( k > 3 \) and \( nn(h) = 3 \). The discretization function \( \text{di}(h)t \) is as follows:

\[
\begin{cases} 
t \ast t_1 & 0 \leq t \leq 1 \\
t_1 + (t - 1) \ast (t_2 - t_1) & 1 \leq t \leq 2 \\
t_2 + (t - 2) \ast (t_3 - t_2) & 2 \leq t \leq 3 \\
t_3 + (t - 3) & 3 \leq t
\end{cases}
\]

The collapsed history \( \text{i}_b(h_0) \) is illustrated in Figure 2.3.

The following theorem relates histories to a special kind of infinite sequences of pairs of event and process states, in which sequences start with an \( \lambda \) action, followed by possibly stuttering actions, then followed by exactly one non-stuttering action etc. Furthermore should every non-\( \lambda \) event be surrounded by \( \lambda \)-events. These kind of sequences are inspired by those defined in [KMP93]. For these kind of sequences a sequence collapse is defined that removes all finite stuttering; with the help of this collapse operator the sequence stutter equivalence operator is defined.

**Definition 11 (Infinite sequences)**

Define a sequence element as a pair \( (\delta, \sigma) \) of an event and a process state. Let \( sel_i \ (i \geq 0) \) be
2.2 Specification of Reactive Systems

The pair \((\delta_i, \sigma_i)\) then the sequence \(\text{seq}\) is a infinite sequence of the form \(\text{seq}_0 \text{seq}_0 \text{seq}_1 \text{seq}_1 \ldots\) where \(\text{seq}_i = (\delta_i, \sigma_i)\) is such that

\[
\delta_i(\epsilon) \neq \lambda,
\]

\[
(\delta_i(\epsilon) = \text{a?} \lor \delta_i(\epsilon) = \text{a!}) \rightarrow \sigma_i \mid_{\sigma_i} = \sigma_{i+1} \mid_{\sigma_{i+1}}
\]

and \(\text{seq}\) is a sequence of the form \((\delta_i, \sigma_i)^n((\delta_{i+2}, \sigma_{i+2})^{n_i}((\delta_{i+4}, \sigma_{i+4})^{k_i})^{l_i})\) where \(n_i > 0, k_i > 0\) and \(l_i \geq 0\), and \(\delta_{i+1}(\epsilon) = \lambda\) and \(\delta_{i+1}(\epsilon) \in \{\text{i, e}\}\).

Let \(\text{SEQ}\) denote the set of all such sequences. Let \(\text{seq}\) be a sequence of the above form then \(\text{seq}'\) is the stutter free sequence obtained from \(\text{seq}\) by deleting all finite stuttering from \(\text{seq}\). Formally: Let \(\text{ns}(\text{seq})\) denote the number of non-stutter steps in \(\text{seq}\), if \(\text{ns}(\text{seq}) = \infty:\)

\[
\begin{align*}
\text{seq}'_{i+1} & = \text{seq}_i \quad 0 \leq i \\
\text{seq}'_{i+1} & = (\delta_{i+1}, \sigma_{i+1}) \quad 0 \leq i
\end{align*}
\]

if \(\text{ns}(\text{seq}) < \infty:\)

\[
\begin{align*}
\text{seq}'_{i+1} & = \text{seq}_i \quad 0 \leq i < \text{ns}(\text{seq}) \\
\text{seq}'_{i+1} & = (\delta_{i+1}, \sigma_{i+1}) \quad k = \text{ns}(\text{seq}) \land k \leq i \\
\text{seq}'_{i+1} & = (\delta_{i+1}, \sigma_{i+1}) \quad 0 \leq i \leq \text{ns}(\text{seq}) \\
\text{seq}'_{i+1} & = (\delta_{k+1}, \sigma_{k+1}) \quad k = \text{ns}(\text{seq}) \land k \leq i
\end{align*}
\]

Let \(\text{seq}_0\) and \(\text{seq}_1\) be sequences then \(\text{seq}_0\) is stutter equivalent to \(\text{seq}_1\) denoted \(\text{seq}_0 \simeq \text{seq}_1\) if: let \(\text{seq}_0' = (\text{seq}_0')_{i \geq 0}\) and \(\text{seq}_1' = (\text{seq}_1')_{i \geq 0}\),

\[
\begin{align*}
\text{ns}(\text{seq}_0) & = \text{ns}(\text{seq}_1) \quad \text{and} \\
\sigma_i^0 & = \sigma_i^1 \quad \text{and} \\
\delta_i^0(\epsilon) & = \delta_i^1(\epsilon)
\end{align*}
\]
The relationship between the sequences and histories is that there exists a function from the stutter equivalence classes of histories to the stutter equivalence classes of sequences and a function from the stutter equivalence classes of sequences to the stutter equivalence classes of histories.

**Theorem 1 (Relationship between histories and infinite sequences)**

Let \( h \in \mathcal{H} / \approx_h \) then \((\text{sel}_i)_{i \geq 0} \in \text{SEQ} / \simeq_s\) where \( \text{sel}_i \) is as follows:

If \( nn(h) < \infty \):

\[
\begin{align*}
\text{sel}_{2+i+1} &= h(i) & 0 \leq i < nn(h) \\
\text{sel}_{2+i} &= \lim_{t \to i} h(t_1) & 0 \leq i \leq nn(h) \\
\text{sel}_{2+i+1} &= \lim_{k \to i} h(t_1) & k = nn(h) \land k \leq i \\
\text{sel}_{2+i} &= \lim_{k \to i} h(t_1) & k = nn(h) \land k \leq i 
\end{align*}
\]

If \( nn(h) = \infty \):

\[
\begin{align*}
\text{sel}_{2+i+1} &= h(i) & 0 \leq i \\
\text{sel}_{2+i} &= \lim_{t \to i} h(t_1) & 0 \leq i
\end{align*}
\]

Let \( \text{seq} = (\text{sel}_i)_{i \geq 0} \in \text{SEQ} / \simeq_s\) then \( h \in \mathcal{H} / \approx_h \) where \( h \) is as follows:

If \( ns(\text{seq}) < \infty \):

\[
\begin{align*}
h(0) &= \text{sel}_0 \\
h(t) &= \text{sel}_{2+i-1} & t \in \mathbb{N} \land 0 < t \leq ns(\text{seq}) \\
h(t) &= \text{sel}_{2+i} & t \in \mathbb{N} \land t > ns(\text{seq}) \\
h(t) &= \text{sel}_{2+i} & i < t < i+1
\end{align*}
\]

If \( ns(\text{seq}) = \infty \):

\[
\begin{align*}
h(0) &= \text{sel}_0 \\
h(t) &= \text{sel}_{2+i-1} & t \in \mathbb{N} \\
h(t) &= \text{sel}_{2+i} & i < t < i+1
\end{align*}
\]

The following sequence corresponds to the history of figure 2.3:

\[
\text{seq} = (\delta_i, \sigma_i)_{i \geq 0},
\]

\[
\begin{align*}
\delta_0(e) &= \lambda & \sigma_0(s) &= 0 & \sigma_0(x) &= 0 \\
\delta_1(e) &= \text{a} & \sigma_1(s) &= 0 & \sigma_1(x) &= 0 \\
\delta_2(e) &= \lambda & \sigma_2(s) &= 0 & \sigma_2(x) &= 1 \\
\delta_3(e) &= \text{i} & \sigma_3(s) &= 0 & \sigma_3(x) &= 1 \\
\delta_4(e) &= \lambda & \sigma_4(s) &= 1 & \sigma_4(x) &= 2 \\
\delta_5(e) &= \text{e} & \sigma_5(s) &= 1 & \sigma_5(x) &= 2 \\
\delta_i(e) &= \lambda & \sigma_i(s) &= 2 & \sigma_i(x) &= 2 & i > 5
\end{align*}
\]

The basis is a pair consisting of a *process basis*, specifying the local and shared variables of the system, and a *action basis* which specifies the input and output communication channels of a system. The following definition introduces basis and history sets that constrain a specific process basis, i.e., specific sets of shared variables and local variables are constrained to change in specific ways, the variables outside this process basis can change without restriction, with exception of the rigid variables which do not change at all.
2.2 Specification of Reactive Systems

Definition 12 (Basis, history set constraining a basis)
A basis (denoted by $B$) is a pair $(B^A, B^P)$, where $B^A$ (called action basis) is a pair $(\text{In}, \text{Out})$ where In is a set of input communication channels and Out is a set of output communication channels, and where $B^P$ (called process basis) is a tuple $(V, X)$ where $V$ a finite set of shared variables and $X$ a finite set of local variables.

Given a history $h \in \mathcal{H}$ and process basis $B^P$ then the process basis restriction of $h$ denoted $h|_{B^P}$ is defined as $\langle \psi, \theta \rangle|_{V \cup X}$.

Given a set of histories $H$ and process basis $B^P$ then $H$ is constrained by $B^P$ iff $\forall h_1, h_2 \in \mathcal{H} : h_1|_{B^P} = h_2|_{B^P} \rightarrow (h_1 \leftrightarrow h_2 \in H)$.

The following definition introduces the notion of history specification which is a pair consisting of a basis and a set of histories constraining the process basis.

Definition 13 (History specification of a system)
A history specification of a system (denoted $S$) is a pair $(B, H)$ where $B$ is a basis and $H$ is a set of histories constraining process basis $B^P$ such that an environment action $e$ doesn’t change the local variables of the system:

$$\forall t : \psi(t)(e) = e \rightarrow \theta(t)|_{X} = \lim_{t \rightarrow h} \theta(t)|_{X}$$

The following definition introduces several notions from topology ([Wri87]) needed for the definition of safety and liveness sets of histories. These definitions of safety and liveness are based on those of [AS85]. Informally a safety set of histories consists of histories where nothing “bad” happens and a liveness set of histories consists of histories where something “good” eventually happens.

Definition 14 (Safety and liveness set)
Let $H$ be a set of histories and $h \in \mathcal{H}$.

- The prefix of $h$ of length $t$ denoted $h|_t$ is defined as

$$h|_t (t_0) \triangleq \begin{cases} \langle \psi(t_0), \theta(t_0) \rangle & 0 \leq t_0 \leq t \\ \langle \psi(0), \theta(t) \rangle & t_0 > t \end{cases}$$

Thus for $t_0 > t$ only stutter actions occur in $h|_t$.

- The distance function $d$ from $\mathcal{H} \times \mathcal{H} \rightarrow \mathbb{R}^{\geq 0}$ is defined as:

$$d(h_1, h_2) \triangleq \begin{cases} 0 & \text{if } h_1 = h_2 \\ 1 & \text{if } h_1(0) \neq h_2(0) \\ 2^{-\sup \{ t \in \mathbb{R}^{\geq 0} : h_1(t) = h_2(t) \}} & \text{otherwise} \end{cases}$$

$(\mathcal{H}, d)$ is a metric space.

- $H$ is called $d$-open iff

$$\forall h \in H : \exists \varepsilon > 0 : \forall h_1 : d(h, h_1) < \varepsilon \rightarrow h_1 \in H$$

13
A Dense Model Formalism

- The topology with \( \{ H \subseteq \mathcal{H} \mid H \text{ is } d\text{-open} \} \) as its basis is called the \( d \) induced topology of \((\mathcal{H}, d)\) denoted \( \tau_d \).
- \( H \) is called a \( \tau_d \)-environment of \( h \) iff
  \[ \exists H_1 \in \tau_d : h \in H_1 \land H_1 \subseteq H \]
- The interior of \( H \) denoted \( \text{in}(H) \) is defined as
  \[ \{ h \in \mathcal{H} \mid H \text{ is a } \tau_d \text{ environment of } h \} \]
- The closure of \( H \) denoted \( \text{cl}(H) \) is defined as \( \mathcal{H} \setminus (\text{in}(\mathcal{H} \setminus H)) \).
- \( H \) is a safety set iff \( \text{cl}(H) = H \).
- \( H \) is a liveness set iff \( \text{cl}(H) = \mathcal{H} \).

Note: the only set that is both a safety and a liveness set is \( \mathcal{H} \) [AS85].

A specification method for systems that uses only sets of histories is not attractive. Therefore the notion of machine is introduced. A machine consists of a set of states and a state-transition relation. The intention is that the set of computations (i.e. histories) of a machine associated to a system should correspond to the history specification of this system. A machine however can only generate safety sets of histories [AS87]. Therefore, a liveness set is specified as a condition on the set of computations (histories) of a machine. Next the formal definition of a machine is given.

**Definition 15 (Machine)**

The machine specification \( M \) of a system is a triple \((B, I, T)\) where:

- \( B \): the basis of \( M \); a tuple \(((\text{In, Out}), (V, X))\). Note: the shared variables will be printed in bold faced style in order to distinguish them from the local variables.
- \( I \): a non-empty subset of \( \Sigma \), the set of initial states, such that
  \[ \forall \sigma_0, \sigma_1 \in \Sigma : (\sigma_0|_{VUX} = \sigma_1|_{VUX}) \rightarrow (\sigma_0 \in I \iff \sigma_1 \in I), \text{ i.e., it constrains the variables from } V \cup X \text{ only.} \]
- \( T \): the state-transition relation (finite), \( T \subseteq \Delta \times \Sigma^2 \), such that
  - \( \forall \sigma_0, \sigma_1 \in \Sigma, \delta \in \Delta : \langle \delta, \sigma_0, \sigma_1 \rangle \in T \rightarrow \sigma_0|_B = \sigma_1|_B, \text{ i.e., the rigid variables don’t change at all.} \)
  - \( \forall \sigma_0, \sigma_1, \sigma_2, \sigma_3 \in \Sigma, \delta \in \Delta : (\sigma_0|_{VUX} = \sigma_2|_{VUX} \land \sigma_1|_{VUX} = \sigma_3|_{VUX}) \rightarrow (\langle \delta, \sigma_0, \sigma_1 \rangle \in T \iff \langle \delta, \sigma_1, \sigma_3 \rangle \in T), \text{ i.e. } T \text{ constrains } B^P \text{ only.} \)
  - \( \forall \sigma_0, \sigma_1 \in \Sigma, \delta \in \Delta : (\langle \delta, \sigma_0, \sigma_1 \rangle \in T \land (\delta(\epsilon) = \text{a?} \lor \delta(\epsilon) = \text{a!})) \rightarrow \sigma_0|_V = \sigma_1|_V, \text{ i.e., a communication action doesn’t change the values of shared variables, and} \)
  - \( \forall \sigma_0, \sigma_1 \in \Sigma, \delta \in \Delta : (\langle \delta, \sigma_0, \sigma_1 \rangle \in T \land \delta(\epsilon) = \epsilon) \rightarrow \sigma_0|_K = \sigma_1|_K, \text{ i.e., an environment action doesn’t change the values of local variables of the system.} \)
2.2 Specification of Reactive Systems

- \( \forall \sigma_0, \sigma_1 \in \Sigma, \delta \in \Delta : (\delta, \sigma_0, \sigma_1) \in T \rightarrow (\delta(e) \notin \{\lambda, i, e\} \land \sigma_0 \neq \sigma_1) \), i.e., no stutter transitions are specified.

The following example is an illustration of the notion of machine.

Example 1

\( M = (B, I, T) \) where:

1. **Basis:** \( B = (\{\text{In}, \text{Out}\}, \{\text{V}, \text{X}\}) \) where

   \[
   \begin{align*}
   \text{In} & \triangleq \{a\} \\
   \text{Out} & \triangleq \emptyset \\
   \text{V} & \triangleq \{v\} \\
   \text{X} & \triangleq \{u\}
   \end{align*}
   \]

2. **Initial States:**

   \( I : \{\sigma \in \Sigma \mid \sigma(u) = 0 \land \sigma(v) = 0\} \)

3. **Transitions:**

   \( T : \)

   \[
   \{(\delta, \sigma_0, \sigma_1) \in \Delta \times \Sigma^2 \}
   \]

   \[
   (a) (\delta(e) = a? \land \sigma_0(u) = 0 \land \sigma_1(u) = 1 \land \sigma_1(v) = \sigma_0(v)) \quad \text{or}
   \]

   \[
   (b) (\delta(e) = i \land \sigma_0(u) = 1 \land \sigma_0(v) = 1 \land \sigma_1(u) = 2 \land \sigma_1(v) = 0) \quad \text{or}
   \]

   \[
   (c) (\delta(e) = e \land \sigma_1(u) = \sigma_0(u) \land \sigma_1(v) = \sigma_0(v) + 1)
   \]

The concepts of event and state functions are related by the notion of computation of a machine \( M \). A computation of \( M \) intuitively expresses that an event function and a state function \( \delta(t) \) together in that at any point \( t \) any triple consisting of (1) the event occurring at \( t \), (2) the state just before and including \( t \), and (3) the state just after \( t \), belongs to the state transition relation of \( M \) (see fig. 2.1). Because a state-transition relations don’t contain stutter steps but histories do, a set of stutter transitions should be defined in order to relate machine computations to histories.

**Definition 16 (Computation)**

Let \( h = \langle \psi, \theta \rangle \in \mathcal{H} \) and \( t \in \mathbb{R}^{\geq 0} \), then define the step occurring at \( t \) in \( h \) by:

\[
\text{Step}_h(t) = \langle \psi(t), \theta(t), \lim_{l \to t} \theta(l) \rangle.
\]

Define the set of stutter steps denoted \( \text{STU} \) as \( \{(\delta, \sigma_0, \sigma_1) \mid \delta(e) \in \{\lambda, i, e\} \land \sigma_0 = \sigma_1\} \).

A computation of a machine \( M = (B, I, T) \) is a history \( h = \langle \psi, \theta \rangle \in \mathcal{H} \) such that:

\[
\theta(0) \in I \quad \text{and}
\forall t : \text{Step}_h(t) \in T \lor \text{Step}_h(t) \in \text{STU}.
\]

Let the set of all computations of \( M \) be defined as:

\[
\text{Comp}(M) \triangleq \{ h \in \mathcal{H} \mid h \text{ is a computation of } M \}.
\]
Lemma 1 (Machine is safety)
Given machine \( M = (B, I, T) \) then
\[
\text{Comp}(M) \text{ is a safety set.}
\]

A proof of this lemma is given in [AL91] (it is also repeated in the appendix). The machine specification of a system now consists of a machine \( M \) and a set of histories \( L \) constraining the basis of this machine such that the closure of the intersection of \( \text{Comp}(M) \) and \( L \) equals \( \text{Comp}(M) \). This is the machine closedness property of a system specification introduced in [AFK88, AL91]. Let \( A \rightarrow B \) denote \( A \cup B \). By a result of [AS85] every set of histories can be written as the intersection of a safety set and a liveness set namely \( cl(\text{Comp}(M) \cap L) \cap cl(\text{Comp}(M) \cap L) \rightarrow (\text{Comp}(M) \cap L) \). By the machine closedness property this can be written as \( \text{Comp}(M) \cap (\text{Comp}(M) \rightarrow L) \). This means that \( \text{Comp}(M) \) specifies the safety properties and \( \text{Comp}(M) \rightarrow L \) the liveness properties of the system.

Definition 17 (Machine specification of a system)
A machine specification \( S \) of a system is a pair \( (B, \text{Comp}(M) \cap L) \) where \( M \) is a machine with basis \( B \) and \( L \) a set of histories constraining only \( B \) such that \( cl(\text{Comp}(M) \cap L) = \text{Comp}(M) \). The set of computations of \( S \), denoted \( \text{Comp}(S) \), is defined as \( \text{Comp}(M) \cap L \).

2.2.2 DTL Specification of Reactive Systems
As mentioned above, the local properties are described by a machine and the liveness properties are described as a set of histories. The dense time temporal logic DTL is introduced to describe both kind of properties. The one used here is a mixture of dense time temporal logics defined in [Sta84, Sta85, BKP86, DK90, KMP93].

Definition 18 (Syntax of DTL)
The syntax of DTL is defined in Table 2.1 where value \( \mu \in \text{Val} \), rigid variable \( n \in \mathbb{R} \), observable variable \( v \in \mathcal{V} \), local variable \( x \in \mathcal{X} \), event variable \( \epsilon \in \mathcal{E} \) and channel \( a \in \text{Chan} \).

<table>
<thead>
<tr>
<th>Table 2.1: Syntax of DTL</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Rigid Expressions</strong></td>
</tr>
<tr>
<td>( rexp ::= \mu \mid n \mid n' \mid n \mid rexp_1 + rexp_2 \mid \ldots )</td>
</tr>
<tr>
<td><strong>Expressions</strong></td>
</tr>
<tr>
<td>( exp ::= rexp \mid v \mid v' \mid v \mid x \mid x' \mid x \mid exp_1 + exp_2 \mid \ldots )</td>
</tr>
<tr>
<td><strong>Event Expressions</strong></td>
</tr>
<tr>
<td>( evexp ::= a? \mid a! \mid i \mid e \mid \lambda \mid \epsilon \mid \epsilon' \mid \epsilon )</td>
</tr>
<tr>
<td><strong>Temporal formulae</strong></td>
</tr>
<tr>
<td>( p ::= \text{true} \mid exp_1 = exp_2 \mid exp_1 &lt; exp_2 \mid evexp_1 = evexp_2 \mid \neg p \mid p_1 \lor p_2 )</td>
</tr>
<tr>
<td>( p_1 \Delta p_2 \mid p_1 \tilde{S} p_2 \mid \exists x.p \mid \exists e.p \mid \exists n.p )</td>
</tr>
</tbody>
</table>

The informal semantics of the most interesting constructs are as follows:
- \( \Delta x \) denotes the previous value of \( x \).
2.2 Specification of Reactive Systems

- $x$ denotes the current value of $x$,
- $x'$ denotes the next value of $x$,
- $\epsilon$ denotes the current action value of $\epsilon$,
- $\dot{\epsilon}$ denotes the previous action value of $\epsilon$,
- $\epsilon'$ denotes the next action value of $\epsilon$,
- $p_1 \mathcal{U} p_2$ denotes strict (present not included in the future) until operator from temporal logic,
- $p_1 \mathcal{S} p_2$ denotes strict (present not included in the past) since operator from temporal logic,
- $\exists x. p$ denotes existential quantification over local variable $x$ of $p$, i.e., hiding,
- $\exists \epsilon. p$ denotes existential quantification over event variable $\epsilon$ of $p$, i.e., hiding.

A state expression is an expression without any primed variables. A state formula is a formula build from state expressions without $\mathcal{U}$ and $\mathcal{S}$ operators.

Table 2.2 lists some frequently used abbreviations: The following example 2 gives some DTL formulae

**Example 2 (Some DTL formulae)**

\[
(\epsilon = a_0 \land x = 0 \land x' = 1) \text{ (a state-transition)},
\]
\[
\Box x > 0 \text{ (a safety property)},
\]
and \[
\Box(x = 0 \rightarrow \Box x > 0) \text{ (a liveness property)}.
\]

Before we give the semantics of DTL formulae we define for a variable $x$ (local process or event) the $x$-variant of a history.

**Definition 19 (x-variant, $\epsilon$-variant and $n$-variant of a history)**

Let $h, h_1 \in \mathcal{H}$.

- Let $x \in \mathcal{X}$ then $h_1$ is a $x$-variant of $h$ if $\psi_1 = \psi$ and $\theta_1|_{x \subseteq \mathcal{X}} = \theta_1|_{x \subseteq \mathcal{X}}$.
- Let $X \subseteq \mathcal{X}$ then $h_1$ is a $X$-variant of $h$ if $\psi_1 = \psi$ and $\theta_1|_{x \subseteq \mathcal{X}} = \theta_1|_{X \subseteq \mathcal{X}}$.
- Let $\epsilon \in \mathcal{E}$ then $h_1$ is a $\epsilon$-variant of $h$ if $\psi_1|_{\epsilon \in \mathcal{E}} = \psi|_{\epsilon \in \mathcal{E}}$ and $\theta_1 = \theta$.
- Let $n \in \mathbb{N}$ then $h_1$ is a $n$-variant of $h$ if $\psi_1 = \psi$ and $\theta_1|_{n \subseteq \mathcal{N}} = \theta_1|_{n \subseteq \mathcal{N}}$.

In the following definition the semantics of DTL is given without using valuation functions for expressions, i.e., this valuation function is implicitly defined by $\models$. By convention, boolean values are not explicitly denoted, i.e., we shall write $(h, t) \models \text{true}$ rather than $(h, t) \models \text{true} \triangleq tt.$

**Definition 20 (Semantics of DTL)**

Let $h \in \mathcal{H}, t \in \mathbb{R}_{\geq 0}, n \in \mathbb{N}, \nu \in \mathcal{V}, x \in \mathcal{X}$, and $\epsilon \in \mathcal{E}$.

- $(h, t) \models \mu \triangleq \mu$,
\begin{table}[h]
\centering
\begin{tabular}{|c|l|}
\hline
\textbf{false} & $\triangleq \neg \text{true}$ \\
p_1 \rightarrow p_2 & $\triangleq \neg p_1 \lor p_2$ \\
p_1 \land p_2 & $\triangleq (\neg p_1 \lor \neg p_2)$ \\
p_1 \leftrightarrow p_2 & $\triangleq (p_1 \rightarrow p_2) \land (p_1 \leftarrow p_2)$ \\
$\forall x. p$ & $\triangleq \neg \exists x. \neg p$ \\
$\exists X.p$ & $\triangleq \exists x_0, \ldots, \exists x_m. p$ \\
\hline
$\Diamond p$ & $\triangleq \text{true} \uparrow p$ \\
$\Box p$ & $\triangleq \neg \Diamond \neg p$ \\
$\bigcirc p$ & $\triangleq p \hat{\uparrow} \text{true}$ \\
$\Diamond p$ & $\triangleq p \hat{\lor} \Diamond p$ \\
$\Box p$ & $\triangleq p \land \Box p$ \\
p_1 \mathcal{U} p_2 & $\triangleq p_1 \lor (p_1 \land (p_1 \hat{\uparrow} p_2))$ \\
\hline
$\Diamond p$ & $\triangleq \text{true} \mathcal{S} p$ \\
$\Box p$ & $\triangleq \neg \Diamond \neg p$ \\
$\bigcirc p$ & $\triangleq p \hat{\mathcal{S}} \text{true}$ \\
$\Diamond p$ & $\triangleq p \hat{\lor} \Diamond p$ \\
$\Box p$ & $\triangleq p \land \Box p$ \\
p_1 \mathcal{S} p_2 & $\triangleq p_1 \lor (p_1 \land (p_1 \hat{\mathcal{S}} p_2))$ \\
\hline
p_1 \Rightarrow p_2 & $\triangleq \Box (p_1 \rightarrow p_2)$ \\
p_1 \Leftarrow p_2 & $\triangleq \Box (p_1 \leftarrow p_2)$ \\
\hline
\end{tabular}
\caption{Used abbreviations}
\end{table}

\begin{itemize}
\item \((h, t) \models n \triangleq \theta(0)(n),\)
\item \((h, t) \models n' \triangleq \theta(0)(n),\)
\item \((h, t) \models \neg n \triangleq \theta(0)(n),\)
\item \((h, t) \models x \triangleq \theta(t)(x),\)
\item \((h, t) \models \mathbf{v} \triangleq \theta(t)(\mathbf{v}),\)
\item \((h, 0) \models \mathbf{v} \triangleq \theta(0)(\mathbf{v}),\)
\item \((h, 0) \models x' \triangleq \lim_{t \downarrow t_1}(\theta(t_1)(x))\)
\item \((h, 0) \models \mathbf{v} \triangleq \theta(0)(\mathbf{v}),\)
\item \((h, 0) \models x' \triangleq \lim_{t \downarrow t_1}(\theta(t_1)(x))\)
\end{itemize}
2.2 Specification of Reactive Systems

- \((h, t) \models v' \triangleq \lim_{t \to t_1} \theta(t_1)(v)\)
- \((h, t) \models \exp_1 + \exp_2 \triangleq \exp_1 + (h, t) \models \exp_2\)
- \((h, t) \models \exp_1 - \exp_2 \triangleq \exp_1 - (h, t) \models \exp_2\)
- \((h, t) \models a ? \triangleq a?\)
- \((h, t) \models a! \triangleq a!\)
- \((h, t) \models i \triangleq i\)
- \((h, t) \models e \triangleq e\)
- \((h, t) \models \lambda \triangleq \lambda\)
- \((h, t) \models \epsilon \triangleq \psi(t)(e)\)
- \((h, 0) \models \epsilon \triangleq \psi(0)(e)\)
  \(t > 0: (h, t) \models \epsilon \triangleq \lim_{t \to t_1} \psi(t_1)(e)\)
- \((h, t) \models e' \triangleq \lim_{t \to t_1} \psi(t_1)(e)\)
- \((h, t) \models \text{true}\)
- \((h, t) \models \exp_1 = \exp_2 \iff (h, t) \models \exp_1 = (h, t) \models \exp_2\)
- \((h, t) \models evexp_1 = evexp_2 \iff (h, t) \models evexp_1 = (h, t) \models evexp_2\)
- \((h, t) \models \exp_1 < \exp_2 \iff (h, t) \models \exp_1 < (h, t) \models \exp_2\)
- \((h, t) \models \neg p \iff (h, t) \not\models p\)
- \((h, t) \models p_1 \lor p_2 \iff (h, t) \models p_1 \lor (h, t) \models p_2\)
- \((h, t) \models p_1 \triangleleft p_2 \iff \exists t_0 > t, (h, t_0) \models p_2 \text{ and for all } t_1 \in (t, t_0), (h, t_1) \models p_1\)
- \((h, t) \models p_1 \triangleleft p_2 \iff \exists t_0 < t, (h, t_0) \models p_2 \text{ and for all } t_1 \in (t_0, t), (h, t_1) \models p_1\)
- \((h, t) \models \exists x.p \iff (h_1, t) \models p, \text{ for some } h_1, \text{ a } x\text{-variant of } h\)
- \((h, t) \models \exists \epsilon.p \iff (h_1, t) \models p, \text{ for some } h_1, \text{ a } \epsilon\text{-variant of } h\)
- \((h, t) \models \exists n.p \iff (h_1, t) \models p, \text{ for some } h_1, \text{ a } n\text{-variant of } h\)
Definition 21 (Satisfiability, validity)
For a DTL formula \( p \) and a history \( h \in \mathcal{H} \), \( h \) satisfies \( p \) denoted \( h \models p \) iff \( (h, 0) \models p \).
A DTL formula \( p \) is satisfiable iff \( h \models p \) for some history \( h \in \mathcal{H} \).
A DTL formula \( p \) is valid, denoted \( \models p \), iff \( h \models p \) for all histories \( h \in \mathcal{H} \).

Later on a restricted version of DTL is considered in order to make it history stutter insensitive.

The following theorem states that the logic DTL is history process state stutter insensitive. Later on a restricted version of DTL is considered in order to make it history stutter insensitive.

Theorem 2 (DTL is history process state stutter insensitive)
Let \( rexp \) be a rigid expression, \( exp \) be an expression, \( evexp \) an event expression and \( p \) a temporal formula then

\[
\begin{align*}
& a \quad \forall t, h_0, h_1 : h_0 \simeq_{\delta_h} h_1 \rightarrow ((h_0, t) \models rexp = (h_1, di(h_1) \circ di^{-1}(h_0)(t)) \models rexp) \\
& b \quad \forall t, h_0, h_1 : h_0 \simeq_{\delta_h} h_1 \rightarrow ((h_0, t) \models exp = (h_1, di(h_1) \circ di^{-1}(h_0)(t)) \models exp) \\
& c \quad \forall t, h_0, h_1 : h_0 \simeq_{\delta_h} h_1 \rightarrow ((h_0, t) \models evexp = (h_1, di(h_1) \circ di^{-1}(h_0)(t)) \models evexp) \\
& d \quad \forall t, h_0, h_1 : h_0 \simeq_{\delta_h} h_1 \rightarrow ((h_0, t) \models p \iff (h_1, di(h_1) \circ di^{-1}(h_0)(t)) \models p)
\end{align*}
\]

The following definitions introduce substitution.

Definition 22 (Non-rigid process variable substitution in expressions)
Define substitution of \( w \in \mathbb{M} \cup \mathbb{X} \) by state expression \( exp \) in expression \( exp_0 \) denoted \( exp_0[exp/w] \) as follows, using \( \equiv \) for syntactic equality:

- \( rexp[exp/w] \equiv rexp \)
- \( v[exp/w] \equiv \begin{cases} exp & \text{if } v \equiv w \\ v & \text{if } v \not\equiv w \end{cases} \)
- \( v'[exp/w] \equiv \begin{cases} exp' & \text{if } v \equiv w \\ v' & \text{if } v \not\equiv w \end{cases} \)
- \( \langle v \rangle[exp/w] \equiv \begin{cases} \langle exp \rangle & \text{if } v \equiv w \\ \langle v \rangle & \text{if } v \not\equiv w \end{cases} \)
- \( x[exp/w] \equiv \begin{cases} exp & \text{if } x \equiv w \\ x & \text{if } x \not\equiv w \end{cases} \)
- \( x'[exp/w] \equiv \begin{cases} exp' & \text{if } x \equiv w \\ x' & \text{if } x \not\equiv w \end{cases} \)
2.2 Specification of Reactive Systems

- \( \cdot x[exp/w] \equiv \begin{cases} 
  \cdot exp & \text{if } x \equiv w \\
  \cdot x & \text{if } x \not\equiv w 
\end{cases} \)

- \( (exp_1 + exp_2)[exp/w] \equiv exp_1[exp/w] + exp_2[exp/w] \)

- ...

**Definition 23 (Rigid variable substitution in expressions)**

Define substitution of \( n \in \mathbb{N} \) by state rigid expression \( rexp \) in expression \( exp_0 \) denoted \( exp_0[rexp/n] \) as follows, using \( \equiv \) for syntactic equality:

- \( \mu[rexp/n] \equiv \mu \)

- \( n_0[rexp/n] \equiv \begin{cases} 
  rexp & \text{if } n \equiv n_0 \\
  n_0 & \text{if } n \not\equiv n_0 
\end{cases} \)

- \( n'_0[rexp/n] \equiv \begin{cases} 
  rexp' & \text{if } n \equiv n_0 \\
  n'_0 & \text{if } n \not\equiv n_0 
\end{cases} \)

- \( \cdot n_0[rexp/n] \equiv \begin{cases} 
  \cdot rexp & \text{if } n \equiv n_0 \\
  \cdot n_0 & \text{if } n \not\equiv n_0 
\end{cases} \)

- \( w[rexp/n] \equiv w, w \in \mathbb{V} \cup \mathbb{X} \)

- \( w'[rexp/n] \equiv w', w \in \mathbb{V} \cup \mathbb{X} \)

- \( \cdot w[rexp/n] \equiv \cdot w, w \in \mathbb{V} \cup \mathbb{X} \)

- \( (exp_1 + exp_2)[rexp/n] \equiv exp_1[rexp/n] + exp_2[rexp/n] \)

- ...

**Definition 24 (Event variable substitution in event expressions)**

Define substitution of \( \epsilon \in \mathbb{E} \) by state event expression \( evexp \) in \( evexp_0 \) denoted \( evexp_0[evexp/\epsilon] \) as follows, using \( \equiv \) for syntactic equality:

- \( \lambda[evexp/\epsilon] \equiv \lambda \)

- \( \text{a?}[evexp/\epsilon] \equiv \text{a?} \)

- \( \text{a!}[evexp/\epsilon] \equiv \text{a!} \)

- \( i[evexp/\epsilon] \equiv i \)

- \( e[evexp/\epsilon] \equiv e \)

- \( \epsilon_0[evexp/\epsilon] \equiv \begin{cases} 
  evexp & \text{if } \epsilon_0 \equiv \epsilon \\
  \epsilon_0 & \text{if } \epsilon_0 \not\equiv \epsilon 
\end{cases} \)

- \( \epsilon'_0[evexp/\epsilon] \equiv \begin{cases} 
  evexp' & \text{if } \epsilon_0 \equiv \epsilon \\
  \epsilon'_0 & \text{if } \epsilon_0 \not\equiv \epsilon 
\end{cases} \)
• $\{v_{exp}/e\} \equiv \begin{cases} \{v_{exp}\} & \text{if } v_0 \equiv e \\ \{v_0\} & \text{if } v_0 \not\equiv e \end{cases}$

Definition 25 (Process and event variable substitution in temporal formulae)
Define substitution for a non-rigid process variable $w \in \mathcal{W} \cup \mathcal{X}$ by state expression $exp$ in a temporal formula $p$, denoted $p[exp/w]$, as follows:

• $true[exp/w] \equiv true$
• $(exp_1 = exp_2)[exp/w] \equiv exp_1[exp/w] = exp_2[exp/w]$
• $(evexp_1 = evexp_2)[exp/w] \equiv (evexp_1 = evexp_2)$
• $(exp_1 < exp_2)[exp/w] \equiv exp_1[exp/w] < exp_2[exp/w]$
• $(\neg p)[exp/w] \equiv \neg(p[exp/w])$
• $(p_1 \lor p_2)[exp/w] \equiv p_1[exp/w] \lor p_2[exp/w]$
• $(p_1 \tilde{U} p_2)[exp/w] \equiv (p_1[exp/w]) \tilde{U} (p_2[exp/w])$
• $(p_1 \tilde{S} p_2)[exp/w] \equiv (p_1[exp/w]) \tilde{S} (p_2[exp/w])$
• $(\exists x.p)[exp/w] \equiv \exists x.(p[exp/w])$ if $x \not\in \text{var}(exp) \cup \{w\}$.
• $(\exists e.p)[exp/w] \equiv \exists e.(p[exp/w])$,
• $(\exists n.p)[exp/w] \equiv \exists n.(p[exp/w])$

Define substitution of rigid process variable $n \in \mathcal{R}$ by state rigid expression $rexp$ in temporal formula $p$ denoted $p[rexp/n]$ as follows:

• $true[rexp/n] \equiv true$
• $(exp_1 = exp_2)[rexp/n] \equiv exp_1[rexp/n] = exp_2[rexp/n]$
• $(evexp_1 = evexp_2)[rexp/n] \equiv (evexp_1 = evexp_2)$
• $(exp_1 < exp_2)[rexp/n] \equiv exp_1[rexp/n] < exp_2[rexp/n]$
• $(\neg p[rexp/n]) \equiv \neg(p[rexp/n])$
• $(p_1 \lor p_2)[rexp/n] \equiv p_1[rexp/n] \lor p_2[rexp/n]$
• $(p_1 \tilde{U} p_2)[rexp/n] \equiv (p_1[rexp/n]) \tilde{U} (p_2[rexp/n])$
• $(p_1 \tilde{S} p_2)[rexp/n] \equiv (p_1[rexp/n]) \tilde{S} (p_2[rexp/n])$
• $(\exists x.p)[rexp/n] \equiv \exists x.(p[rexp/n])$,
• $(\exists e.p)[rexp/n] \equiv \exists e.(p[rexp/n])$,
• $(\exists n_0.p)[rexp/n] \equiv \exists n_0.(p[rexp/n])$, if $n_0 \not\in \text{var}(rexp) \cup \{n\}$.
2.2 Specification of Reactive Systems

Define substitution of event variable \( e \in \mathcal{E} \) by state event expression \( \text{evexp} \) in temporal formula \( p \) denoted \( p[\text{evexp}/e] \) as follows:

- \( \text{true}[\text{evexp}/e] \equiv \text{true} \)
- \( (\text{exp}_1 = \text{exp}_2)[\text{evexp}/e] \equiv (\text{exp}_1 = \text{exp}_2) \)
- \( (\text{evexp}_1 = \text{evexp}_2)[\text{evexp}/e] \equiv (\text{evexp}_1[\text{evexp}/e] = \text{evexp}_2[\text{evexp}/e]) \)
- \( (\text{exp}_1 < \text{exp}_2)[\text{evexp}/e] \equiv (\text{exp}_1 < \text{exp}_2) \)
- \( (-p)[\text{evexp}/e] \equiv -(p[\text{evexp}/e]) \)
- \( (p_1 \lor p_2)[\text{evexp}/e] \equiv (p_1[\text{evexp}/e]) \lor (p_2[\text{evexp}/e]) \)
- \( (p_1 \bar{\Delta} p_2)[\text{evexp}/e] \equiv (p_1[\text{evexp}/e]) \bar{\Delta} (p_2[\text{evexp}/e]) \)
- \( (p_1 \tilde{\Delta} p_2)[\text{evexp}/e] \equiv (p_1[\text{evexp}/e]) \tilde{\Delta} (p_2[\text{evexp}/e]) \)
- \( (\exists x.p)[\text{evexp}/e] \equiv \exists x.(p[\text{evexp}/e]) \)
- \( (\exists_0.p)[\text{evexp}/e] \equiv \exists_0.(p[\text{evexp}/e]) \) where \( \epsilon_0 \not\in \text{ev}(\text{evexp}) \cup \{e\} \).

The following introduces the history variant of a history.

**Definition 26 (History variant)**

The history variant of a history with respect to non-rigid process variable \( w \in \mathcal{W} \cup \mathcal{X} \), and a state expression \( \text{exp} \), denoted by \( (h : w \rightsquigarrow \text{exp}) \), is defined for \( w_1 \in \mathcal{W} \cup \mathcal{X} \) as follows: Let \( \mu \in \text{Val} \) and \( \sigma \in \Sigma \) then

\[
(s : w \mapsto \mu)(w_1) \triangleq \begin{cases} 
\mu(w_1) & \text{if } w_1 \equiv w \\
\sigma(w_1) & \text{if } w_1 \not\equiv w
\end{cases}
\]

then

\[(h : w \rightsquigarrow \text{exp})(t_0) \triangleq \langle \psi_h(t_0), (\theta_h(t_0) : w \mapsto (h, t_0) = \text{exp}) \rangle \]

The history variant of a history with respect to rigid process variable \( n \in \mathcal{R} \), and a state rigid expression \( \text{rexp} \), denoted by \( (h : n \rightsquigarrow \text{rexp}) \), is defined for \( n_1 \in \mathcal{R} \) as follows: Let \( \mu \in \text{Val} \) and \( \sigma \in \Sigma \) then

\[
(s : n \mapsto \mu)(n_1) \triangleq \begin{cases} 
\mu(n_1) & \text{if } n_1 \equiv n \\
\sigma(n_1) & \text{if } n_1 \not\equiv n
\end{cases}
\]

then

\[(h : n \rightsquigarrow \text{rexp})(t_0) \triangleq \langle \psi_h(t_0), (\theta_h(t_0) : n \mapsto (h, t_0) = \text{rexp}) \rangle \]

The history variant of a history with respect to event variable \( e \in \mathcal{E} \), and a state event expression \( \text{evexp} \), denoted by \( (h : e \rightsquigarrow \text{evexp}) \), is defined for \( e_1 \in \mathcal{E} \) as follows: Let \( a \in \mathcal{A} \) and \( \delta \in \Delta \) then

\[
(s : e \mapsto a)(e_1) \triangleq \begin{cases} 
a & \text{if } e_1 \equiv e \\
\delta(e_1) & \text{if } e_1 \not\equiv e
\end{cases}
\]

23
A Dense Model Formalism

then

\[(h : \epsilon \leadsto \text{ev exp})(t_0) \triangleq \langle (\psi_h(t_0) : \epsilon \mapsto (h, t_0) \models \text{ev exp}), \theta_h(t_0) \rangle \]

The following substitution lemma holds.

**Lemma 2 (Substitution lemma)**

Let \( \text{exp}_0 \) be an expression, \( \text{exp} \) be a state expression, \( \nu \in \mathfrak{X} \cup \mathfrak{I} \), \( \text{r exp} \) be a state rigid expression, \( \eta \in \mathfrak{N} \), \( \text{ev exp}_0 \) an event expression, \( \text{ev exp} \) a state event expression, \( \epsilon \in \mathfrak{E} \), and \( \pi \) a temporal formula. Then the following holds:

\[
\begin{align*}
    a \quad (h, t) \models \text{exp}_0[\text{exp}/\nu] & \models \text{exp}_0 \\
    b \quad (h, t) \models \text{exp}_0[\text{r exp}/\eta] & \models \text{exp}_0 \\
    c \quad (h, t) \models \text{ev exp}_0[\text{ev exp}/\epsilon] & \models \text{ev exp}_0 \\
    d \quad (h, t) \models \pi[\text{exp}/\nu] & \text{iff} \ ( (h : \omega \leadsto \text{exp}), t ) \models \pi \\
    e \quad (h, t) \models \pi[\text{r exp}/\eta] & \text{iff} \ ( (h : \nu \leadsto \text{r exp}), t ) \models \pi \\
    f \quad (h, t) \models \pi[\text{ev exp}/\epsilon] & \text{iff} \ ( (h : \epsilon \leadsto \text{ev exp}), t ) \models \pi \\
\end{align*}
\]

The following proof system for DTL is inspired on [Bur82, Bur84, BKP86, MP89]. An erroneous variant of it appeared in [BKP86] where these authors state that it is “an almost verbatim copy of [Bur84]” indeed “almost” their axiom F5 was not copied well. Furthermore a link with the proof system of [KMP93] is established via axioms AX7b-AX7f, i.e., these axioms are needed for deriving their proof system. Note: because the models of [Bur82, Bur84] need not to satisfy the finite variability condition, and the persistency condition (once in an interval “going back or forward” doesn’t bring you outside that interval, and the induction axiom. This is the crucial difference between the model of [KMP93] and ours and the one in [Bur82, Bur84]. The difference between the model of [KMP93] and our model is that we have additional compositionality information as reflected in axioms AX0, AX5 and AX6.

The proof system is for the pure logic, i.e., it is not meant for a specific reactive system. Axioms AX0-AX9 characterize our notion of histories; they should follow from the definition of history (Def. 9), and, because a history is a pair consisting of a event and a state function, also from Definition 7 and 8. Ax10 and Ax11 are the axioms for substitution and quantification. Axioms F1-F7 are the axioms of the future part of DTL and P1-P7 the past part. As rules we take standard ones, i.e., the modus ponens, generalization, specialization, instantiation and universal generalization.

**Definition 27 (Proof system for DTL)**

Let \( \eta \in \mathfrak{N} \), \( \nu \in \mathfrak{N} \), \( \nu \in \mathfrak{N} \cup \mathfrak{X} \), \( \mathfrak{X} \in \mathfrak{X} \) and \( \epsilon \in \mathfrak{E} \).

**Axioms** All the axioms for state formulae.

\[
\text{AX0:} \quad (\epsilon = a) \lor \epsilon = \epsilon' \lor \epsilon = i \lor \epsilon = e) \Rightarrow (\epsilon' = \lambda \land \epsilon = \epsilon)
\]

Non-\( \lambda \) actions are points surrounded by \( \lambda \) actions conform Definition 7.

\[
\text{AX1:} \quad \text{first} \rightarrow \epsilon = \lambda \land \nu' = \nu \land x' = x
\]
2.2 Specification of Reactive Systems

The initially stuttering requirement conform Definition 7 and 8.

AX2 : □(x = x ∧ v = v)

The process variables are left continuous variables conform Definition 8.

AX3 : O(x' = x ∧ v' = v) ∧ (((x' = x) ∨ v' = v) ⇒ O(x'' = x' ∧ v'' = v'))

The value of process variables are maintained during an interval, conform Definition 8.

AX4 : □(n = n' ∧ n = n)

The rigid variables don’t change at all conform Definition 8.

AX5 : (c = a? ∨ c = a!) ⇒ v' = v

Communication actions don’t change the shared variables conform Definition 9.

AX6 : c = λ ⇒ (v' = v ∧ x' = x)

A λ action causes no state change.

AX7a : ¹p ⇒ ¹ ¹p
AX7b : ¬O p ⇒ O¬p
AX7c : ¬O p ⇒ O¬p
AX7d : O O p ⇒ O p
AX7e : O O p ⇒ O p
AX7f : (p ∧ p ⇒ O p ∧ O p ⇒ p) ⇒ □ p

The underlying structure is dense (a), and satisfies the finite variability condition (b & c), and is persistent (d & e). Axiom (f) is the induction axiom. For an explanation of d–f see [KMP93].

AX8 : □ ¹true

There is no last element, i.e., the future is unbounded.

AX9 : □ ¹ ¹false

There exists a first element.

AX10 : (exp1 = exp2) ⇒ (p[exp1/w] ⇔ p[exp2/w])
      ∧ (rexp1 = rexp2) ⇒ (p[rexp1/n] ⇔ p[rexp2/n])
      ∧ (evexp1 = evexp2) ⇒ (p[evexp1/e] ⇔ p[evexp2/e])
where $p$ is a state formula and none of the variables appearing in respectively $\text{exp}_1$, $\text{exp}_2$, $\text{rexp}_1$, $\text{rexp}_2$, $\text{evexp}_1$ and $\text{evexp}_2$ is quantified in $p$.
Replacement of equal expressions.

$$AX11 : \quad (\forall x.p) \Rightarrow p[\text{exp}/x]$$
$$\wedge \quad (\forall n.p) \Rightarrow p[\text{rexp}/n]$$
$$\wedge \quad (\forall e.p) \Rightarrow p[\text{evexp}/e]$$

where none of the variables appearing in $\text{exp}$, $\text{rexp}$ and $\text{evexp}$ is quantified in $p$.

Quantifier instantiation.

$$F1: \quad \Box(p \rightarrow q) \Rightarrow (r \mathcal{U} p \rightarrow r \mathcal{U} q)$$

$\mathcal{U}$ is monotonic in its second argument.

$$F2: \quad \Box(p \rightarrow q) \Rightarrow (p \mathcal{U} r \rightarrow q \mathcal{U} r)$$

$\mathcal{U}$ is monotonic in its first argument.

$$F3: \quad (p \land r \mathcal{U} q) \Rightarrow (r \mathcal{U} (q \land r \mathcal{S} p))$$

The relation of reflection holding between past and future.

$$F4: \quad (q \mathcal{U} p \land \neg (r \mathcal{U} p)) \Rightarrow q \mathcal{U} (q \land \neg r)$$

$$F5: \quad q \mathcal{U} p \Rightarrow (q \land q \mathcal{U} p) \mathcal{U} p$$

$$F6: \quad q \mathcal{U} (q \land q \mathcal{U} p) \Rightarrow q \mathcal{U} p$$

$$F7: \quad (q \mathcal{U} p \land s \mathcal{U} r) \Rightarrow (q \land s) \mathcal{U} (p \land r) \lor (q \land s) \mathcal{U} (p \land s) \lor (q \land s) \mathcal{U} (q \land r)$$

The underlying structure is linear.

$$P1: \quad \Box(p \rightarrow q) \Rightarrow (r \mathcal{S} p \rightarrow r \mathcal{S} q)$$

$\mathcal{S}$ is monotonic in its second argument.

$$P2: \quad \Box(p \rightarrow q) \Rightarrow (p \mathcal{S} r \rightarrow q \mathcal{S} r)$$

$\mathcal{S}$ is monotonic in its first argument.

$$P3: \quad (p \land r \mathcal{S} q) \Rightarrow (r \mathcal{S} (q \land r \mathcal{U} p))$$
### 2.2 Specification of Reactive Systems

The relation of reflection holding between past and future.

\[ P4 : (q \dot{S} p \land \neg (r \dot{S} p)) \Rightarrow q \dot{S} (q \land \neg r) \]

\[ P5 : \quad q \dot{S} p \Rightarrow (q \land q \dot{S} p) \dot{S} p \]

\[ P6 : \quad q \dot{S} (q \land q \dot{S} p) \Rightarrow q \dot{S} p \]

\[ P7 : (q \dot{S} p \land s \dot{S} r) \Rightarrow (q \land s) \dot{S} (p \land r) \lor (q \land s) \dot{S} (p \land s) \lor (q \land s) \dot{S} (q \land r) \]

The underlying structure is linear.

**Rules**

\[
\begin{align*}
    p, \quad p \rightarrow q & \quad \Rightarrow q
\end{align*}
\]

*The Modus Ponens.*

\[
\begin{align*}
    p & \quad \text{for state formula } p \text{ in which all occurrences of} \\
    \square p & \quad \text{parameterized sentence symbols in } p \text{ are rigid}
\end{align*}
\]

**Generalization.**

\[
\begin{align*}
    \quad \square p & \quad \text{for state formula } p \\
    p & \quad \text{where } p \text{ doesn’t contain variables which are bound in } p
\end{align*}
\]

**Specialization.**

\[
\begin{align*}
    p & \quad \text{where } p \text{ doesn’t contain variables which are bound in } p \\
    p[p_1/p_0] & \quad \text{for } x \text{ not free in } p
\end{align*}
\]

**Instantiation.**

\[
\begin{align*}
    p_0 \Rightarrow p_1 & \quad \text{for } x \text{ not free in } p_0 \\
    \forall x.p_1 & \quad \text{where } x \text{ doesn’t contain variables which are bound in } p_0
\end{align*}
\]

\[
\begin{align*}
    p_0 \Rightarrow p_1 & \quad \text{for } n \text{ not free in } p_0 \\
    \forall n.p_1 & \quad \text{where } n \text{ doesn’t contain variables which are bound in } p_0
\end{align*}
\]

\[
\begin{align*}
    p_0 \Rightarrow p_1 & \quad \text{for } \epsilon \text{ not free in } p_0 \\
    \forall \epsilon.p_1 & \quad \text{where } \epsilon \text{ doesn’t contain variables which are bound in } p_0
\end{align*}
\]

*Universal Generalization.*

27
The following definition characterizes a machine $M$ in DTL. This kind of DTL formulae is history stutter insensitive.

**Definition 28 (Machine in DTL)**

Given basis $B = (\langle \text{In}, \text{Out} \rangle, (V, X))$. Let $\text{In}$ be defined as $\{a? | a \in \text{In}\}$ and let $\text{Out}$ be defined as $\{a! | a \in \text{Out}\}$. Let $I$ be a DTL formula over $V \cup X$ without the $\mathcal{S}$, $\mathcal{U}$ and $\exists$ operators. Let $T$ be a finite set of DTL formulae $\tau$ of the form $(\text{event}_r \land \text{trans}_r)$ where $\text{event}_r$ is of the form $\epsilon = a_r$ where $a_r \in \{i, e\} \cup \text{In} \cup \text{Out}$, and $\text{trans}_r$ a DTL formula over $V \cup X$ and $V' \cup X'$ (variables primed with $'$) without the $\mathcal{S}$, $\mathcal{U}$ and $\exists$ operators such that $(\epsilon = e \Rightarrow \bigwedge_{x \in X} x' = x)$, i.e., an environment action doesn’t change the local variables of the system. Define the stutter step, denoted by $\text{stut}$, as $\epsilon = \lambda \lor (\epsilon = i \land (V, X') = (V, X)) \lor (\epsilon = e \land (V, X') = (V, X))$. Let $T$ be the DTL formula $\text{stut} \lor \bigvee_{\tau \in T} \tau$. A machine in DTL is defined as $(B, I \land \Box T)$.

**Lemma 3**

Given a machine in DTL $(B, I \land \Box T)$ then there exists a semantic machine $M = (B, I, T)$ such that $\text{Comp}(M) = \text{Hist}(I \land \Box T)$.

The following example is an illustration of a machine in DTL.

**Example 3**

Machine $M$ in example 1 as DTL-formula:

1. **Basis**: $B = (\langle \text{In}, \text{Out} \rangle, (V, X))$ where
   \[
   \begin{align*}
   \text{In} & \overset{=} {=} \{a\}, \\
   \text{Out} & \overset{=} {=} \emptyset, \\
   V & \overset{=} {=} \{v\}, \\
   X & \overset{=} {=} \{u\}
   \end{align*}
   \]

2. **Initial States**: $I \overset{=} {=} (v, u) = (0, 0)$

3. **Transitions**: $T \overset{=} {=} \left(\begin{array}{l}
   \left(\epsilon = a? \land u = 0 \land (v, u)' = (v, 1)\right) \lor \\
   \left(\epsilon = i \land (v, u) = (1, 1) \land (v, u)' = (0, 2)\right) \lor \\
   \left(\epsilon = e \land (v, u)' = (v + 1, u)\right) \lor \\
   \text{stut}
   \end{array}\right)
   \]

The machine specification of a system in DTL is as follows.

**Definition 29 (Machine specification of a system in DTL)**

Given a machine $(B, I \land \Box T)$ in DTL. Let $\text{WF} \subseteq T$ be the set of weak fair transitions and $\text{SF} \subseteq T$ be the set of strong fair transitions. For $\tau \in T$ define the enabledness condition for $\tau$ denoted $\text{En}(\tau)$ as $\exists \bar{v}_0.\tau[\bar{v}_0/\bar{v}]$ where $\tau[\bar{v}_0/\bar{v}]$ denotes the substitution of $\bar{v}_0$ (a list of variables not in $V \cup X$) for $\bar{v}'$ (the list of primed variables in $\tau$). Let $L$ be the DTL formula $\Lambda_{\text{wef}}(\Box En(\tau) \rightarrow \Box \Box \tau) \land \Lambda_{\text{esf}}(\Box \Box En(\tau) \rightarrow \Box \Box \tau)$. The **machine specification of a system in DTL** is then a tuple $(B, I \land \Box T \land L)$.
2.3 Refinement and Composition of Reactive System Specifications

Note: in above definition $L$ is such that $cl(Hist(I \wedge \Box T) \cap Hist(L)) = Hist(I \wedge \Box T)$, i.e., it satisfies the machine closedness property. With this the following lemma is straightforward.

**Lemma 4**

Given DTL machine specification $(B, I \wedge \Box T \wedge L)$ of a system, there exists a semantic machine specification $S = (B, \text{Comp}(M) \cap L)$ such that $\text{Comp}(M) \cap L = Hist(I \wedge \Box T \wedge L)$.

2.3 Refinement and Composition of Reactive System Specifications

In this section the notion of refinement and composition of reactive systems is introduced. Intuitively refinement means that the set of histories of a concrete system is a subset of the set of histories of an abstract system. Composition means that the histories of the component systems are “merged” into composite histories, i.e., the histories of the composed system. Our merge operator is based on the merge operator of Aczel [Acz83]. Both are first defined at the semantic level and then for the DTL specifications.

2.3.1 Semantic Refinement and Composition of Specifications

In this section refinement and composition of reactive systems is defined at the semantical level. Refinement means that the set of histories of a concrete system is a subset of the set of histories of an abstract system. Because histories also contain local information the subset relation doesn’t correspond directly with refinement. The local information should first be projected away. The following definition captures this projection of local information.

**Definition 30 (Observable system specification)**

Given system specification $S = (B, H)$ where $B = ((\text{In}, \text{Out}), (V, X))$. The **observable system specification** is defined as $(\Sigma(B), \mathcal{O}_X(H))$ where $\Sigma(B)$ denotes the observable basis and is defined as $\Sigma(B) \triangleq ((\text{In}, \text{Out}), (V, \emptyset))$ and $\mathcal{O}_X(H)$ denotes the set of observable histories corresponding to $H$ and is defined as

$$\{h \in H | \exists h_1 \in H : h \text{ is an X-variant of } h_1\}$$

**Definition 31 (Refinement of systems)**

Given concrete system $S_c \triangleq (B_c, H_c)$ and abstract system $S_a \triangleq (B_a, H_a)$. $S_c$ refines $S_a$ denoted by $S_c \text{ ref } S_a$ iff $\Sigma(B_c) = \Sigma(B_a)$ and $\mathcal{O}_X(H_c) \subseteq \mathcal{O}_X(H_a)$.

A more general definition of refinement would be one wherein both the abstract and concrete system are composed of subsystems. Therefore the notion of composition is introduced. Intuitively the composition of two systems is that matching histories are merged into one history. A history of one system matches a history of the other system if for all time points $t$

1. the state information of the two histories at time $t$ are same and...
So if the two components each perform an action this prohibited because we want to model interleaving where only communication actions can possible occur simultaneously. Two matching histories are then merged into one history by (1) “copying” the state-information of the two histories; and in case (2a) the resulting event becomes \( \lambda \), and in case (2b) the resulting event becomes \( e \), and in case (2c) the resulting event becomes \( i \), and in case (2d) the resulting event becomes \( i \), and in case (2e) the resulting event becomes the communication action.

**Definition 32 (Composition of two systems)**

Given systems \( S_i = (B_i, H_i) \) with \( B_i = ((\text{In}_i, \text{Out}_i), (V_i, X_i)) \) \((i = 1, 2)\) such that \( \text{In}_1 \cap \text{In}_2 = \emptyset \), \( \text{Out}_1 \cap \text{Out}_2 = \emptyset \) and \( X_1 \cap X_2 = \emptyset \). The composed system \( S = S_1 \| S_2 \) is defined as \((B, H) = ((\text{In}_1 \setminus \text{Out}_2 \cup \text{In}_2 \setminus \text{Out}_1, \text{Out}_1 \setminus \text{Out}_2 \cup \text{In}_1), (V_1 \cup V_2, X_1 \cup X_2))\) and \( H = H_1 \otimes H_2 \). The \( \otimes \) is the merge operator which merges the histories \( h_1 \in H_1 \) and \( h_2 \in H_2 \) into one history \( h \) and which is defined as follows:

\[
H_1 \otimes H_2 = \{ h \in H \mid \exists h_1 \in H_1, h_2 \in H_2, \otimes (h, h_1, h_2) \}
\]

where for \( h = \langle \psi, \theta \rangle \) and \( h_j = \langle \psi_j, \theta_j \rangle \) \((j = 1, 2)\),

\[
\otimes (h, h_1, h_2) \text{ iff} \quad \theta = \theta_1 \land \theta = \theta_2
\]

\[
\forall t : \\
\psi(t)(\epsilon) = \lambda \land \psi_1(t)(\epsilon) = \lambda \land \psi_2(t)(\epsilon) = \lambda \\
\psi(t)(\epsilon) = e \land \psi_1(t)(\epsilon) = e \land \psi_2(t)(\epsilon) = e \\
\psi(t)(\epsilon) = i \land \psi_1(t)(\epsilon) = i \land \psi_2(t)(\epsilon) = e \\
\psi(t)(\epsilon) = i \land \psi_1(t)(\epsilon) = e \land \psi_2(t)(\epsilon) = i \\
\exists a \in \text{In}_1 \cap \text{Out}_2 : \psi(t)(\epsilon) = i \land \psi_1(t)(\epsilon) = a? \land \psi_2(t)(\epsilon) = a! \\
\exists a \in \text{In}_2 \cap \text{Out}_1 : \psi(t)(\epsilon) = i \land \psi_1(t)(\epsilon) = a! \land \psi_2(t)(\epsilon) = a? \\
\exists a \in \text{In}_1 \setminus \text{Out}_2 : \psi(t)(\epsilon) = a? \land \psi_1(t)(\epsilon) = a? \land \psi_2(t)(\epsilon) = e \\
\exists a \in \text{Out}_1 \setminus \text{In}_2 : \psi(t)(\epsilon) = a! \land \psi_1(t)(\epsilon) = a! \land \psi_2(t)(\epsilon) = e \\
\exists a \in \text{Out}_2 \setminus \text{In}_1 : \psi(t)(\epsilon) = a? \land \psi_1(t)(\epsilon) = e \land \psi_2(t)(\epsilon) = a? \\
\exists a \in \text{Out}_2 \setminus \text{In}_1 : \psi(t)(\epsilon) = a! \land \psi_1(t)(\epsilon) = e \land \psi_2(t)(\epsilon) = a!
\]

The following Lemma expresses that the “making observable” operation and the merge operator are monotonic and that the “making observable” operation on the composed system is equal to the “making observable” operation on the components.
2.3 Refinement and Composition of DTL Specifications

Lemma 5 (Properties of $\mathcal{O}$ and $\otimes$)
Given systems $(B_1, H_0)$, $(B_1, H_1)$, $(B_2, H_2)$ and $(B_2, H_3)$ then

(a) $H_0 \subseteq H_1$ implies $H_0 \otimes H_2 \subseteq H_1 \otimes H_2$
(b) $\mathcal{O}_{B_2}(H_1 \otimes H_2) = \mathcal{O}_{B_1}(H_1) \otimes \mathcal{O}_{B_2}(H_2)$
(c) $H_0 \subseteq H_1$ implies $\mathcal{O}_{B_1}(H_0) \subseteq \mathcal{O}_{B_1}(H_1)$
(d) $(H_0 \cap H_1) \otimes (H_2 \cap H_3) \subseteq (H_0 \otimes H_2) \cap (H_1 \otimes H_3)$

The following theorem of compositional refinement can be inferred from the above lemma.

Theorem 3 (Compositional refinement)
Given concrete systems $S_i = (B_i, H_i)$ ($i = 1, 2$) and abstract systems $S_j = (B_j, H_j)$ ($j = 3, 4$) such that $\mathcal{O}(B_1) = \mathcal{O}(B_3)$ and $\mathcal{O}(B_2) = \mathcal{O}(B_4)$ then $S_1$ ref $S_3$ and $S_2$ ref $S_4$ implies $S_1 \parallel S_2$ ref $S_3 \parallel S_4$.

It is very common that a shared variable is only used by the subcomponents of a system and not by the environment of the system. This variable acts then as a local variable for the system. The following definition introduces encapsulation which makes certain shared variables local to the system.

Definition 33 (Encapsulation)
Given system $S = (B, H)$ where $B = ((\text{In}, \text{Out}), (V, X))$ then encapsulation of $V_1$ in $S$ with $V_1 \subseteq V$ is denoted by $S \upharpoonright V_1$ and defined by $(B_1, \text{Enc}_{V_1}(H))$ where $B_1 \triangleq ((\text{In}, \text{Out}), (V \setminus V_1, X \cup \text{ren}(V_1)))$ where ren is a mapping from the shared variables to the local variables and intuitively “renames” the shared variables of $V_1$ to fresh local variables (not already in $X$). The encapsulation operator $\text{Enc}_{V_1}(H)$ is defined as

$$\{ h \in H \mid h \in H \land \forall t : \psi(t)(e) = e \rightarrow \theta(t)[V_1] = \lim_{t \leftarrow t_1} \theta(t_1)[V_1]\}$$

As ren mapping in above definition we usually take the identity mapping (almost it transforms bold variables names to non-bold variables names) because those shared variables that we want to make local are not yet in the set of local variables. In the following when ren is not given this identity mapping should be assumed.

2.3.2 Refinement and Composition of DTL Specifications

In this section the refinement and composition notion of the previous section are translated into DTL by defining it for machine specifications (Def. 29). This means that first the observable machine specification should be defined in DTL.

Definition 34 (Observable machine specification in DTL)
Given machine specification $(B, I \land \Box T \land L)$ in DTL and then the corresponding observable machine specification is defined as $(\mathcal{O}(B), (\exists X. (I \land \Box T \land L)))$.

The following lemma expresses that existential quantification relates to the semantic notion of observable histories.
Lemma 6
Given DTL machine specification $S = (B, I \land \square T \land L)$ then $O_X(Hist(I \land \square T \land L)) = Hist((\exists X . (I \land \square T \land L)))$

Theorem 4 (Refinement of machine specifications)
Given concrete machine specification $S_c \triangleq (B_c, I_c \land \square T_c \land L_c)$ where $B_c \triangleq (B^A_c, (V_c, X_c))$ and abstract machine specification $S_a \triangleq (B_a, I_a \land \square T_a \land L_a)$ where $B_a \triangleq (B^A_a, (V_a, X_a))$. Then $S_c$ refines $S_a$ denoted $S_c \triangleright S_a$ iff

$$D(B_c) = D(B_a) \text{ and } (\exists X_a . (I_a \land \square T_a \land L_a)) \rightarrow (\exists X_a . (I_a \land \square T_a \land L_a))$$

Composition of DTL machine specifications can be defined in the same way as in the previous section.

Definition 35 (Composition of two DTL machine specifications)
Given DTL machine system specifications $S_i \triangleq (B_i, I_i \land \square T_i \land L_i)$ where $B_i \triangleq (B^A_i, B^P_i)$, for $i = 1, 2$. Let $B^A \odot B^A (\epsilon, \epsilon_1, \epsilon_2)$ be defined as

\[
\begin{align*}
&\forall \epsilon = \lambda \land \epsilon_1 = \lambda \land \epsilon_2 = \lambda \\
&\forall \epsilon = \epsilon_1 \land \epsilon = \epsilon_2 = \epsilon \\
&\forall \epsilon = \epsilon_2 \land \epsilon_1 = \epsilon_1 \land \epsilon_2 = \epsilon \\
&\forall \epsilon = \epsilon_2 \land \epsilon_1 = \epsilon \land \epsilon_2 = \epsilon \\
&\forall V_{a \in I_1 \cap O_{out}} \epsilon = \epsilon_1 = a? \land \epsilon_2 = a! \\
&\forall V_{a \in I_2 \cap O_{out}} \epsilon = \epsilon_1 = a! \land \epsilon_2 = a? \\
&\forall V_{a \in I_1 \cap O_{out}} \epsilon = a? \land \epsilon_1 = a? \land \epsilon_2 = \epsilon \\
&\forall V_{a \in I_2 \cap O_{out}} \epsilon = a! \land \epsilon_1 = a! \land \epsilon_2 = \epsilon \\
&\forall V_{a \in I_1 \cap O_{out}} \epsilon = a? \land \epsilon_1 = a! \land \epsilon_2 = a? \\
&\forall V_{a \in I_2 \cap O_{out}} \epsilon = a! \land \epsilon_1 = a! \land \epsilon_2 = a! \\
\end{align*}
\]

Then the composed machine system specification $S$ is defined as $(B, H)$ where

$$H \triangleq \exists \epsilon_1, \epsilon_2, B^A \odot B^A (\epsilon, \epsilon_1, \epsilon_2) \land (I_1 \land \square T_1 \land L_1)[\epsilon_1/\epsilon] \land (I_2 \land \square T_2 \land L_2)[\epsilon_2/\epsilon]$$

$$B \triangleq ((I_{out} \setminus O_{out} \cup O_{out}), (V_{out} \cup V_{out} \setminus O_{out}), (V_{out} \cup V_{out} \setminus O_{out} \setminus V_{out}), (V_{out} \cup V_{out} \setminus O_{out} \setminus V_{out} \setminus V_{out})).)$$

This definition can be easily extended for $n$ DTL specifications. One has then to define a predicate $\odot B^A (\epsilon, \tau)$ corresponding to the operation of merging $n$ components.

Theorem 5 (Semantic merge is almost conjunction)
Given machine system specifications $(B_i, I_i \land \square T_i \land L_i)$ where $B_i \triangleq ((I_i, O_{out}), (V_i, X_i))$, for $i = 1, 2$ and composed machine system specification as in definition 35, i.e., $(B, H)$ where $H \triangleq \exists \epsilon_1, \epsilon_2, B^A \odot B^A (\epsilon, \epsilon_1, \epsilon_2) \land (I_1 \land \square T_1 \land L_1)[\epsilon_1/\epsilon] \land (I_2 \land \square T_2 \land L_2)[\epsilon_2/\epsilon]$ and

$$B \triangleq ((I_{out} \setminus O_{out} \cup O_{out}), (V_{out} \cup V_{out} \setminus O_{out} \setminus V_{out} \setminus V_{out}), (V_{out} \cup V_{out} \setminus O_{out} \setminus V_{out} \setminus V_{out} \setminus V_{out}), (V_{out} \cup V_{out} \setminus O_{out} \setminus V_{out} \setminus V_{out} \setminus V_{out} \setminus V_{out})).)$$

then

$$Hist(I_1 \land \square T_1 \land L_1) \otimes Hist(I_2 \land \square T_2 \land L_2) = Hist(H)$$
Encapsulation of shared variables for DTL specifications is defined as follows.

**Definition 36 (Encapsulation)**
Given machine specification \( S \triangleq (B, H) \) then encapsulation of \( V_1 \) in \( S \) with \( V_1 \subseteq V \) denoted by \( S \downarrow V_1 \) is defined as \( (B_1, H \wedge (\epsilon = e \Rightarrow V'_1 = V_1)) \) where \( B_1 \triangleq (E, V \setminus V_1, X \cup V_1) \).

The following theorem states that above definition indeed captures encapsulation.

**Theorem 6**
Given machine specification \( S \triangleq (B, H) \) and given set of shared variables \( V_1 \subseteq V \) then

\[
Enc_{V_1}(Hist(H)) = Hist(H \wedge (\epsilon = e \Rightarrow V'_1 = V_1))
\]

**Example 4**
Abstract machine specification \( S_a \triangleq (B, I \wedge T \wedge L) \) is refined by the composition of concrete machine specifications \( S_{c_1} \triangleq (B_1, I_1 \wedge T_1 \wedge L_1) \) and \( S_{c_2} \triangleq (B_2, I_2 \wedge T_2 \wedge L_2) \).

The abstract machine specification \( S_a \) is defined as follows:

1. **Basis** \( B = ((\text{In}, \text{Out}), (\text{V}, \text{X})) \)

   \[
   \begin{align*}
   \text{In} & \triangleq \{b\}, \\
   \text{Out} & \triangleq \{a\}, \\
   \text{V} & \triangleq \{s\}, \\
   X & \triangleq \{x\}
   \end{align*}
   \]

2. **Initial States**

   \( I \triangleq (s, x) = (0, 0) \)

3. **Transitions**

   \( T \triangleq \)

   \[
   \begin{align*}
   &\forall \left( \epsilon = a! \wedge x = 0 \wedge (s, x)' = (s, 1) \right) \\
   &\forall \left( \epsilon = b? \wedge x = 1 \wedge (s, x)' = (s, 2) \right) \\
   &\forall \left( \epsilon = i \wedge (s, x) = (1, 2) \wedge (s, x)' = (0, x) \right) \\
   &\forall \left( \epsilon = e \wedge (s, x)' = (1, x) \right) \\
   &\forall \text{stut}_a
   \end{align*}
   \]

   These transitions are illustrated in figure 2.4. Note: the stutter transitions are not drawn in all subsequent figures in order to minimize the number of edges.

4. **Liveness**

   \( L \triangleq \text{true} \)
The definition of $S_{c_1}$ is as follows:

1. **Basis** $B_1 = ((\text{In}_1, \text{Out}_1), (\text{V}_1, \text{X}_1))$

   \begin{align*}
   \text{In}_1 & \triangleq \{c\}, \\
   \text{Out}_1 & \triangleq \{a\}, \\
   \text{V}_1 & \triangleq \{s\}, \\
   \text{X}_1 & \triangleq \{t\}
   \end{align*}

2. **Initial States**

   $$I_1 \triangleq (s,t) = (0,0)$$

3. **Transitions**

   $$T_1 \triangleq$$

   $$\forall \left( \epsilon = a! \land t = 0 \land (s,t)' = (s,1) \right)$$

   $$\forall \left( \epsilon = c? \land t = 1 \land (s,t)' = (s,2) \right)$$

   $$\forall \left( \epsilon = e \land (s,t)' = (1,t) \right)$$

   $$\forall \left( \epsilon = e \land (s,t)' = (0,t) \right)$$

   $$\forall \text{stut}_1$$

   These transitions are illustrated in figure 2.5

4. **Liveness**

   $$L_1 \triangleq \text{true}$$

The definition of $S_{c_2}$ is as follows:
2.3 Refinement and Composition of Reactive System Specifications

![Concrete Machine Diagram]

Figure 2.5: Concrete machine 1

1. **Basis** $B_2 = ((\text{In}_2, \text{Out}_2), (V_2, X_2))$

   \[
   \begin{align*}
   \text{In}_2 & \trianglerighteq \{b\}, \\
   \text{Out}_2 & \trianglerighteq \{c\}, \\
   V_2 & \trianglerighteq \{s\}, \\
   X_2 & \trianglerighteq \{u\}
   \end{align*}
   \]

2. **Initial States**

   \[
   I_2 \trianglerighteq (s, u) = (0, 0)
   \]

3. **Transitions**

   \[
   T_2 \trianglerighteq \\
   \forall \left( \epsilon = c! \land u = 0 \land (s, u)' = (s, 1) \right) \\
   \forall \left( \epsilon = b? \land u = 1 \land (s, u)' = (s, 2) \right) \\
   \forall \left( \epsilon = i \land (s, u) = (1, 2) \land (s, u)' = (0, u) \right) \\
   \forall \left( \epsilon = e \land (s, u)' = (1, u) \right) \\
   \forall \text{stut}_2
   \]

   These transitions are illustrated in figure 2.6

4. **Liveness**

   \[
   L_2 \trianglerighteq \text{true}
   \]

According to definition 35 the composition of $S_{c_1}$ and $S_{c_2}$ is as follows:

\[
( ( (\text{In}_1 \setminus \text{Out}_2 \cup \text{In}_2 \setminus \text{Out}_1, \text{Out}_1 \setminus \text{Out}_2 \cup \text{In}_1), (V_1 \cup V_2, X_1 \cup X_2)) , \\
\exists \epsilon_1, \epsilon_2. (B_1^\dagger \circ B_2^\dagger (\epsilon, \epsilon_1, \epsilon_2) \land (I_1 \land \Box T_1 \land L_1)[\epsilon_1/\epsilon] \land (I_2 \land \Box T_2 \land L_2)[\epsilon_2/\epsilon])
\]
Let $H_c \triangleq \exists \epsilon_1, \epsilon_2. (B_1^c \cap B_2^c (\epsilon_1, \epsilon_2) \land (I_1 \land \Box T_1 \land L_1)) [\epsilon_1/\epsilon] \land (I_2 \land \Box T_2 \land L_2) [\epsilon_2/\epsilon]$ and $H_a \triangleq I \land \Box T \land L$, then the composition of $S_{c_1}$ and $S_{c_2}$ refines $S_a$ iff

1. $\mathcal{O}(B_c) = \mathcal{O}(B)$
2. $\models (\exists t, u. (H_c)) \rightarrow (\exists x. (H_a))$

The following section will show that both conditions hold. Hence we have refinement.
2.4 Proving Refinement of Reactive System Specifications

This section explains how refinement of reactive systems can be proved. The standard technique of Abadi & Lamport [AL91] is used, i.e., refinement is proven by providing a refinement mapping from the concrete system to the abstract system. Firstly we give its definition at the semantic level and then for DTL specifications.

2.4.1 Proving Semantic Refinement of Specifications

Refinement of reactive systems is proved by means of a refinement mapping from the concrete system to the abstract system. A refinement mapping maps a history at the concrete level to a history at the abstract level, more specifically, it maps the states appearing in the concrete history to states appearing in the abstract history.

**Definition 37 (Refinement mapping between systems)**

Given concrete system $S_c \triangleq (B_c, H_c)$ and abstract system $S_a \triangleq (B_a, H_a)$ s.t. $\mathcal{O}(B_c) = \mathcal{O}(B_a)$. A refinement mapping from $S_c$ to $S_a$ is a mapping $f$ from states appearing in histories of $H_c$ to states appearing in histories of $H_a$, i.e., $f$ is mapping from with $f : \Sigma \rightarrow \Sigma$ s.t.

- The values of observable variables are not changed, i.e., for all $\sigma \in \Sigma$: $\sigma|^{|_{V_c} = f(\sigma)|^{|_{V_a}}.
- For all $h_c \in H_c$ there exists a $h_a \in H_a$ s.t. for all $t \in \mathbb{R}^{\geq 0}$, $\psi_c(t) = \psi_a(f(t))$.

**Lemma 7**

Given concrete system $S_c \triangleq (B_c, H_c)$ and abstract system $S_a \triangleq (B_a, H_a)$ s.t. $\mathcal{O}(B_c) = \mathcal{O}(B_a)$. If there exists a refinement mapping from $S_c$ to $S_a$, then $S_c \text{ ref } S_a$.

The concept of refinement mappings can also be applied to machine specifications. A machine specification is of the form $(B, \text{Comp}(M) \cap L)$. Refinement means then that $f(\text{Comp}(M_c) \cap L_c) \subseteq \text{Comp}(M_a) \cap L_a$ for refinement mapping $f$. This can be split into (1) $f(\text{Comp}(M_c) \cap L_c) \subseteq \text{Comp}(M_a)$ and (2) $f(\text{Comp}(M_c) \cap L_c) \subseteq L_a$. From $f(\text{Comp}(M_c)) \subseteq \text{Comp}(M_a)$ follows (1) because $f(\text{Comp}(M_c) \cap L_c) \subseteq f(\text{Comp}(M_a))$. So the verification condition can be split into a condition on machines and a condition involving machines together with supplementary conditions. This leads to the following definition.

**Definition 38 (Refinement mapping between machine specifications)**

Given concrete machine specification $S_c \triangleq (B_c, \text{Comp}(M_c) \cap L_c)$, where $M_c \triangleq (B_c, I_c, T_c)$, and abstract machine specification $S_a \triangleq (B_a, \text{Comp}(M_a) \cap L_a)$, where $M_a \triangleq (B_a, I_a, T_a)$. A refinement mapping from machine specification $S_c$ to machine specification $S_a$ is a mapping $f : \Sigma \rightarrow \Sigma$ s.t.

- For all $\sigma \in \Sigma$, $\sigma|^{|_{V_c} = f(\sigma)|^{|_{V_a}}.
- For all $\sigma_c \in I_c$, there exist $\sigma_a \in I_a$ s.t. $\sigma_a = f(\sigma_c)$.
Given concrete machine specification \( S_c \) and abstract machine specification \( S_a \), the following rule does the job:

\[
\text{S}_c \xrightarrow{\text{ref}} \text{S}_a
\]

The following lemma expresses that refinement mappings are indeed sound for proving refinement of machine specifications.

**Lemma 8**

Given concrete machine specification \( S_c \overset{\Delta}{=} (B_c, \text{Comp}(M_c) \cap L_c) \) and abstract machine specification \( S_a \overset{\Delta}{=} (B_a, \text{Comp}(M_a) \cap L_a) \) s.t. \( \mathcal{D}(B_c) = \mathcal{D}(B_a) \). If there exists a refinement mapping from \( S_c \) to \( S_a \) then \( S_c \) ref \( S_a \).

### 2.4.2 Proving Refinement of DTL Specifications

Proving refinement of machine specifications in DTL means according to Theorem 4 that the observable bases are equal and that a formula with two existential quantifications is valid. More specifically:

Given concrete machine specification \( S_c \overset{\Delta}{=} (B_c, I_c \land \square T_c \land L_c) \) and abstract machine specification \( S_a \overset{\Delta}{=} (B_a, I_a \land \square T_a \land L_a) \). Then \( S_c \) refines \( S_a \) and defined by

\[
\mathcal{D}(B_c) = \mathcal{D}(B_a) \quad \text{and} \quad (\exists x_c . (I_c \land \square T_c \land L_c)) \rightarrow (\exists x_a . (I_a \land \square T_a \land L_a))
\]

So we must have a rule to prove the following:

\[
\exists x_0 . p_0 \rightarrow \exists x_1 . p_1
\]

The following rule does the job:

\[
p_0 \rightarrow p_1 [\text{exp}/x_1] \quad \text{for } x_0 \text{ not free in } p_1
\]

\[
\exists x_0 . p_0 \rightarrow \exists x_1 . p_1 \quad \text{none of the variables appearing in } \text{exp} \text{ is quantified in } p_1
\]

as the following derivation shows:

\[
\begin{align*}
p_0 & \rightarrow p_1 [\text{exp}/x_1] & \text{for } x_0 \text{ not free in } p_1 \\
\exists x_0 . p_0 & \rightarrow \exists x_1 . p_1 & \text{none of the variables appearing in } \text{exp} \text{ is quantified in } p_1
\end{align*}
\]
2.4 Proving Refinement of Reactive System Specifications

From the previous section it should be clear that this $\text{exp}$ is exactly the refinement mapping $f$, and that the proof can be split in a safety part and a liveness part (i.e., the proof of $p_0 \rightarrow p_1 [\text{exp} / x_1]$ of above rule is split into a safety and a liveness part). This culminates in the following proof rule for refinement based on similar ones in [Lam91, KMP93].

**Rule 1 (Proof rule for refinement)**

Given concrete machine specification $\mathcal{S}_c \triangleq (B_c, I_c \land \Box T_c \land L_c)$ and abstract machine specification $\mathcal{S}_a \triangleq (B_a, I_a \land \Box T_a \land L_a)$ s.t. $\mathcal{O}(B_c) = \mathcal{O}(B_a)$. Let $f$ be a refinement mapping from $\mathcal{S}_c$ to $\mathcal{S}_a$ then

$$
\begin{array}{l}
\mathcal{S}_c \models I_c \rightarrow I_a[f / X_a] \\
\mathcal{S}_c \models T_c \rightarrow T_a[f / X_a] \\
\mathcal{S}_c \models L_c[f / X_a]
\end{array}
$$

$$
\models (\exists X_c, (I_c \land \Box T_c \land L_c)) \rightarrow (\exists X_a, (I_a \land \Box T_a \land L_a))
$$

When $L_a$ is of the form

$$
\bigwedge_{\tau \in \text{WF}_a} (\Diamond \Box \text{En}(\tau) \rightarrow \Box \Diamond \tau) \land \bigwedge_{\tau \in \text{SF}_a} (\Box \Diamond \text{En}(\tau) \rightarrow \Box \Diamond \tau)
$$

(see Def. 29), the last premise of above rule can be split into

$$
\begin{array}{l}
\mathcal{S}_c \models \bigwedge_{\tau \in \text{WF}_a} (\Diamond \Box \text{En}(\tau) \rightarrow \Box \Diamond \tau)[f / X_a] \\
\mathcal{S}_c \models \bigwedge_{\tau \in \text{SF}_a} (\Box \Diamond \text{En}(\tau) \rightarrow \Box \Diamond \tau)[f / X_a]
\end{array}
$$

This is equal to

$$
\begin{array}{l}
\mathcal{S}_c \models \bigwedge_{\tau \in \text{WF}_a} (\text{En}(\tau) \Rightarrow \Diamond(\text{En}(\tau) \Rightarrow \tau))[f / X_a] \\
\mathcal{S}_c \models \bigwedge_{\tau \in \text{SF}_a} (\Box \Diamond \text{En}(\tau) \Rightarrow \Diamond \tau)[f / X_a]
\end{array}
$$

using some temporal logic calculus. So one gets the following proof rule, similar rules appearing in [Lam91, KMP93].

**Rule 2 (Proof rule for refinement)**

Given concrete machine specification $\mathcal{S}_c \triangleq (B_c, I_c \land \Box T_c \land L_c)$ where $L_c$ is of the form

$$
\bigwedge_{\tau \in \text{WF}_a} (\Diamond \Box \text{En}(\tau) \rightarrow \Box \Diamond \tau) \land \bigwedge_{\tau \in \text{SF}_a} (\Box \Diamond \text{En}(\tau) \rightarrow \Box \Diamond \tau)
$$

Furthermore given abstract machine specification $\mathcal{S}_a \triangleq (B_a, I_a \land \Box T_a \land L_a)$ where $L_a$ is of the form $\bigwedge_{\tau \in \text{WF}_a} (\Diamond \Box \text{En}(\tau) \rightarrow \Box \Diamond \tau) \land \bigwedge_{\tau \in \text{SF}_a} (\Box \Diamond \text{En}(\tau) \rightarrow \Box \Diamond \tau)$. Let $\mathcal{O}(B_c) = \mathcal{O}(B_a)$. Let $f$ be a refinement mapping from $\mathcal{S}_c$ to $\mathcal{S}_a$ then

$$
\begin{array}{l}
\mathcal{S}_c \models I_c \rightarrow I_a[f / X_a] \\
\mathcal{S}_c \models T_c \rightarrow T_a[f / X_a] \\
\mathcal{S}_c \models L_c[f / X_a]
\end{array}
$$

$$
\models (\exists X_c, (I_c \land \Box T_c \land L_c)) \rightarrow (\exists X_a, (I_a \land \Box T_a \land L_a))
$$

Rule 1 is used in the following example for proving refinement of example 4.
Example 5

From example 4 we have:
Let $H_c = \exists \varepsilon_1, \varepsilon_2, (B_1 \otimes B_2) (\varepsilon, \varepsilon_1, \varepsilon_2) \wedge (I_1 \wedge \Box T_1 \wedge L_1) [\varepsilon_1/\varepsilon] \wedge (I_2 \wedge \Box T_2 \wedge L_2) [\varepsilon_2/\varepsilon]$ and $H_a = I \wedge \Box T \wedge L$ then the composition of $S_{c_1}$ and $S_{c_2}$ refines $S_a$ iff

$$
\begin{align*}
(1) & \quad \mathcal{D}(B_c) = \mathcal{D}(B) \\
(2) & \quad \models (\exists t . u . (H_c)) \rightarrow (\exists x . (H_a))
\end{align*}
$$

Because the observable bases are equal (1) holds. (2) is proven with rule 1. This means one has to find a refinement mapping $f$. In order to find such a mapping the picture of the $S_{c_1} \parallel S_{c_2}$ is given. (Note only the reachable states are drawn):

![Transition diagram](image)

Figure 2.7: Transitions of $S_{c_1} \parallel S_{c_2}$

Relating the above figure with figure 2.4 one sees that $f$ is as defined follows:

$$
\text{if} \\
\begin{align*}
t &= 0 \wedge u = 0 & \text{then} & f(s, t, u) = t \\
t &= 1 \wedge u = 0 & \text{then} & f(s, t, u) = t \\
t &= 2 \wedge u = 1 & \text{then} & f(s, t, u) = t - u \\
t &= 2 \wedge u = 2 & \text{then} & f(s, t, u) = u
\end{align*}
$$

fi

The following premises should be valid in order to apply the rule:

- $S_c \models (s, t, u) = (0, 0, 0) \rightarrow ((s, x) = (0, 0)) [f/x]$  
  Substitution means replacing $x$ by $t$ because $t = 0 \wedge u = 0$. This results in:

$$(s, t, u) = (0, 0, 0) \rightarrow (s, t) = (0, 0)$$

This is valid.

- $S_c \models (\varepsilon = a! \wedge t = 0 \wedge (s, t, u)' = (s, 1, u))$

$\rightarrow (\varepsilon = a! \wedge x = 0 \wedge (s, x)' = (s, 1)) [f/x]$
2.4 Proving Refinement of Reactive System Specifications

Substitution means replacing $x$ by $t$ because $t = 0$, and replacing $x'$ by $t'$ because $t' = 1$. This results in:

$$S_c \models (\epsilon = a! \land t = 0 \land (s, t, u)' = (s, 1, u))$$
$$\rightarrow \quad (\epsilon = a! \land t = 0 \land (s, t)' = (s, 1))$$

This is valid.

- $S_c \models (\epsilon = b? \land u = 1 \land (s, t, u)' = (s, t, 2))$
$$\rightarrow \quad (\epsilon = b? \land x = 1 \land (s, x)' = (s, 2)) [f/x]$$

Substitution means replacing $x$ by $t - u$ because $u = 1$, and replacing $x'$ by $u'$ because $u' = 2$. This results in:

$$S_c \models (\epsilon = b? \land u = 1 \land (s, t, u)' = (s, t, 2))$$
$$\rightarrow \quad (\epsilon = b? \land t - u = 0 \land (s, u)' = (s, 2))$$

This is valid because from figure 2.7 one sees that $u = 1 \Rightarrow t = 2$ holds.

- $S_c \models (\epsilon = e \land (s, t, u)' = (1, t, u))$
$$\rightarrow \quad (\epsilon = e \land (s, x)' = (1, x)) [f/x]$$

Substitution means replacing $x$ by $f$, and replacing $x'$ by $f'$. This results in:

$$S_c \models (\epsilon = e \land (s, t, u)' = (1, t, u))$$
$$\rightarrow \quad (\epsilon = e \land (s, f)' = (s, f))$$

This is valid because $(t, u)' = (t, u) \Rightarrow f' = f$.

- $S_c \models (\epsilon = i \land (t, u) = (1, 0) \land (s, t, u)' = (s, 2, 1))$
$$\rightarrow \quad \text{stut}_e [f/x]$$

Because $\text{stut}_e \triangleq$

$$\epsilon = \lambda \quad \vee \quad (\epsilon = i \land (s, x)' = (s, x))$$
$$\vee \quad (\epsilon = e \land (s, x)' = (s, x))$$
it suffices to prove:

\[ S_c \models \left( \epsilon = i \land (t, u) = (1, 0) \land (s, t, u)' = (s, 2, 1) \right) \]
\[ \rightarrow \left( \epsilon = i \land (s, x)' = (s, x) \right)[f/x] \]

Substitution means replacing \( x \) by \( t \) because \( t = 1 \land u = 0 \), and replacing \( x' \) by \( t' - u' \) because \( t' = 2 \land u' = 1 \). This results in:

\[ S_c \models \left( \epsilon = i \land (t, u) = (1, 0) \land (s, t, u)' = (s, 2, 1) \right) \]
\[ \rightarrow \left( \epsilon = i \land (s, t - u)' = (s, t) \right) \]

This is valid.

- \( S_c \models \left( \epsilon = i \land (s, u) = (1, 2) \land (s, t, u)' = (0, t, u) \right) \]
\[ \rightarrow \left( \epsilon = i \land (s, x) = (1, 2) \land (s, x)' = (0, x) \right)[f/x] \]

Substitution means replacing \( x \) by \( u \) because \( u = 2 \land t = 2 \), and replacing \( x' \) by \( u' \) because \( u' = 2 \land t' = 2 \). This results in:

\[ S_c \models \left( \epsilon = i \land (s, u) = (1, 2) \land (s, t, u)' = (0, t, u) \right) \]
\[ \rightarrow \left( \epsilon = i \land (s, u) = (1, 2) \land (s, u)' = (s, u) \right) \]

This is valid.

- \( S_c \models \text{stut}_c \rightarrow \text{stut}_a[f/x] \)

Definition of \( \text{stut}_a \) and \( \text{stut}_c \) results in

\[ S_c \models \epsilon = \lambda \]
\[ \vee \left( \epsilon = i \land (s, t, u)' = (s, t, u) \right) \]
\[ \vee \left( \epsilon = e \land (s, t, u)' = (s, t, u) \right) \]
\[ \rightarrow \epsilon = \lambda \]
\[ \vee \left( \epsilon = i \land (s, x)' = (s, x) \right)[f/x] \]
\[ \vee \left( \epsilon = e \land (s, x)' = (s, x) \right)[f/x] \]

This is valid.

- \( S_c \models \text{true}[f/x] \)

This is valid.

So \( S_1 \models S_2 \) ref \( S_a \).
2.5 Relative Refinement and Composition of Reactive System Specifications

In this section the concept of relative refinement and composition in the development of systems is explained. Ordinary refinement stipulates that the set of histories generated by the concrete system is included in the set of histories generated by the abstract system. Relative refinement means that this inclusion almost holds, i.e., if one leaves some of the histories generated at the concrete level out of account this inclusion holds. Histories generated by the abstract system can also be left out because a concrete system could be an abstract system in a next refinement step. Ordinary composition means that the generated by the abstract system can also be left out because a concrete system could be a safety set, liveness set or neither of them. In the third subsection a condition similar to machine closedness is imposed on a relative system, i.e., the relative system can then be split into a safety part and a liveness part. Using this fact a proof rule for relative refinement is constructed in the last subsection based on rule given in Section 2.4. Again we formulate these concepts first in terms of sets of histories and then in DTL.

2.5.1 Semantic Relative Refinement and Composition of Specifications

Definition 39 (Relative refinement of systems)

Given concrete system \( S_c = (B_c, H_c) \) and a set \( W_c \) of allowed histories for \( S_c \) \((W_c \subseteq \mathcal{H} \) constraining \( B_c \)) and abstract system \( S_a = (B_a, H_a) \) together with a set \( W_a \) of allowed histories for \( S_a \) \((W_a \subseteq \mathcal{H} \) constraining \( B_a \)). Let \( G_c \triangleq H_c \cap W_c \) and \( G_a \triangleq H_a \cap W_a \).

Then \( S_c \) relatively refines \( S_a \) with respect to \((W_c,W_a)\), denoted by \( S_c \xrightarrow{W_c,W_a} S_a \), iff \( \mathcal{D}(B_c) = \mathcal{D}(B_a) \) and \( \mathcal{O}_{X_c}(G_c) \subseteq \mathcal{O}_{X_c}(G_a) \).

Relativizing can also be used for composition, i.e., if during composition one gets unwanted histories these are removed, using a set that characterizes the allowed histories.

Definition 40 (Relative composition of two systems)

Given systems \( S_i = (B_i, H_i) \) where \( B_i = ((\text{In}_i, \text{Out}_i), (V_i, X_i)) \) and \( i = 1, 2 \) such that \( X_1 \cap X_2 = \emptyset \) and given sets \( W_i \subseteq \mathcal{H} \) constraining \( B_i \). Let \( \bar{W} \) denote \((W_1, W_2)\). Then the relative composed system \( S \) with respect to \( \bar{W} \), denoted \( S \xrightarrow{\bar{W}} S \), is defined as \((B, H)\) with \( B \triangleq ((\text{In}_1 \setminus \text{Out}_2 \cup \text{In}_2 \setminus \text{Out}_1, \text{Out}_1 \setminus \text{In}_2 \cup \text{Out}_2 \setminus \text{In}_1), (V_1 \cup V_2, X_1 \cup X_2)) \), and \( H \triangleq H_1(\mathcal{W})H_2 \triangleq (H_1 \cap W_1) \oplus (H_2 \cap W_2) \).

The following is a compositional relative refinement theorem.

Theorem 7 (Compositional relative refinement)

Given concrete systems \( S_i = (B_i, H_i) \) \((i = 1, 2)\) and given set \( W_c \) constraining \( B_{12} \) (the
Theorem /8/ (Relative refinement to DTL machine specifications) 

Given abstract systems $S_j = (B_j, H_j)$ ($j = 3, 4$) and given set $W_a$ constraining $B_{34}$ (the basis of $S_3 \parallel S_4$). Then the following holds:

$$
\begin{align*}
(H_1 \otimes H_2) \cap (W_1 \otimes W_2) &\subseteq (H_1 \cap W_1) \otimes (H_2 \cap W_2) \\
W_e &\subseteq W_1 \otimes W_2 \\
W_{e3} \otimes W_{e4} &\subseteq W_a \\
S_1 \mathrel{w_1 \text{ref}} W_{e3} S_3 &\quad W_{e2} \text{ constraining } B_i \ (i = 1, 2) \\
S_2 \mathrel{w_2 \text{ref}} W_{e4} S_4 &\quad W_{aj} \text{ constraining } B_j \ (j = 3, 4)
\end{align*}
$$

If the extra requirements $W$ don’t constrain the $\epsilon$-variables then the following lemma can be used to prove the first premise of above theorem.

**Lemma 9**

Given systems $S_i = (B_i, H_i)$ and sets $W_i$ constraining $B_i$ ($i = 1, 2$) with no restrictions on the event variables. Then the following holds:

$$
(H_1 \cap W_1) \otimes (H_2 \cap W_2) = H_1 \otimes H_2 \cap W_1 \otimes W_2
$$

In case the abstract requirement $W_a$ can’t be decomposed into component requirements the following rule can be used.

**Lemma 10**

Given concrete systems $S_i = (B_i, H_i)$ ($i = 1, 2$) and given set $W_i$ constraining $B_{12}$. And given abstract systems $S_j = (B_j, H_j)$ ($j = 3, 4$) and given set $W_a$ constraining $B_{34}$ without restricting the $\epsilon$ variables. Then the following holds:

$$

\begin{align*}
H_1 \otimes H_2 \cap W_1 \otimes W_2 &\subseteq (H_1 \cap W_1) \otimes (H_2 \cap W_2) \\
W_e &\subseteq W_1 \otimes W_2 \\
S_1 \mathrel{w_1 \text{ref}} W_{e} S_3 &\quad W_{e2} \text{ constraining } B_i \ (i = 1, 2) \\
S_2 \mathrel{w_2 \text{ref}} W_{e} S_4 &\quad W_{ej} \text{ constraining } B_j \ (j = 3, 4)
\end{align*}
$$

The following lemma is useful for proving the second premise of the theorem.

**Lemma 11**

Given sets $W_i$ ($i = 1, 2$) not restricting the $\epsilon$ variables then

$$
W_1 \otimes W_2 = W_1 \cap W_2.
$$

2.5.2 Relative Refinement and Composition of DTL Specifications

**Theorem 8 (Relative refinement of DTL machine specifications)**

Given concrete machine specification $S_c \triangleq (B_c, I_c \land \Box T_c \land L_c)$ and DTL formula $W_c$ over $B_c$ and abstract machine specification $S_a \triangleq (B_a, I_a \land \Box T_a \land L_a)$ and DTL formula $W_a$ over $B_a$. Let $G_c \triangleq I_c \land \Box T_c \land L_c \land W_c$ and $G_a \triangleq I_a \land \Box T_a \land L_a \land W_a$. Then $S_c \mathrel{Hist(W_c) \text{ref}} Hist(W_a) S_a$ if

$$
\mathcal{D}(B_c) = \mathcal{D}(B_a) \text{ and } \mathcal{I}(\exists X_c . (G_c)) \Rightarrow (\exists X_a . (G_a))
$$
2.5 Relative Refinement and Composition of Reactive System Specifications

**Definition 41 (Relative composition of two DTL machine specifications)**

Given machine system specifications \( (B_i, I_i \wedge \square T_i \wedge L_i) \) where \( B_i \triangleq ((I_{i_1}, \text{Out}_{i_1}), (V_{i_1}, X_{i_1})) \), and given DTL formulae \( W_i \) over \( B_i \) for \( i = 1, 2 \). Then the relative composed machine specification \( S \) w.r.t. \( W \) is defined as \( (B, H) \) where \( B \triangleq ((I_{1_1}, \text{Out}_{1_1} \cup I_{2_1}, \text{Out}_{2_1} \setminus \text{Out}_{1_1} \cup \text{Out}_{1_2} \setminus \text{Out}_{2_1} \setminus I_{1_1}, (V_{1_1} \cup V_{2_1}, X_{1_1} \cup X_{2_1})) \) and \( H \triangleq \exists \epsilon_1, \epsilon_2, (B^\epsilon_1 \cap B^\epsilon_2 (\epsilon, \epsilon_1, \epsilon_2) \wedge (I_1 \wedge \square T_1 \wedge L_1 \wedge W_1) [\epsilon_1 / \epsilon] \wedge (I_2 \wedge \square T_2 \wedge L_2 \wedge W_2) [\epsilon_2 / \epsilon] \).

**Theorem 9 (Relative composition corresponds to semantic merge)**

Given machine system specifications \( (B_i, I_i \wedge \square T_i \wedge L_i) \) where \( B_i \triangleq ((I_{i_1}, \text{Out}_{i_1}), (V_{i_1}, X_{i_1})) \), and given DTL formulae \( W_i \) over \( B_i \) for \( i = 1, 2 \) and let \( \overline{W} \triangleq (\text{Hist}(W_1), \text{Hist}(W_2)) \) and given the relative composed system as in Def. 41, i.e., \( (B, H) \) where \( B \triangleq ((I_{1_1} \setminus \text{Out}_{1_2} \cup \text{Out}_{1_1} \setminus \text{Out}_{1_2} \setminus \text{Out}_{2_1} \setminus I_{1_1}, (V_{1_1} \setminus V_{2_1}, X_{1_1} \setminus X_{2_1})) \) and \( H \triangleq \exists \epsilon_1, \epsilon_2, (B^\epsilon_1 \cap B^\epsilon_2 (\epsilon, \epsilon_1, \epsilon_2) \wedge (I_1 \wedge \square T_1 \wedge L_1 \wedge W_1) [\epsilon_1 / \epsilon] \wedge (I_2 \wedge \square T_2 \wedge L_2 \wedge W_2) [\epsilon_2 / \epsilon] \) then

\[
\text{Hist}(I_1 \wedge \square T_1 \wedge L_1) \overline{W} \text{Hist}(I_2 \wedge \square T_2 \wedge L_2) = \text{Hist}(H)
\]

### 2.5.3 Proving Semantic Relative Refinement of Specifications

The above sections explain the purpose of the restricting set \( W \). In order to prove relative refinement we must know how this set \( W \) looks like. Is it a safety set, a liveness set or neither of them? A result of [AS85] states that every set of histories can be represented as the intersection of a safety and a liveness set. Now lemma 1 expresses that for a machine \( M_1 \), \( \text{Comp}(M_1) \) is a safety set. So we will represent \( W \) as a machine \( M_1 \) and an external set \( L_1 \). \( W \) is machine closed, i.e., \( \text{cl} (\text{Comp}(M_1) \cap L_1) = \text{Comp}(M_1) \). We also require that \( (B, \text{Comp}(M) \cap \text{Comp}(M_1) \cap L \cap L_1) \) is machine closed, i.e., \( \text{cl} (\text{Comp}(M) \cap \text{Comp}(M_1) \cap L \cap L_1) = \text{Comp}(M) \cap \text{Comp}(M_1) \), because this is the system that is used in the relative refinement relation. We want to use the refinement mappings of Def. 38 to prove relative refinement of systems. This means that \( \text{Comp}(M) \cap \text{Comp}(M_1) \) should be represented as a machine \( M_2 \) such that \( \text{Comp}(M_2) = \text{Comp}(M) \cap \text{Comp}(M_1) \). The following lemma expresses that this \( M_2 \) can be constructed from \( M \) and \( M_1 \).

**Lemma 12**

Given machines \( M \triangleq (B, I, T) \) and \( M_1 \triangleq (B, I_1, T_1) \). Define machine \( M_2 \) as \( (B, I_2, T_2) \) where \( I_2 \) and \( T_2 \) are as follows:

- \( I_2 \triangleq I \cap I_1 \), and
- \( T_2 \triangleq T \cap T_1 \).

Then \( \text{Comp}(M_2) = \text{Comp}(M) \cap \text{Comp}(M_1) \).

Now the technique of refinement mappings from Section 2.4 can be applied to prove relative refinement of systems. This is expressed in the following definition.

**Definition 42 (Relative refinement mapping between machine specifications)**

Given concrete machine specification \( S_c \triangleq (B_c, \text{Comp}(M_c) \cap L_c) \) and set \( W_c = \text{Comp}(M_{c1}) \cap L_{c1} \), and given abstract machine specification \( S_a \triangleq (B_a, \text{Comp}(M_a) \cap L_a) \) and set \( W_a = \text{Comp}(M_{a1}) \cap L_{a1} \). A relative refinement mapping from machine specification \( S_c \) to machine specification \( S_a \) is a mapping \( f : \Sigma \to \Sigma \) s.t.
A Dense Model Formalism

- For all \( \sigma \in \Sigma \), \( \sigma\vert_{V_c} = f(\sigma)\vert_{V_c} \).
- For all \( \sigma \in I_c \cap I_{a1} \), exist \( \sigma_a \in I_a \cap I_{a1} \) s.t. \( \sigma_a = f(\sigma_c) \).
- For all \( (d, \sigma_{a1}, \sigma_{c1}) \in T_c \cap T_{c1} \), \( (d, f(\sigma_{a1}), f(\sigma_{c1})) \in T_a \cap T_{a1} \) or \( (f(\sigma_{a1}) = f(\sigma_{c1}) \land d(\epsilon) \in \{\lambda, i, e\}) \).
- For all \( h_c \in \text{Comp}(M_c) \cap \text{Comp}(M_{a1} \cap I_c) \) there exist \( h_a \in I_a \cap I_{a1} \) s.t. for all \( t \in \mathbb{R}^\geq, (\psi_c(t), \theta_c(t)) = (\psi_a(t), \theta(t)_{a}) \) and \( f(\theta_c(t)) = \theta_a(t) \).

The following lemma expresses that relative refinement mappings are indeed sufficient for proving relative refinement of machine specifications.

**Lemma 13**

Given concrete machine specification \( S_c \overset{\Delta}{=} (B_c, \text{Comp}(M_c) \cap I_c) \) and set \( W_c = \text{Comp}(M_{c1}) \cap I_{c1} \), and given abstract machine specification \( S_a \overset{\Delta}{=} (B_a, \text{Comp}(M_a) \cap I_a) \) and set \( W_a = \text{Comp}(M_{a1}) \cap I_{a1} \) s.t. \( \mathcal{D}(B_c) = \mathcal{D}(B_a) \). If there exists a relative refinement mapping from \( S_c \) to \( S_a \) then \( S_c \text{ ref } W_c \ S_a \).

### 2.5.4 Proving Relative Refinement of DTL Specifications

Using the results of the previous section and Section 2.4 it is not surprising that following rule can be applied to prove relative refinement of systems.

**Rule 3 (Proof rule for relative refinement)**

Given concrete machine specification \( S_c \overset{\Delta}{=} (B_c, I_c \land \Box T_c \land I_c) \) and \( W_c \overset{\Delta}{=} I_{c1} \land \Box T_{c1} \land I_{c1} \) and abstract machine specification \( S_a \overset{\Delta}{=} (B_a, I_a \land \Box T_a \land I_a) \) and \( W_a \overset{\Delta}{=} I_{a1} \land \Box T_{a1} \land I_{a1} \) s.t. \( \mathcal{D}(B_c) = \mathcal{D}(B_a) \). Let \( f \) be a relative refinement mapping from \( S_c \) to \( S_a \) then

\[
\begin{align*}
S_c \cap \text{Hist}(W_c) \models & (I_c \land I_{c1}) \rightarrow (I_a \land I_{a1}) \lbrack f \rvert_{X_a} \\
S_c \cap \text{Hist}(W_c) \models & (T_c \land T_{c1}) \rightarrow (T_a \land T_{a1}) \lbrack f \rvert_{X_a} \\
S_c \cap \text{Hist}(W_c) \models & (L_c \land L_{c1}) \lbrack f \rvert_{X_a} \\
\models & (\exists X_c \cdot (I_c \land \Box T_c \land I_c \land W_c)) \rightarrow (\exists X_a \cdot (I_a \land \Box T_a \land I_a \land W_a))
\end{align*}
\]
Chapter 3
Readers/Writers Example

3.1 Introduction

The relative refinement technique will now be used to formalize Dijkstra's development strategy for the readers/writers problem. The readers/writers problem, described intuitively, is as follows: given $N$ readers and $M$ writers, a reader performs, cyclically, non-critical action \texttt{NCS} and critical action \texttt{READ}, and a writer performs, again cyclically, non-critical action \texttt{NCS} and critical action \texttt{WRITE}. These readers and writers must be synchronized in such a way that if a writer performs the \texttt{WRITE} action it is the only process that performs a critical action, i.e. mutual exclusion is required (ME). Furthermore, it is necessary that any request to execute the critical action is eventually granted, i.e. eventual access should hold (EA). It is this synchronizer that has to be developed. But before we give the development we formulate an abstract specification for the problem.

The abstract specification of Dijkstra consists of a program, implementing the above readers and writers, and the requirements ME and EA. In our formalism this will be represented by system $S_0$ and requirement $W_0$. The development process has four steps: in the first step Dijkstra gives an implementation by a program that produces undesirable deadlocked computations. In our formalism the first implementation is represented by system $S_1$ and a requirement $W_1$ which removes the deadlocked computations. We will prove that $S_1$ relatively refines $S_0$ with respect to $(W_1,W_0)$. In the second step Dijkstra uses the split binary semaphore technique to delete the deadlocked computations from the first implementation; he obtains by this technique a second implementation that introduces as undesirable computations new deadlocked ones. In our formalism the second implementation is represented by the system $S_2$ and the requirement $W_2$ that removes the newly introduced deadlocked computations. We will prove that $S_2$ relatively refines $S_1$ with respect to $(W_2,W_1)$. These deadlocked computations are deleted in the third step resulting in a third implementation that contains as undesirable computations unnecessarily blocking ones. These computations are not deadlocking computations but only computations that are inefficient because they suspend a reader or writer unnecessarily. In our formalism the third implementation will be represented by the system $S_3$ and the requirement $W_3$ that removes the unnecessarily blocking computations. It is proved that $S_3$ relatively refines $S_2$ with respect to $(W_3,W_2)$. In the fourth step, these unnecessarily
Readers/Writers Example

blocking computations are deleted and also the resulting implementation is cleaned up. In
our formalism the fourth implementation will be represented by system \( S_4 \) and it is proved
that \( S_4 \) relatively refines \( S_3 \) with respect to \((\text{true}, W_3)\), i.e., in the fourth step no further
requirements are imposed.

3.2 The abstract specification

Here Dijkstra’s strategy [Dij79] is followed and it is shown how the informal approach used
there can be formalized.
Dijkstra rewrites the informal specification as follows: as a first step, he describes readers
and writers by programs (he assumes that the semantics of these programs is intuitively
clear):

\[
\text{reader}_i^0: \quad \text{do true} \rightarrow \text{NCS;} \text{READ} \quad \text{od}
\]
\[
\text{writer}_j^0: \quad \text{do true} \rightarrow \text{NCS;} \text{WRITE} \quad \text{od}
\]

He then combines these programs into one parallel program \( \text{Syn}^0 \). \( \text{Syn}^0 \) denotes the abstract
specification and is defined as follows:

\[
\text{Syn}^0 : \|_{i=1}^N \text{reader}_i^0 \|_{j=1}^M \text{writer}_j^0,
\]

where \(\|_{i=1}^N \text{reader}_i^0\) is a notation for the \( N \)-fold parallel composition of \( \text{reader}_i^0 \). Finally
he formulates an informal requirement to exclude from \( \text{Syn}^0 \) the unwanted sequences.
This requirement is the same as in the introduction: \( \text{ME} \) and \( \text{EA} \). The complete abstract
specification is thus \( \text{Syn}^0 \) plus this requirement.

Each \( \text{reader}_i^0 \) and \( \text{writer}_j^0 \) is represented respectively by DTL machine specification \( S_{r_i}^0 \)
and \( S_{w_j}^0 \). We will incorporate the requirement \( \text{EA} \) as a liveness requirement in each machine
specification. The parallel composition of all the separate machine specifications \( S_0 = \|_{i=1}^N
S_{r_i}^0 \|_{j=1}^M S_{w_j}^0 \) then corresponds to \( \text{Syn}^0 \) plus \( \text{EA} \). \( \text{ME} \) will be incorporated as an extra
requirement on \( S_0 \). The following sections will give in detail the machine specifications \( S_{r_i}^0 \)
and \( S_{w_j}^0 \), and the extra requirement \( W_0 \).

3.2.1 Specification \( S_{r_i}^0 \)

The formal specification \( S_{r_i}^0 = (B_{r_i}, H_{r_i}) \) where \( H_{r_i} \triangleq L_{r_i} \land \Box T_{r_i} \land L_{r_i} \) and \( B_{r_i}, L_{r_i}, T_{r_i} \)
and \( L_{r_i} \) are as follows:

1. Basis \( B_{r_i} \triangleq (\text{In}_{r_i}, \text{Out}_{r_i}, (V_{r_i}, X_{r_i})) \)

\[
\begin{align*}
\text{Out}_{r_i} & \triangleq \emptyset, \\
\text{In}_{r_i} & \triangleq \emptyset, \\
V_{r_i} & \triangleq \{s_r\}, \\
X_{r_i} & \triangleq \emptyset
\end{align*}
\]
3.2 The abstract specification

- \( s_r = 0 \): reader\(_i^0 \) is non critical.
- \( s_r = 1 \): reader\(_i^0 \) is critical.

2. Initial States:

\[ I_r^0 = s_r = 0 \]

Reader\(_i^0 \) starts in the non critical state.

3. Transitions:

\[ T_r^0 = \]

\[ \tau_{r_i,1} = (\epsilon = i \land s_r = 0 \land s'_r = 1) \]

Reader\(_i^0 \) becomes critical.

\[ \tau_{r_i,2} = \lor (\epsilon = i \land s_r = 1 \land s'_r = 0) \]

Reader\(_i^0 \) becomes critical.

\[ \tau_{r_i,0} = \lor \text{stut}_{r_i} \]

These transitions are illustrated in figure 3.1

4. Liveness

As discussed above \( L_r \) should express the \( \mathbb{E}A \) requirement, i.e., all the transitions are weakly fair.

Let \( WF_r^i = \{ \tau_{r_i,k} \mid k \in \{1, 2\} \} \) and \( SF_r^i = \emptyset \) then

\[ L_r^i = \bigwedge_{\tau \in WF_r^i} (\Diamond \Box En(\tau) \rightarrow \Box \Diamond \tau) \land \bigwedge_{\tau \in SF_r^i} (\Box \Diamond En(\tau) \rightarrow \Box \Diamond \tau) \]

3.2.2 Specification \( S_{w_j} \)

The formal specification \( S_{w_j} = (B_{w_j}, H_{w_j}) \) where \( H_{w_j} = I_{w_j} \land \Box T_{w_j} \land L_{w_j} \) and \( B_{w_j}, I_{w_j}, T_{w_j} \) and \( L_{w_j} \) are as follows:

49
1. **Basis** $B_{w_j} \triangleq ((\text{In}_{w_j}, \text{Out}_{w_j}), (V_{w_j}, X_{w_j}))$

\[
\begin{align*}
\text{Out}_{w_j} & \triangleq \emptyset, \\
\text{In}_{w_j} & \triangleq \emptyset, \\
V_{w_j} & \triangleq \{s_{w_j}\}, \\
X_{w_j} & \triangleq \emptyset
\end{align*}
\]

- $s_{w_j} = 0$: writer$_j^0$ is non critical.
- $s_{w_j} = 1$: writer$_j^0$ is critical.

2. **Initial States:**

\[I_{w_j} \triangleq s_{w_j} = 0\]

Writer$_j^0$ starts in the non critical state.

3. **Transitions:**

\[T_{w_j} \triangleq \]

\[
\begin{align*}
\tau_{w_j,1} & \quad (\epsilon = i \land s_{w_j} = 0 \land s'_{w_j} = 1) \\
\text{Writer}_{j} \text{ becomes critical.}
\end{align*}
\]

\[
\begin{align*}
\tau_{w_j,2} & \quad \vee (\epsilon = i \land s_{w_j} = 1 \land s'_{w_j} = 0) \\
\text{Writer}_{j} \text{ becomes non critical.}
\end{align*}
\]

\[
\begin{align*}
\tau_{w_j,0} & \quad \vee \text{st ut}_{w_j}
\end{align*}
\]

These transitions are illustrated in figure 3.2

![Figure 3.2: Transitions of writer$_j^0$.](image)

4. **Liveness**

As discussed above $I_{w_j}$ should express the EA requirement, i.e., all the transitions are weakly fair.

Let $WF_{w_j} \triangleq \{\tau_{w_j,k} \mid k \in \{1, 2\}\}$ and $SF_{w_j} \triangleq \emptyset$ then

\[I_{w_j} \triangleq \bigwedge_{\tau \in WF_{w_j}} (\Diamond \Box En(\tau) \rightarrow \Box \Diamond \tau) \land \bigwedge_{\tau \in SF_{w_j}} (\Box \Diamond En(\tau) \rightarrow \square \Diamond \tau)\]
3.2.3 Requirement $W_0$

The extra condition on the composed system should express the mutual exclusion property $ME$. A reader (writer) is critical if $s_r = 1$ ($s_w = 1$). Let $\bar{z}(i : 1 \leq i \leq N : s_r = 1)$ denote the number of components such that $s_r = 1$. The condition is then as follows:

$$W_0 \triangleq \square \left( \bar{z}(j : 1 \leq j \leq M : s_{w_j} = 1) = 0 \lor \bar{z}(i : 1 \leq i \leq N : s_r = 1) = 0 \land \bar{z}(j : 1 \leq j \leq M : s_{w_j} = 1) = 1 \right)$$

As seen in Section 2.5 $W_0$ should be defined as a machine and a liveness condition in order to apply the proof rule for relative refinement. This can be done quite easily. The liveness condition is $\text{true}$. Define $p$ as $\bar{z}(j : 1 \leq j \leq M : s_{w_j} = 1) = 0 \lor \bar{z}(i : 1 \leq i \leq N : s_r = 1) = 0 \land \bar{z}(j : 1 \leq j \leq M : s'_{w_j} = 1) = 0 \lor \bar{z}(i : 1 \leq i \leq N : s'_r = 1) = 0 \land \bar{z}(j : 1 \leq j \leq M : s'_w = 1) = 1)$. Then $p \land \square((p \land p') \lor \text{stut}_0)$ is the machine in DTL corresponding to $W_0$.

3.3 The first development step

Dijkstra’s next step is to translate the informally stated requirement into formal program form, i.e. to transform reader$^i$ and writer$^j$ in such a way that they satisfy the mutual-exclusion requirement $ME$. We discuss this translation informally.

He introduces shared variables $aw$ and $ar$ and binary semaphore $x$. Shared variable $ar$ represents the number of readers which may execute their $READ$, and $aw$ represents the number of writers which may execute their $WRITE$. A reader increases $ar$ by 1 if it is allowed to execute its $READ$ and decreases $ar$ by 1 if it is finished with executing its $READ$. Since $ar$ will be changed and accessed by several readers, Dijkstra protects the operation of increasing and decreasing $ar$ by semaphore operations $P$ and $V$ on binary semaphore $x$ to ensure that only one reader changes $ar$ at a time, i.e. mutual exclusion. The synchronization requirement is brought into reader$^i$ by guarding the increasing operation of $ar$ with condition $aw = 0$, i.e., the number of writers that may execute their $WRITE$ equals zero. The same can be done for writer$^j$. The initial values of the shared variables are 0 and the initial value of semaphore $x$ is 1. This results in the following programs:

reader$^i$:
   \begin{align*}
   \text{do true} & \rightarrow \text{NCS;} \\
   & P(x);(*) \text{ if } aw = 0 \rightarrow ar := ar + 1 \text{ fi;} V(x); \text{ READ;} \\
   & P(x); ar := ar - 1; V(x)
   \end{align*}

od

writer$^j$:
   \begin{align*}
   \text{do true} & \rightarrow \text{NCS;} \\
   & P(x); (+) \text{ if } aw = 0 \land ar = 0 \rightarrow aw := aw + 1 \text{ fi;} V(x); \text{ WRITE;} \\
   & P(x); aw := aw - 1; V(x)
   \end{align*}

od

$$\text{Syn}^1 : \bigvee_{i=1}^N \text{ reader}^i \| \bigvee_{j=1}^M \text{ writer}^j$$

51
This first approximation can deadlock. A deadlocked sequence is for instance:

A writer starts in the initial state and then executes \( \text{NCS;} P(x); (+) \), as result of that the value of \( aw \) changes in 1. A reader then executes \( \text{NCS;} P(x); (*) \) and blocks in the \( \text{if-fi} \) clause of \( (*) \) because \( aw = 1 \) and the semantics of this \( \text{if-fi} \) is such that when no guard is fulfilled it blocks. Then no reader or writer can then execute \( (*); \) or \( (+) \) because \( x = 0 \) and \( x \) holds this value forever. The requirement is thus that these deadlocked sequences are not generated.

Now \( \text{Syn}^1 \) will be specified in Stark’s formalism. Like the abstract specification each reader \( r_i \) and writer \( w_j \) is represented by a separate machine specification \( S_{r_i} \) and \( S_{w_j} \). The composed system \( S_1 = \prod_{i=1}^N S_{r_i} \parallel \prod_{j=1}^M S_{w_j} \) and corresponds with \( \text{Syn}^1 \). For \( S_1 \) the extra requirement \( W_1 \) for excluding deadlocked computations is formulated. In the following subsections we give DTL machine specifications \( S_{r_i} \) and \( S_{w_j} \), and the extra requirement \( W_1 \).

### 3.3.1 Specification \( S_{r_i} \)

The formal specification \( S_{r_i} = (B_{r_i}, H_{r_i}) \) where \( H_{r_i} = L_{r_i} \land \Box T_{r_i} \land L_{r_i} \) and \( B_{r_i}, L_{r_i}, T_{r_i} \)

1. **Basis** \( B_{r_i} = ((\text{In}_{r_i}, \text{Out}_{r_i}),(\text{V}_{r_i}, \text{X}_{r_i})) \)

   \[
   \begin{align*}
   \text{In}_{r_i} & \overset{\Delta}{=} \emptyset, \\
   \text{Out}_{r_i} & \overset{\Delta}{=} \emptyset, \\
   \text{V}_{r_i} & \overset{\Delta}{=} \{x, ar, aw, s_r\}, \\
   \text{X}_{r_i} & \overset{\Delta}{=} \{\ell_{r_i}\}
   \end{align*}
   \]

   - \( \ell_{r_i} = 0 \): reader \( r_i \) is non critical.
   - \( \ell_{r_i} = 1 \): reader \( r_i \) has passed its first P-operation.
   - \( \ell_{r_i} = 2 \): reader \( r_i \) has increased \( ar \) by 1.
   - \( \ell_{r_i} = 3 \): reader \( r_i \) is critical.
   - \( \ell_{r_i} = 4 \): reader \( r_i \) has passed its second P-operation.
   - \( \ell_{r_i} = 5 \): reader \( r_i \) has decreased \( ar \) by 1.

2. **Initial States:**

   \( L_{r_i} = \Psi_1 = (1, 0, 0, 0, 0) \)

3. **Transitions:**

   \( T_{r_i} \)

---

52
3.3 The first development step

\( \tau_{r_1}^{i_1} \quad (\epsilon = 1 \land \ell_{r_1} = 0 \land x = 1 \land \Psi'_1 = \Psi_1[0, 1/x, \ell_{r_1}]) \)

Reader\(_i^1\) executes the first P-action.

\( \tau_{r_1}^{i_2} \quad \forall (\epsilon = 1 \land \ell_{r_1} = 1 \land aw = 0 \land \Psi'_1 = \Psi_1[ar + 1, 2/ar, \ell_{r_1}]) \)

Reader\(_i^1\) can increase the number of active readers by if the number of active writers is zero.

\( \tau_{r_1}^{i_3} \quad \forall (\epsilon = 1 \land \ell_{r_1} = 2 \land \Psi'_1 = \Psi_1[1, 1, 3/s_., x, \ell_{r_1}]) \)

Reader\(_i^1\) becomes critical.

\( \tau_{r_1}^{i_4} \quad \forall (\epsilon = 1 \land \ell_{r_1} = 3 \land x = 1 \land \Psi'_1 = \Psi_1[0, 4/x, \ell_{r_1}]) \)

Reader\(_i^1\) executes its second P-action.

\( \tau_{r_1}^{i_5} \quad \forall (\epsilon = 1 \land \ell_{r_1} = 4 \land \Psi'_1 = \Psi_1[ar - 1, 5/ar, \ell_{r_1}]) \)

Reader\(_i^1\) decreases the number of active readers by one.

\( \tau_{r_1}^{i_6} \quad \forall (\epsilon = 1 \land \ell_{r_1} = 5 \land \Psi'_1 = \Psi_1[0, 1, 0/s_., x, \ell_{r_1}]) \)

Reader\(_i^1\) becomes non critical.

\( \tau_{r_1}^{i_7} \quad \forall (\epsilon = 1 \land x = 1 \land \Psi'_1 = \Psi_1[0/x]) \)

The environment executes a P-operation on x.

\( \tau_{r_1}^{i_8} \quad \forall (\epsilon = 1 \land \ell_{r_1} \in \{0, 3\} \land x = 0 \land \Psi'_1 = \Psi_1[1/x]) \)

The environment executes a V-operation on x.

\( \tau_{r_1}^{i_9} \quad \forall \text{ stat}_{r_1} \)

These transitions are illustrated in figure 3.3

4. Liveness:

L\(_i^1\) expresses that the P- and V-operations on the semaphore x are strongly fair and all the other transitions are weakly fair.

Let WF\(_{r_1} \triangleq \{\tau_{r_1}^{i, k} \mid k \in \{2, 5\}\} \) and SF\(_{r_1} \triangleq \{\tau_{r_1}^{i, k} \mid k \in \{1, 3, 4, 6, 7, 8\}\} \) then

\[ L_{r_1} \triangleq \bigwedge_{\tau \in \text{WF}_{r_1}} (\diamond \Box En(\tau) \rightarrow \Box \diamond \tau) \land \bigwedge_{\tau \in \text{SF}_{r_1}} (\Box \diamond En(\tau) \rightarrow \Box \diamond \tau) \]

3.3.2 Specification \( S_{w_j} \)

The formal specification \( S_{w_j} \triangleq (B_{w_j}, H_{w_j}) \) where \( H_{w_j} \triangleq I_{w_j} \wedge \Box T_{w_j} \wedge I_{w_j} \) and \( B_{w_j}, I_{w_j}, T_{w_j} \) and \( I_{w_j} \) are as follows:

1. Basis \( B_{w_j} = ((In_{w_j}, Out_{w_j}), (V_{w_j}, X_{w_j})) \)

\[ \text{In}_{w_j} \triangleq \emptyset, \]
\[ \text{Out}_{w_j} \triangleq \emptyset, \]
\[ V_{w_j} \triangleq \{x, ar, aw, s_{w_j}\}, \]
\[ X_{w_j} \triangleq \{\ell_{w_j}\} \]

53
Transitions:

\[
\begin{array}{c}
(\ell_{r_1^1}, s_{r_1}, x) = (0, 0, 0) \\
(\ell_{r_1^1}, s_{r_1}, x) = (0, 0, 1) \\
(\ell_{r_1^1}, s_{r_1}, x) = (1, 0, 0) \\
(\ell_{r_1^1}, s_{r_1}, x) = (2, 0, 0) \\
(\ell_{r_1^1}, s_{r_1}, x) = (3, 1, 0) \\
(\ell_{r_1^1}, s_{r_1}, x) = (3, 1, 0)
\end{array}
\]

Figure 3.3: Transitions of reader_1.

- $\ell_{w_j^1} = 0$: writer_j is non-critical.
- $\ell_{w_j^1} = 1$: writer_j has passed its first P-operation.
- $\ell_{w_j^1} = 2$: writer_j has increased aw by 1.
- $\ell_{w_j^1} = 3$: writer_j is critical.
- $\ell_{w_j^1} = 4$: writer_j has passed its second P-operation.
- $\ell_{w_j^1} = 5$: writer_j has decreased aw by 1.

Let $\Psi_1 = (x, ar, aw, s_{w_j}, \ell_{w_j^1})$ and $\Psi'_1 = (x', ar', aw', s'_{w_j}, \ell'_{w_j^1})$.

2. Initial States:

\[ I_{w_j^1} \triangleq \Psi_1 = (1, 0, 0, 0) \]

3. Transitions:

\[ T_{w_j^1} = \]

\[ \tau_{w_j^1, 1} \quad \left( \epsilon = 1 \land \ell_{w_j^1} = 0 \land x = 1 \land \Psi'_1 = \Psi_1 \left[ 0, 1/x, \ell_{w_j^1} \right] \right) \]

Writer_j executes the first P-action.

\[ \tau_{w_j^1, 2} \quad \left( \epsilon = 1 \land \ell_{w_j^1} = 1 \land ar = 0 \land aw = 0 \land \Psi'_1 = \Psi_1 \left[ aw + 1, 2/aw, \ell_{w_j^1} \right] \right) \]

Writer_j can increase the numbers of active writers if the number of active writers and readers is zero.
3.3 The first development step

\[ \tau_{w_{j,3}} \quad \vee \quad (\epsilon = i \land \ell_{w_i} = 2 \land \Psi' = \Psi_1[1, 1, 3/s_{w_j}, x, \ell_{w_j}]) \]

Writer\(^1\) becomes critical.

\[ \tau_{w_{j,4}} \quad \vee \quad (\epsilon = i \land \ell_{w_i} = 3 \land x = 1 \land \Psi' = \Psi_1[0, 4/x, \ell_{w_j}]) \]

Writer\(^1\) executes the second P-action.

\[ \tau_{w_{j,5}} \quad \vee \quad (\epsilon = i \land \ell_{w_i} = 4 \land \Psi' = \Psi_1[a w - 1, 5/a w, \ell_{w_j}]) \]

Writer\(^1\) decreases the number of active writers by one.

\[ \tau_{w_{j,6}} \quad \vee \quad (\epsilon = e \land x = 1 \land \Psi' = \Psi_1[0/x]) \]

The environment executes a P-operation on x.

\[ \tau_{w_{j,7}} \quad \vee \quad (\epsilon = e \land \ell_{w_i} \in \{0, 3\} \land x = 0 \land \Psi' = \Psi_1[1/x]) \]

The environment executes a V-operation on x.

\[ \tau_{w_{j,0}} \quad \vee \quad \text{stut}_{w_{j}} \]

These transitions are illustrated in figure 3.4.

![Transitions of writer\(^1\)](image)

Figure 3.4: Transitions of writer\(^1\).

4. Liveness

L\(_{w_{j}}\) expresses that the P- and V-operations on the semaphore x are strongly fair and all the other transitions are weakly fair.
Let $WF_{w_j} \triangleq \{ \tau_{w_j, k} \mid k \in \{2, 5\} \}$ and $SF_{w_j} \triangleq \{ \tau_{w_j, k} \mid k \in \{1, 3, 4, 6, 7, 8\} \}$ then

$$L_{w_j} \triangleq \bigwedge_{\tau \in WF_{w_j}} (\Diamond \Box En(\tau) \rightarrow \Box \Diamond \tau) \land \bigwedge_{\tau \in SF_{w_j}} (\Box \Diamond En(\tau) \rightarrow \Diamond \tau)$$

### 3.3.3 Requirement $W_1$

The condition should express that the described deadlocked sequences don’t occur, i.e., it when $ar$ is increased by 1 then $aw = 0$ and when $aw$ is increased by 1 then $aw = 0$ and $ar = 0$. Formally:

$$W_1 \triangleq \Box \left( \left( \bigwedge_{i=1}^{N} \ell_{r_i} = 1 \rightarrow aw = 0 \right) \land \left( \bigwedge_{j=1}^{M} \ell_{w_j} = 1 \rightarrow (ar = 0 \land aw = 0) \right) \right)$$

This corresponds to the following machine: Let

$$p_{ri} \triangleq \ell_{r_i} = 1 \rightarrow aw = 0$$
$$p'_{ri} \triangleq \ell'_{r_i} = 1 \rightarrow aw' = 0$$
$$p_{w_j} \triangleq \ell_{w_j} = 1 \rightarrow (ar = 0 \land aw = 0)$$
$$p'_{w_j} \triangleq \ell'_{w_j} = 1 \rightarrow (ar' = 0 \land aw' = 0)$$

then $W_1$ is the conjunction of the machines $p_{ri} \land \Box((p_{ri} \land p'_{ri}) \lor stut)$ and $p_{wj} \land \Box((p_{wj} \land p'_{wj}) \lor stut)$ for $1 \leq i \leq N$ and $1 \leq j \leq M$.

### 3.3.4 $S_1$ relatively refines $S_0$

Since the semaphore $x$ and the shared variables $ar$ and $aw$ are used only by the subcomponents of $S_1$, we should prove $S_1 \mid \{x, ar, aw\}$ relatively refines $S_0$ instead of $S_1$ relatively refines $S_0$. According to definition 35, 36 and theorem 8 $S_1 \mid \{x, ar, aw\}$ relatively refines $S_0$ with respect to $(W_1, W_0)$ iff the following holds:

$$\mathfrak{D}(B_1) = \mathfrak{D}(B_0) \text{ and } \models (\exists X_1. (G_1 \land (e = e \Rightarrow (x, ar, aw)' = (x, ar, aw))) \rightarrow (\exists X_0. (G_0)))$$

where $X_1$ are the local variables from $S_1$, i.e., $X_1 \triangleq \{ \ell_{r_i} \mid i = 1, \ldots, N \} \cup \{ \ell_{w_j} \mid j = 1, \ldots, M \} \cup \{x, ar, aw\}$ and $G_1$ is the composition of $S_{r_1} (i = 1, \ldots, N)$ and $S_{w_1} (j = 1, \ldots, M)$ and $W_1$.

Let $\bar{\epsilon}_1 \triangleq \epsilon_1, \ldots, \epsilon_{1,N}, \epsilon_{1,N+1}, \ldots, \epsilon_{1,N+M}$, and

Let $B_{r_1}^A \triangleq B_{r_1}^1, \ldots, B_{r_1}^A, B_{w_1}^m, \ldots, B_{w_1}^{M}$

then $G_1 \triangleq$

$$\left( \exists \bar{\epsilon}_1. \bigcirc_{B_r^A}(\epsilon, \bar{\epsilon}_1) \land \bigwedge_{i=1}^{N} H_{r_i}[\epsilon_{1,i}/\epsilon] \land \bigwedge_{j=1}^{M} H_{w_j}[\epsilon_{1,N+j}/\epsilon] \right) \land W_1$$

$X_0$ are the local variables from $S_0$, i.e., $X_0 \triangleq \emptyset$ and $G_0$ is the composition of $S_{r_i} (i = 1, \ldots, N)$ and $S_{w_j} (j = 1, \ldots, M)$ and $W_0$,
3.3 The first development step

let $\overline{} \triangleq \epsilon_{1,1}, \ldots, \epsilon_{0,N}, \epsilon_{0,N+1}, \ldots, \epsilon_{0,N+M}$, and
let $B_{\overline{0}} \triangleq B_{\overline{0},1}, \ldots, B_{\overline{0},N}, B_{\overline{0},N+1}, \ldots, B_{\overline{0},N+M}$
then $G_{\overline{0}} \triangleq \left( \exists \overline{r}_{0} \in \mathcal{R}_{\overline{0}} \right) \wedge \bigwedge_{i=1}^{N} H_{i,0}^{\epsilon} \left[ \epsilon_{i,0} \right] \wedge \bigwedge_{j=1}^{M} H_{0,j}^{\epsilon} \left[ \epsilon_{0,j+1} \right] \wedge W_{0}$

Since $W_{0}$ can’t be decomposed into sub-requirements but doesn’t constrain the $\epsilon$ variables and $W_{1}$ can be decomposed into sub-requirements $W_{r_{i}} \triangleq \Box_{\overline{0}}(\ell_{r_{i}} = 1 \rightarrow aw = 0)$ for reader$i$ and $W_{w_{i}} \triangleq \Box_{\overline{0}}(\ell_{w_{i}} = 1 \rightarrow (aw = 0) \land ar = 0)$, and doesn’t constrain the $\epsilon$ variables
Lemma 9, 10 and 11 can be used for the proof, i.e., following proof rule can be used

\[ W_{1} \subseteq \bigcap_{i=1}^{N} W_{r_{i}} \cap \bigcap_{j=1}^{M} W_{w_{j}} \]

\[ S_{r_{i}} \rightarrow \text{ref } W_{r_{i}} \quad S_{w_{j}} \rightarrow \text{ref } W_{w_{j}} \quad W_{r_{i}} \text{ constraining } B_{r_{i}} \]

\[ S_{r_{i}} \rightarrow \text{ref } W_{r_{i}} \quad S_{w_{j}} \rightarrow \text{ref } W_{w_{j}} \quad W_{w_{j}} \text{ constraining } B_{w_{j}} \]

This means we have to prove for $i = 1, \ldots, N$ and $j = 1, \ldots, M$:

1. $(\exists \overline{r}_{1} \cdot (H_{r_{1}}^{\epsilon} \wedge W_{r_{1}})) \rightarrow H_{r_{0}}^{\epsilon} \wedge W_{0}$
2. $(\exists \overline{r}_{1} \cdot (H_{w_{1}}^{\epsilon} \wedge W_{w_{1}})) \rightarrow H_{w_{0}}^{\epsilon} \wedge W_{0}$
3. $W_{1} \rightarrow (W_{r_{1}} \wedge W_{w_{1}})$

ad (1) Rule 3 will be used to prove (1). This means one has to prove

(a) $S_{1} \cap \text{Hist}(W_{1}) \models (L_{1} \land p_{r_{i}}) \rightarrow L_{0}^{\epsilon} \land p$
(b) $S_{1} \cap \text{Hist}(W_{1}) \models T_{r_{1}} \land ((p_{r_{i}} \land p_{r_{i}'}) \lor \text{stut}_{r_{i}}) \rightarrow T_{r_{0}} \land ((p \land p') \lor \text{stut}_{0})$
(c) $S_{1} \cap \text{Hist}(W_{1}) \models L_{r_{1}}^{\epsilon}$

(a) Proof 1

\[ L_{r_{i}}^{\epsilon} \land p_{r_{i}} \]

% Def. $L_{r_{i}}, p_{r_{i}}$

\[ (x, ar, aw, s_{r_{i}}, \ell_{r_{i}}^{\epsilon}) = (1, 0, 0, 0, 0) \land (\ell_{r_{i}}^{\epsilon} \rightarrow aw = 0) \]

% \[ 0 \leq \#(j : 1 \leq j \leq M : s_{w_{j}} = 1) \leq aw \]

% \[ 0 \leq \#(i : 1 \leq j \leq M : s_{r_{i}} = 1) \leq ar \]

% \[ s_{r_{i}} = 0 \]

\[ \wedge (\#(j : 1 \leq j \leq M : s_{w_{j}} = 1) = 0 \land \#(i : 1 \leq j \leq M : s_{r_{i}} = 1) = 0) \]

% Def. $L_{0}^{\epsilon}, p$

\[ L_{0}^{\epsilon} \land p \]

(b) Proof 2

Since $T_{r_{i}}$ is of the form $\text{stut}_{r_{i}} \lor \forall \epsilon(\epsilon = a_{r} \land trans_{r})$ then $T_{r_{i}} \land ((p_{r_{i}} \land p_{r_{i}'}) \lor \text{stut}_{r_{i}})$ is equal to $\text{stut}_{r_{i}} \lor \forall \epsilon(\epsilon = a_{r} \land trans_{r} \land p_{r_{i}} \land p_{r_{i}'})$. $T_{r_{0}}$ is of the form $\text{stut}_{r_{0}} \lor \forall \epsilon(\epsilon = a_{r} \land trans_{r})$ so $T_{r_{0}} \land ((p \land p') \lor \text{stut}_{0})$ is equal to $\text{stut}_{r_{0}} \lor \forall \epsilon(\epsilon = a_{r} \land trans_{r} \land p \land p')$.  

57
\[-\tau_{\bar{i}2} \land p_{\bar{i}} \land p'_{\bar{i}}
= \left(\epsilon = i \land \ell_i = 1 \land p_{\bar{i}} \land \Psi'_i = \Psi_1 \left[0,1/x,\ell_i\right]\right)
\rightarrow \text{stut}_{p_{\bar{i}}}
\text{since } s_{\bar{i}} \text{ doesn't change.}\]

\[-\tau_{\bar{i},i} \land p_{\bar{i}} \land p'_{\bar{i}}
= \left(\epsilon = i \land \ell_i = 1 \land p_{\bar{i}} \land \Psi'_i = \Psi_1 \left[\ar+1,2/\ar,\ell_i\right]\right)
\rightarrow \text{stut}_{p_{\bar{i}}}
\text{since } s_{\bar{i}} \text{ doesn't change.}\]

\[-\tau_{\bar{i}4} \land p_{\bar{i}} \land p'_{\bar{i}}
= \left(\epsilon = i \land \ell_i = 3 \land \Psi'_i = \Psi_1 \left[0,4/x,\ell_i\right]\right)
\rightarrow \text{stut}_{p_{\bar{i}}}
\text{since } s_{\bar{i}} \text{ doesn't change.}\]

\[-\tau_{\bar{i}5} \land p_{\bar{i}} \land p'_{\bar{i}}
= \left(\epsilon = i \land \ell_i = 4 \land \Psi'_i = \Psi_1 \left[\ar-1,5/\ar,\ell_i\right]\right)
\rightarrow \text{stut}_{p_{\bar{i}}}
\text{since } s_{\bar{i}} \text{ doesn't change.}\]

\[-\tau_{\bar{i}6} \land p_{\bar{i}} \land p'_{\bar{i}}
= \left(\epsilon = i \land \ell_i = 5 \land \Psi'_i = \Psi_1 \left[0,1,0/s_{\bar{i}},\ell_i\right]\right)
\rightarrow \left(\epsilon = i \land s_{\bar{i}} = 1 \land p \land p' \land s'_{\bar{i}} = 0\right)
= \tau_{\bar{i},i} \land p \land p'
\text{because } 0 \leq \ell_i \leq M : s_{\bar{i}} = 1 \text{ and } 0 \leq \ell_i \leq N : s_{\bar{i}} = 1 \leq \ar.\]

\[-\tau_{\bar{i},i} \land p_{\bar{i}} \land p'_{\bar{i}}
= \left(\epsilon = i \land \ell_i = 1 \land p_{\bar{i}} \land \Psi'_i = \Psi_1 [0/\ell_i]\right)
\rightarrow \text{stut}_{p_{\bar{i}}}
\text{since } s_{\bar{i}} \text{ doesn't change.}\]

\[-\tau_{\bar{i},i} \land p_{\bar{i}} \land p'_{\bar{i}}
= \left(\epsilon = i \land \ell_i \in \{0,3\} \land \Psi'_i = \Psi_1 [1/x]\right)
\rightarrow \text{stut}_{p_{\bar{i}}}
\text{since } s_{\bar{i}} \text{ doesn't change.}\]

\[-\text{stut}_{p_{\bar{i}}} \rightarrow \text{stut}_{p_{\bar{i}}}
\text{since } s_{\bar{i}} \text{ doesn't change.}\]
3.4 The second development step

\[ L_{r,0} = (\phi \Box En(\tau_{i,1}) \rightarrow \Box \Box \tau_{i,1}) \land (\phi \Box En(\tau_{r,2}) \rightarrow \Box \Box \tau_{r,2}) \]

\[ = (\phi \Box (\epsilon = i \land s_{r} = 0) \rightarrow \Box (\epsilon = i \land s_{r} = 0 \land s'_{r} = 1)) \]

From the fact that (see Figure 3.3) transitions \( \tau_{i,1} \) and \( \tau_{r,2} \) are strongly fair and \( \tau_{i,2} \) is weakly fair follows that \( (\phi \Box En(\tau_{i,1}) \rightarrow \Box \Box \tau_{i,1}) \), i.e., \( \tau_{i,1} \) is weakly fair. From the fact that (see Figure 3.3) transitions \( \tau_{r,4} \) and \( \tau_{r,6} \) are strongly fair and \( \tau_{r,5} \) is weakly fair follows that \( (\phi \Box En(\tau_{r,2}) \rightarrow \Box \Box \tau_{r,2}) \), i.e., \( \tau_{r,2} \) is weakly fair.

ad (2) Analogue to the proof of (1).

ad (3) This is trivial because \( W_{1} \leftrightarrow (\land \omega_{i=1}^{N} W_{r,i} \land \land \omega_{j=1}^{M} W_{w,j}) \).

3.4 The second development step

As seen in section 3.3 the first implementation can generate deadlocked sequences. In this step we change the components of the first implementation in such a way that deadlock inside a PV-section is not possible anymore. This is the same as is done by Dijkstra: he massages reader\(^1\) and writer\(^1\) into reader\(^2\) and writer\(^2\) so that no deadlocked sequences inside a PV-section are generated any more.

One such deadlocked sequence generated by the first implementation is as follows: suppose reader\(^1\) has gained the access-right for the shared variables (first PV-segment) and suppose \( aw = 1 \) (a writer is executing WRITE). Then reader\(^1\) can never increase \( ar \) by 1, i.e., reader\(^1\) has deadlocked.

Dijkstra uses the split binary semaphore technique to prevent programs from becoming deadlocked inside a PV-section. The idea is that we must prevent programs from getting the access-right (get into a PV-section) for the shared variables if we know that they can not give it back (get deadlocked inside a PV-section). For reader\(^1\) this means: never let it enter the first PV-section if \( aw \) does not equal zero. For writer\(^1\) this means: never let it enter the first PV-section if \( aw \) or \( ar \) does not equal zero. Reader\(^2\) and writer\(^2\) never block in their second PV-section.

How does one prevent that reader\(^2\) gets deadlocked inside a PV-section? This is done as follows: reader\(^1\) chooses, when it gives the access-right back, who can have it thereafter. reader\(^1\) executes therefore the following piece of program as replacement for \( \psi(m) \):

\[ \text{CHOOSE: if true} \rightarrow V(m) \; \Box \; aw = 0 \rightarrow V(r) \; \Box \; aw = 0 \land ar = 0 \rightarrow V(w) \; \Box \]

We have to split semaphore \( mx \) in three pieces. If \( aw \) equals zero then a reader is allowed to enter its first PV-section, i.e., this PV-section is not guarded by \( P(mx) \) but by \( P(r) \). We do this substitution for all PV-sections of reader\(^1\) and writer\(^1\). So we have replaced \( mx \) by three other binary semaphores.

What is the initial value of these semaphores? If they all have initial value 1 then more than one program can have access-right to the shared variables, i.e., only one has initial
value 1. Semaphore r can not have initial value 1 because if no reader wants to execute READ then no writer can execute WRITE. The same holds for semaphore w. Thus m has initial value 1. But then no reader or writer can enter the first PV-section. The solution of this problem is that we insert a PV-section \((P(m); \text{CHOOSE})\) at front of the first one. This is in short what Dijkstra does to prevent that reader\(^1\) and writer\(^1\) get deadlocked inside a PV-section. The result of this transformation is:

\[
\text{reader}^2_i: \quad \text{do } \text{true } \rightarrow \text{NCS}; \\
\quad P(m); \text{CHOOSE}; \\
\quad P(r); ar:=ar+1; \text{CHOOSE}; \\
\quad \text{READ}; \\
\quad P(m); ar:=ar-1; \text{CHOOSE} \\
\text{od}
\]

\[
\text{writer}^2_j: \quad \text{do } \text{true } \rightarrow \text{NCS}; \\
\quad P(m); \text{CHOOSE}; \\
\quad P(w); aw:=aw+1; \text{CHOOSE}; \\
\quad \text{WRITE}; \\
\quad P(m); aw:=aw-1; \text{CHOOSE} \\
\text{od}
\]

\[\text{Syn}^2 : \big\| \bigoplus_{i=1}^N \text{reader}^2_i \big\| \bigoplus_{j=1}^M \text{writer}^2_j\]

\text{Syn}^2 generates no sequences that can deadlock inside a PV-section. But \text{Syn}^2 can generate sequences that can deadlock outside these sections, e.g. initially a reader\(^1\) can choose for a \(V(w)\) operation, and get blocked by a \(P(r)\) operation. Then no other reader or writer can enter the first PV-section because semaphore m equals zero.

In the following sections the DTL machine specifications \(S_x^2\) (corresponding to program reader\(^1\)) and \(S_y^2\) (corresponding to program writer\(^1\)), and the extra requirement \(W_2\), excluding computations that deadlock outside PV-sections, are given.

### 3.4.1 Specification \(S_x^2\)

The formal specification \(S_x^2 = (B_x^2, H_x^2)\) where \(H_x^2 = L_x^2 \land T_r^2 \land L_w^2\) and \(B_x^2, L_x^2, T_r^2\) and \(L_w^2\) are as follows:

1. **Basis** \(B_x^2 = ((\text{In}_x^2, \text{Out}_x^2), (V_x^2, X_x^2))\)

\[\begin{align*}
\text{In}_x^2 & \trianglerighteq \emptyset, \\
\text{Out}_x^2 & \trianglerighteq \emptyset, \\
V_x^2 & \trianglerighteq \{m, r, w, ar, aw, s_r\}, \\
X_x^2 & \trianglerighteq \{\ell_x^2\}
\end{align*}\]

- \(\ell_x^2 = 0\): reader\(^1\) is non critical.
### 3.4 The second development step

- $\ell_{r_2}^i = 6$: reader$^i_1$ has executed first P-action on $m$.
- $\ell_{r_2}^i = 7$: reader$^i_1$ has executed first CHOOSE.
- $\ell_{r_2}^i = 1$: reader$^i_1$ has executed P-action on $r$.
- $\ell_{r_2}^i = 2$: reader$^i_1$ has increased $ar$ by 1.
- $\ell_{r_2}^i = 3$: reader$^i_1$ is critical.
- $\ell_{r_2}^i = 4$: reader$^i_1$ has executed second P-action on $m$.
- $\ell_{r_2}^i = 5$: reader$^i_1$ has decreased $ar$ by 1.

Let $\Psi_2 \triangleq (m, r, w, ar, aw, s_r, \ell_{r_2})$ and $\Psi'_2 \triangleq (m', r', w', ar', aw', s'_r, \ell'_{r_2})$.

#### 2. Initial States:

$L_2 \triangleq \Psi_2 = (1, 0, 0, 0, 0, 0, 0)$

#### 3. Transitions:

Let $CHO(X) \triangleq \Psi'_2 = \Psi_2 \left[ 1, X / m, \ell_{r_2}^i \right]$

\[ \lor (aw = 0 \land \Psi'_2 = \Psi_2 \left[ 1, X / r, \ell_{r_2}^i \right]) \]

\[ \lor (aw = 0 \land ar = 0 \land \Psi'_2 = \Psi_2 \left[ 1, X / w, \ell_{r_2}^i \right]) \]

$T_{r_2} \triangleq$

\[ \tau_{r_2,1} \quad (\epsilon = i \land (\ell_{r_2}^i, m) = (0, 1) \land \Psi'_2 = \Psi_2 \left[ 0, 6 / m, \ell_{r_2}^i \right]) \]

Reader$^i_1$ executes its first P-action on $m$.

\[ \tau_{r_2,2} \quad \lor (\epsilon = i \land \ell_{r_2}^i = 6 \land CHO(7)) \]

Reader$^i_1$ executes the first CHOOSE.

\[ \tau_{r_2,3} \quad \lor (\epsilon = i \land (\ell_{r_2}^i, r) = (7, 1) \land \Psi'_2 = \Psi_2 \left[ 0, 1 / r, \ell_{r_2}^i \right]) \]

Reader$^i_1$ executes P-action on $r$.

\[ \tau_{r_2,4} \quad \lor (\epsilon = i \land \ell_{r_2}^i = 1 \land \Psi'_2 = \Psi_2 \left[ ar + 1, 2 / ar, \ell_{r_2}^i \right]) \]

Reader$^i_1$ increases the number of active readers by one.

\[ \tau_{r_2,5} \quad \lor (\epsilon = i \land \ell_{r_2}^i = 2 \land CHO(3)) \land \Psi'_2 = \Psi_2 \left[ 1 / s_r \right] \]

Reader$^i_1$ becomes critical.

\[ \tau_{r_2,6} \quad \lor (\epsilon = i \land (\ell_{r_2}^i, m) = (3, 1) \land \Psi'_2 = \Psi_2 \left[ 0, 4 / m, \ell_{r_2}^i \right]) \]

Reader$^i_1$ executes the second P-action on $m$.

\[ \tau_{r_2,7} \quad \lor (\epsilon = i \land \ell_{r_2}^i = 4 \land \Psi'_2 = \Psi_2 \left[ ar - 1, 5 / ar, \ell_{r_2}^i \right]) \]

Reader$^i_1$ decreases the number of active readers by one.

\[ \tau_{r_2,8} \quad \lor (\epsilon = i \land \ell_{r_2}^i = 5 \land CHO(0)) \land \Psi'_2 = \Psi_2 \left[ 0 / s_r \right] \]

Reader$^i_1$ becomes non critical.
\[ \tau^2_{1,9} \quad \forall \left( \epsilon = e \land m = 1 \land \Psi'_2 = \Psi_2[0/m] \right) \\
\text{The environment executes a P-operation on } m. \\
\tau^2_{1,10} \quad \forall \left( \epsilon = e \land r = 1 \land \Psi'_2 = \Psi_2[0/r] \right) \\
\text{The environment executes a P-operation on } r. \\
\tau^2_{1,11} \quad \forall \left( \epsilon = e \land w = 1 \land \Psi'_2 = \Psi_2[0/w] \right) \\
\text{The environment executes a P-operation on } w. \\
\tau^2_{1,12} \quad \forall \left( \epsilon = e \land \ell_{r,2} \in \{0,7,3\} \land m = 0 \land \Psi'_2 = \Psi_2[1/m] \right) \\
\text{The environment executes a V-operation on } m. \\
\tau^2_{1,13} \quad \forall \left( \epsilon = e \land \ell_{r,2} \in \{0,7,3\} \land r = 0 \land \Psi'_2 = \Psi_2[1/r] \right) \\
\text{The environment executes a V-operation on } r. \\
\tau^2_{1,14} \quad \forall \left( \epsilon = e \land \ell_{r,2} \in \{0,7,3\} \land w = 0 \land \Psi'_2 = \Psi_2[1/w] \right) \\
\text{The environment executes a V-operation on } w. \\
\tau^2_{1,15} \quad \forall \text{stut}_r \\
\text{These transitions are illustrated in figure 3.5} \\
\]

4. Liveness

L^2 expresses that the P- and V-operations on the semaphores m, r and w are strongly fair and all the other transitions are weakly fair.

Let \( WF_{r,2} \triangleq \{ \tau^2_{r,k} \mid k \in \{4,7\} \} \) and \\
\( SF_{r,2} \triangleq \{ \tau^2_{r,k} \mid k \in \{1,2,3,5,6,8,9,10,11,12,13,14\} \} \) then \\
\[ L^2 \triangleq \bigwedge_{\tau \in WF_{r,2}} (\Box \Box En(\tau) \rightarrow \Box \Box \tau) \land \bigwedge_{\tau \in SF_{r,2}} (\Box \Box En(\tau) \rightarrow \Box \Box \tau) \]

3.4.2 Specification \( S^{2}_{w^2} \)

The formal specification \( S^{2}_{w^2} \triangleq (B^{2}_{w^2}, H^{2}_{w^2}) \) where \( H^{2}_{w^2} \triangleq I^{2}_{w^2} \land \Box T^{2}_{w^2} \land L^{2}_{w^2} \) and \( B^{2}_{w^2}, I^{2}_{w^2}, T^{2}_{w^2} \) and \( L^{2}_{w^2} \) are as follows:

1. Basis \( B^{2}_{w^2} = (\langle In^{2}_{w^2}, Out^{2}_{w^2} \rangle, \langle V^{2}_{w^2}, X^{2}_{w^2} \rangle) \)

\[ \begin{align*}
In^{2}_{w^2} & \triangleq \emptyset, \\
Out^{2}_{w^2} & \triangleq \emptyset, \\
V^{2}_{w^2} & \triangleq \{m, r, w, ar, aw, s_{w^2}\}, \\
X^{2}_{w^2} & \triangleq \{\ell^{2}_{w^2}\}
\end{align*} \]

- \( \ell^{2}_{w^2} = 0 \): writer^2 is non critical.
- \( \ell^{2}_{w^2} = 6 \): writer^2 has executed first P-action on m.
3.4 The second development step

Let \( \Psi_i \triangleq (m, r, w, ar, aw, s_{w_j}, \ell_{w_j}^i) \) and \( \Psi'_i \triangleq (m', r', w', ar', aw', s'_{w_j}, \ell'_{w_j}^i) \).

2. Initial States:

\[ I_{w_j} \triangleq \Psi_2 = (1, 0, 0, 0, 0, 0) \]
3. Transitions:
Let $CHO(X) \triangleq \forall \Psi'_2 = \Psi_2 \left[ 1, X/m, \ell_{w_j}^2 \right]
\lor (aw = 0 \land \Psi'_2 = \Psi_2 \left[ 1, X/r, \ell_{w_j}^2 \right])
\lor (aw = 0 \land ar = 0 \land \Psi'_2 = \Psi_2 \left[ 1, X/w, \ell_{w_j}^2 \right])$

$$T_{w_j}^2 \triangleq \begin{cases}
\tau_{w_j,3}^2 & (\epsilon = i \land (\ell_{w_j}^2, m) = (0, 1) \land \Psi'_2 = \Psi_2 \left[ 0, 6/m, \ell_{w_j}^2 \right]) \\
\text{Writer}_j^3 \text{ executes its first P-action on } m. \\
\tau_{w_j,2}^2 & (\epsilon = i \land \ell_{w_j}^2 = 6 \land CHO(7)) \\
\text{Writer}_j^3 \text{ executes the first CHOOSE.} \\
\tau_{w_j,3}^2 & (\epsilon = i \land (\ell_{w_j}^2, w) = (7, 1) \land \Psi'_2 = \Psi_2 \left[ 0, 1/w, \ell_{w_j}^2 \right]) \\
\text{Writer}_j^3 \text{ executes P-action on } w. \\
\tau_{w_j,4}^2 & (\epsilon = i \land \ell_{w_j}^2 = 1 \land \Psi'_2 = \Psi_2 \left[ aw + 1, 2/aw, \ell_{w_j}^2 \right]) \\
\text{Writer}_j^3 \text{ increases the number of active writers by one.} \\
\tau_{w_j,5}^2 & (\epsilon = i \land \ell_{w_j}^2 = 2 \land CHO(3) \land \Psi'_2 = \Psi_2 \left[ 1/s_{w_j} \right]) \\
\text{Writer}_j^3 \text{ becomes critical.} \\
\tau_{w_j,6}^2 & (\epsilon = i \land (\ell_{w_j}^2, m) = (3, 1) \land \Psi'_2 = \Psi_2 \left[ 0, 4/m, \ell_{w_j}^2 \right]) \\
\text{Writer}_j^3 \text{ executes the second P-action on } m. \\
\tau_{w_j,7}^2 & (\epsilon = i \land \ell_{w_j}^2 = 4 \land \Psi'_2 = \Psi_2 \left[ aw - 1, 5/aw, \ell_{w_j}^2 \right]) \\
\text{Writer}_j^3 \text{ decreases the number of active writers by one.} \\
\tau_{w_j,8}^2 & (\epsilon = i \land \ell_{w_j}^2 = 5 \land CHO(0) \land \Psi'_2 = \Psi_2 \left[ 0/s_{w_j} \right]) \\
\text{Writer}_j^3 \text{ becomes non critical.} \\
\tau_{w_j,9}^2 & (\epsilon = e \land m = 1 \land \Psi'_2 = \Psi_2 \left[ 0/m \right]) \\
\text{The environment executes a P-operation on } m. \\
\tau_{w_j,10}^2 & (\epsilon = e \land r = 1 \land \Psi'_2 = \Psi_2 \left[ 0/r \right]) \\
\text{The environment executes a P-operation on } r. \\
\tau_{w_j,11}^2 & (\epsilon = e \land w = 1 \land \Psi'_2 = \Psi_2 \left[ 0/w \right]) \\
\text{The environment executes a P-operation on } w. \\
\tau_{w_j,12}^2 & (\epsilon = e \land \ell_{w_j}^2 \in \{0, 7, 3\} \land m = 0 \land \Psi'_2 = \Psi_2 \left[ 1/m \right]) \\
\text{The environment executes a V-operation on } m. \\
\tau_{w_j,13}^2 & (\epsilon = e \land \ell_{w_j}^2 \in \{0, 7, 3\} \land r = 0 \land \Psi'_2 = \Psi_2 \left[ 1/r \right]) \\
\text{The environment executes a V-operation on } r. \\
\tau_{w_j,14}^2 & (\epsilon = e \land \ell_{w_j}^2 \in \{0, 7, 3\} \land w = 0 \land \Psi'_2 = \Psi_2 \left[ 1/w \right]) \\
\text{The environment executes a V-operation on } w.
\end{cases}
3.4 The second development step

\[ \tau_{w^2,j} \cap \text{stut}_{w^2,j} \]

These transitions are illustrated in figure 3.6

![Diagram](image_url)

Figure 3.6: Transitions of writer^2_j.

4. Liveness:

\( L_{w^2_j} \) expresses that the P- and V-operations on the semaphores m, r and w are strongly fair and all the other transitions are weakly fair.

Let \( WF_{w^2_j} \triangleq \{ \tau_{w^2,j,k} \mid k \in \{4,7\} \} \) and

\[ SF_{w^2_j} \triangleq \{ \tau_{w^2,j,k} \mid k \in \{1,2,3,5,6,8,9,10,11,12,13,14\} \} \]

then

\[ L_{w^2_j} \triangleq \bigwedge_{\tau \in WF_{w^2_j}} (\Box \Diamond E_n(\tau) \rightarrow \Diamond \Box \tau) \wedge \bigwedge_{\tau \in SF_{w^2_j}} (\Box \Diamond E_n(\tau) \rightarrow \Diamond \Box \tau) \]

3.4.3 Requirement W_2

W_2 should express that reader^2_j and writer^2_j executes \texttt{CHOOSE} in such a way that no deadlocked computations are generated, i.e.,
• a $V(m)$ is executed if the number of readers and writers that are bound to execute a $P(m)$ is greater than zero,

• a $V(r)$ is executed if the number of readers that are bound to execute a $P(r)$ is greater than zero,

• a $V(w)$ is executed if the number of writers that are bound to execute a $P(w)$ is greater than zero.

Let $q$ be defined as

$\forall j (q : 1 \leq k \leq N : \ell_{i,2} \in \{0, 1, 2, 3, 4, 5\}) + \\
\forall j (n : 1 \leq n \leq M : \ell_{w,2} \in \{0, 1, 2, 3, 4, 5\}) > 0 \\
\forall (r = 1 \land aw = 0 \land (k : 1 \leq k \leq N : \ell_{i,2} \in \{6, 7\}) > 0) \\
\forall (w = 1 \land aw = 0 \land ar = 0 \land (n : 1 \leq n \leq M : \ell_{w,2} \in \{6, 7\}) > 0)$

Then $W_2$ is as follows

$W_2 \triangleq \Box \left( \bigwedge_{i=1}^{N} (\ell_{i,2} \in \{0, 7, 3\} \rightarrow q) \land (\bigwedge_{j=1}^{M} (\ell_{w,2} \in \{0, 7, 3\} \rightarrow q) \right)$

The same construction as in the previous development step is used to write this down as a machine. Let $p_2 \triangleq (\bigwedge_{i=1}^{N} \ell_{i,2} \in \{0, 7, 3\} \rightarrow q) \land (\bigwedge_{j=1}^{M} \ell_{w,2} \in \{0, 7, 3\} \rightarrow q)$ then $W_2 = p_2 \land \Box ((p_2 \land p_2') \lor \text{stut}_2)$. Again the liveness part of $W_2$ equals true.

**3.4.4 $S_2$ relatively refines $S_1$**

Since the semaphore $x$ and the shared variables $ar$ and $aw$ are used only by the sub-components of $S_2$ and the semaphores $m$, $w$ and $r$ and the shared variables $ar$ and $aw$ only by the subcomponents of $S_2$, we should prove $S_2 \models \{ m, w, r, ar, aw \}$ relatively refines $S_1 \models \{ x, ar, aw \}$. According to definition 35, 36 and theorem 8 $S_2 \models \{ m, r, w, ar, aw \}$ relatively refines $S_1 \models \{ x, ar, aw \}$ with respect to ($W_2$, $W_1$) iff the following holds:

$\mathcal{D}(B_2) = \mathcal{D}(B_1)$ and

$\models (\exists X_2 . (G_2 \land (\epsilon = e \Rightarrow (m, r, w, ar, aw)' = (m, r, w, ar, aw)))) \\
\rightarrow (\exists X_1 . (G_1 \land (\epsilon = e \Rightarrow (x, ar, aw)' = (x, ar, aw))))$

where $X_2$ are the local variables from $S_2$, i.e., $X_2 \triangleq \{ \ell_{i,2} \mid i = 1, \ldots, N \} \cup \{ \ell_{w,2} \mid j = 1, \ldots, M \} \cup \{ m, r, w, ar, ar \}$ and $G_2$ is the composition of $S_{\ell_2}$ ($i = 1, \ldots, N$) and $S_{w_2}$ ($j = 1, \ldots, M$) and $W_2$,

let $\tilde{e}_2 \triangleq \{ \epsilon_{1,1}, \ldots, \epsilon_{2,N}, \epsilon_{2,1+N}, \ldots, \epsilon_{2,N+M} \}$, and

let $\tilde{B}_2 \triangleq \{ B_{\ell_2}^{A}, B_{\ell_2}^{B}, B_{\ell_2}^{C}, B_{\ell_2}^{D}, \ldots, B_{\ell_2}^{M} \}$

then $G_2 \triangleq$

$(\exists \tilde{e}_2 . \circ \tilde{B}_2 (\epsilon, \tilde{e}_2) \land \bigwedge_{i=1}^{N} H_{\ell_2} [\epsilon_{i+2} \mid \epsilon] \land \bigwedge_{j=1}^{M} H_{w_2} [\epsilon_{2,N+j} \mid \epsilon]) \land W_2$
3.4 The second development step

$X_1$ are the local variables from $S_1$, i.e., $X_1 \triangleq \{ \ell_{t_1} \mid i = 1, \ldots, N \} \cup \{ \ell_{w_i} \mid j = 1, \ldots, M \} \cup \{ x, \mathbf{a}, \mathbf{aw} \}$ and $G_1$ is the composition of $S_{r_i} (i = 1, \ldots, N)$ and $S_{w_j} (j = 1, \ldots, M)$ and $W_1$.

let $\bar{\tau}_1 \triangleq \epsilon_{1,1}, \ldots, \epsilon_{1,N}, \epsilon_{1,N+1}, \ldots, \epsilon_{1,N+M}$, and

let $B^A \triangleq B^A_{r_1}, \ldots, B^A_{r_N}, B^A_{w_1}, \ldots, B^A_{w_M}$

then $G_1 \triangleq$

$$(\exists \bar{\tau}_1 \cdot \bar{B}^A (\tau, \bar{\tau}_1) \land \bigwedge_{i=1}^N H_{r_i} \downarrow [\epsilon_{1,i}/\epsilon] \land \bigwedge_{j=1}^M H_{w_j} \downarrow [\epsilon_{1,N+j}/\epsilon]) \land W_1$$

As seen in the previous development step $W_1$ is $\epsilon$-free and can be decomposed into sub-requirements $W_{r_i}$ and $W_{w_j}$ ($i = 1, \ldots, N$ and $j = 1, \ldots, M$). $W_2$ however can’t be decomposed into sub-requirements but it is $\epsilon$-free. Now Lemma 9, 10 and 11 can be used for the proof, i.e., following proof rule can be used

\[
\begin{align*}
&\bigotimes_{i=1}^N H_{r_i} \bigotimes_{j=1}^M H_{w_j} \cap W_2 \subseteq \\
&\bigotimes_{i=1}^N (H_{r_i} \cap W_2) \bigotimes_{j=1}^M (H_{w_j} \cap W_2) \\
&\bigcap_{i=1}^N W_{r_i} \cap \bigcap_{j=1}^M W_{w_j} \subseteq W_1 \quad W_{r_i} \text{ constraining } B_{r_i}^A \\
&S_{r_i} \text{ ref } W_{r_i} \text{ S}_1 \\
&S_{w_j} \text{ ref } W_{w_j} \text{ S}_1 \\
&S_2 \text{ ref } W \text{ S}_1
\end{align*}
\]

This means we have to prove for $i = 1, \ldots, N$ and $j = 1, \ldots, M$:

1. $$(\exists X_{r_2} \cdot (H_{r_2} \land W_2)) \rightarrow (\exists X_{r_1} \cdot (H_{r_1} \land W_{r_1}))$$
2. $$(\exists X_{w_2} \cdot (H_{w_2} \land W_2)) \rightarrow (\exists X_{w_1} \cdot (H_{w_1} \land W_{w_1}))$$
3. $$(W_{r_i} \land W_{w_j}) \rightarrow W_1$$
4. 

\[
(\exists \bar{\tau}_2 \cdot \bar{B}^A (\tau, \bar{\tau}_2) \land \bigwedge_{i=1}^N H_{r_i} \downarrow [\epsilon_{2,i}/\epsilon] \land \bigwedge_{j=1}^M H_{w_j} \downarrow [\epsilon_{2,N+j}/\epsilon]) \land W_2 \\
\rightarrow \\
(\exists \bar{\tau}_2 \cdot \bar{B}^A (\tau, \bar{\tau}_2) \land \bigwedge_{i=1}^N H_{r_i} \downarrow W_2 \land \bigwedge_{j=1}^M H_{w_j} \downarrow W_2) \downarrow [\epsilon_{2,i}/\epsilon] \land \bigwedge_{j=1}^M (H_{w_j} \downarrow W_2) \downarrow [\epsilon_{2,N+j}/\epsilon])
\]

ad (1) Rule 3 will be used to prove (1). This means one has to prove (a), (b) and (c) below, for $f$ the refinement mapping from $S_2$ to $S_1$, defined as: $f = f_x, f_{r_1}, f_{aw}, f_{aw}$ where $f_{r_1}$ is defined as

\[
\begin{align*}
&if \\
&\quad \ell_{r_2} = 6 \quad then \quad \ell_{r_1} - 6 \\
&\quad \ell_{r_2} = 7 \quad then \quad \ell_{r_1} - 7 \\
&\quad \ell_{r_2} \neq 6 \land \ell_{r_2} \neq 7 \quad then \quad \ell_{r_1} \\
&fi
\end{align*}
\]

and $f_x$ is defined as

\[
\begin{align*}
&if \\
&\quad \ell_{r_2} = 6 \quad then \quad m - 1 \\
&\quad \ell_{r_2} \neq 6 \quad then \quad m + r + w \\
&fi
\end{align*}
\]
, i.e., the first PV-section is stuttering and semaphore \( x \) is split into semaphores \( m, r \) and \( w \). Note: the refinement mappings for \( aw \) and \( ar \) are equal to the identity mapping, so we can leave them out.

\[(a) \quad S_2 \cap Hist(W_2) = (L_2 \wedge p_2) \rightarrow (L_1 \wedge p_{r1}) \left[ f/X_1 \right] \]

\[(b) \quad S_2 \cap Hist(W_2) = T_{r2} \wedge ((p_2 \wedge p'_{r2}) \vee \text{stut}_2) \rightarrow (T_{r1} \wedge ((p_{r1} \wedge p'_{r1}) \vee \text{stut}_{r1})) \left[ f/X_1 \right] \]

\[(c) \quad S_2 \cap Hist(W_2) = L_{ri} \left[ f/X_1 \right] \]

(a) **Proof 3**

\[\begin{align*}
I_{r2} \wedge p_2 &= \quad \text{Def. } I_{ri} \\
&= (m, r, w, ar, aw, s_{ri}, \ell_{r2}) = (1, 0, 0, 0, 0, 0) \\
\rightarrow &= \quad \text{Def. } f_x, f_{\ell_{ri}} \\
&= ((x, ar, aw, s_{ri}, \ell_{ri}) = (1, 0, 0, 0, 0) \wedge \ell_{ri} = 1 \rightarrow aw = 0) \left[ f/X_1 \right] \\
&= \quad \text{Def. } L_{ri}, p_{ri} \\
&= (L_{ri} \wedge p_{ri}) \left[ f/X_1 \right]
\end{align*}\]

(b) **Proof 4**

Since \( T_{r2} \) is of the form \( \text{stut}_{r2} \vee \bigvee \epsilon = a_r \wedge \text{trans}_r \) then \( T_{r2} \wedge ((p_2 \wedge p'_{r2}) \vee \text{stut}_2) \) is equal to \( \text{stut}_{r2} \vee \bigvee \epsilon = a_r \wedge \text{trans}_r \wedge p_2 \wedge p'_{r2} \). \( T_{r1} \) is of the form \( \text{stut}_{r1} \vee \bigvee \epsilon = a_r \wedge \text{trans}_r \) so \( T_{r1} \wedge ((p_{r1} \wedge p'_{r1}) \vee \text{stut}_{r1}) \) is equal to \( \text{stut}_{r1} \vee \bigvee \epsilon = a_r \wedge \text{trans}_r \wedge p_{r1} \wedge p'_{r1} \).

\[\begin{align*}
\tau_{r2} \wedge p_2 \wedge p'_{r2} &= \quad \epsilon = i \wedge \ell_{r2}, m = (0, 1) \wedge p_2 \wedge p'_{r2} \wedge \Psi_2 = \Psi_2 \left[ 0, 6/m, \ell_{r2} \right] \\
\rightarrow &= \quad \epsilon = i \wedge \Psi'_1 = \Psi_1 \\
&= \text{stut}_{r2} \left[ f/X_1 \right]
\end{align*}\]

The first P-operation of \( \text{reader}_2 \) is an stuttering step in \( \text{reader}_1 \).

\[\begin{align*}
\tau_{r2} \wedge p_2 \wedge p'_{r2} &= \quad \epsilon = i \wedge \ell_{r2} = 6 \wedge p_2 \wedge p'_{r2} \wedge \text{CHO}(7) \\
\rightarrow &= \quad \epsilon = i \wedge \Psi'_1 = \Psi_1 \\
&= \text{stut}_{r2} \left[ f/X_1 \right]
\end{align*}\]

The first V-operation of \( \text{reader}_2 \) is an stuttering step in \( \text{reader}_1 \).

\[\begin{align*}
\tau_{r2} \wedge p_2 \wedge p'_{r2} &= \quad \epsilon = i \wedge \ell_{x, r} = (7, 1) \wedge p_2 \wedge p'_{r2} \wedge \Psi' = \Psi_2 \left[ 0, 1/r, \ell_{x, r} \right] \\
\rightarrow &= \quad \epsilon = i \wedge \ell_{r1} = 0 \wedge x = 1 \wedge p_{ri} \wedge p'_{r1} \wedge \Psi_1 = \Psi_1 \left[ 0, 1/x, \ell_{r1} \right] \left[ f/X_1 \right] \\
&= \quad \tau_{r1} \wedge p_{ri} \wedge p'_{r1} \left[ f/X_1 \right]
\end{align*}\]

The second P-operation of \( \text{reader}_2 \) corresponds to the first P-operation of \( \text{reader}_1 \).
3.4 The second development step

\[ \tau_{2,4} \land p_2 \land p'_2 \]
\[ \Rightarrow (\epsilon = i \land \ell_{2,2} = 1 \land p_2 \land p'_2 \land \Psi'_2 = \Psi_2 \left[ \frac{ar + 1,2}{ar, \ell_{2,2}} \right]) \]
\[ \Rightarrow (\epsilon = i \land (\ell_{r,1}, aw) = (1,0) \land \rho_{r,1} \land p'_{r,1} \land \Psi'_1 = \Psi_1 \left[ \frac{ar + 1,2}{ar, \ell_{r,1}} \right]) \]
\[ \Rightarrow \left[ f/X_1 \right] \]
\[ = (\tau_{r,1,2} \land \rho_{r,1} \land p'_{r,1}) \left[ f/X_1 \right] \]

The ar increment step of reader\(^3\) corresponds to the ar increment step of reader\(^1\).

\[ \tau_{2,5} \land p_2 \land p'_2 \]
\[ \Rightarrow (\epsilon = i \land \ell_{2,2} = 2 \land p_2 \land p'_2 \land \Psi'_2 = \Psi_2 \left[ \frac{1}{s_r} \right]) \]
\[ \Rightarrow (\epsilon = i \land \ell_{r,1} = 2 \land \rho_{r,1} \land p'_{r,1} \land \Psi'_1 = \Psi_1 \left[ \frac{1,3}{s_r, x, \ell_{r,1}} \right]) \]
\[ \Rightarrow \left[ f/X_1 \right] \]
\[ = (\tau_{r,1,3} \land \rho_{r,1} \land p'_{r,1}) \left[ f/X_1 \right] \]

If reader\(^3\) becomes critical then reader\(^1\) becomes critical.

\[ \tau_{2,6} \land p_2 \land p'_2 \]
\[ \Rightarrow (\epsilon = i \land (\ell_{r,2}, m) = (3,1) \land p_2 \land p'_2 \land \Psi'_2 = \Psi_2 \left[ \frac{0,4}{m, \ell_{2,2}} \right]) \]
\[ \Rightarrow (\epsilon = i \land \ell_{r,1} = 3 \land x = 1 \land \rho_{r,1} \land p'_{r,1} \land \Psi'_1 = \Psi_1 \left[ \frac{0,4}{x, \ell_{r,1}} \right]) \]
\[ \Rightarrow \left[ f/X_1 \right] \]
\[ = (\tau_{r,1,4} \land \rho_{r,1} \land p'_{r,1}) \left[ f/X_1 \right] \]

The third P-operation of reader\(^3\) corresponds to the second P-operation of reader\(^1\).

\[ \tau_{2,7} \land p_2 \land p'_2 \]
\[ \Rightarrow (\epsilon = i \land \ell_{2,2} = 4 \land p_2 \land p'_2 \land \Psi'_2 = \Psi_2 \left[ \frac{0,5}{ar, \ell_{2,2}} \right]) \]
\[ \Rightarrow (\epsilon = i \land \ell_{r,1} = 4 \land \rho_{r,1} \land p'_{r,1} \land \Psi'_1 = \Psi_1 \left[ \frac{0,5}{ar, \ell_{r,1}} \right]) \]
\[ \Rightarrow \left[ f/X_1 \right] \]
\[ = (\tau_{r,1,5} \land \rho_{r,1} \land p'_{r,1}) \left[ f/X_1 \right] \]

The ar decrement step of reader\(^3\) corresponds to the ar decrement step of reader\(^1\).

\[ \tau_{2,8} \land p_2 \land p'_2 \]
\[ \Rightarrow (\epsilon = i \land \ell_{2,2} = 5 \land p_2 \land p'_2 \land \Psi'_2 = \Psi_2 \left[ \frac{0}{s_r} \right]) \]
\[ \Rightarrow (\epsilon = i \land \ell_{r,1} = 5 \land \rho_{r,1} \land p'_{r,1} \land \Psi'_1 = \Psi_1 \left[ \frac{0,1}{s_r, x, \ell_{r,1}} \right]) \]
\[ \Rightarrow \left[ f/X_1 \right] \]
\[ = (\tau_{r,1,6} \land \rho_{r,1} \land p'_{r,1}) \left[ f/X_1 \right] \]

If reader\(^3\) becomes non-critical then reader\(^1\) becomes non-critical.

\[ \tau_{2,9} \land p_2 \land p'_2 \]
\[ \Rightarrow (\epsilon = e \land m = 1 \land p_2 \land p'_2 \land \Psi'_2 = \Psi_2 \left[ \frac{0}{m} \right]) \]
\[ \Rightarrow (\epsilon = e \land x = 1 \land \rho_{r,1} \land p'_{r,1} \land \Psi'_1 = \Psi_1 \left[ \frac{0}{x} \right]) \]
\[ \Rightarrow \left[ f/X_1 \right] \]
\[ = (\tau_{r,1,7} \land \rho_{r,1} \land p'_{r,1}) \left[ f/X_1 \right] \]

If the environment of reader\(^3\) executes a P-operation then the environment of reader\(^1\) also executes a P-operation.
\[\tau_{i_1,0} \wedge p_1 \wedge p_2'\]
\[= (\epsilon = e \land r = 1 \land p_1 \land p_2' \land \Psi'_2 = \Psi_2 [0/r])\]
\[\Rightarrow (\epsilon = e \land x = 1 \land p_{r_i} \land p'_{r_i} \land \Psi'_1 = \Psi_1 [0/x]) \left[ f/X_1 \right]\]
\[= (\tau_{i,r} \land p_{r_i} \land p'_{r_i}) \left[ f/X_1 \right]\]

If the environment of reader\(^2\) executes a P-operation then the environment of reader\(^1\) also executes a P-operation.

\[\tau_{i_1,1} \wedge p_1 \wedge p_2'\]
\[= (\epsilon = e \land w = 1 \land p_1 \land p_2' \land \Psi'_2 = \Psi_2 [0/w])\]
\[\Rightarrow (\epsilon = e \land x = 1 \land p_{r_i} \land p'_{r_i} \land \Psi'_1 = \Psi_1 [0/x]) \left[ f/X_1 \right]\]
\[= (\tau_{i,r} \land p_{r_i} \land p'_{r_i}) \left[ f/X_1 \right]\]

If the environment of reader\(^2\) executes a P-operation then the environment of reader\(^1\) also executes a P-operation.

\[\tau_{i_1,2} \wedge p_1 \wedge p_2'\]
\[= (\epsilon = e \land \ell_{r_1} \in \{0, 7, 3\} \land m = 0 \land p_2 \land p'_2 \land \Psi'_2 = \Psi_2 [1/m])\]
\[\Rightarrow (\epsilon = e \land x = 0 \land p_{r_i} \land p'_{r_i} \land \Psi'_1 = \Psi_1 [1/x]) \left[ f/X_1 \right]\]
\[= (\tau_{i,r} \land p_{r_i} \land p'_{r_i}) \left[ f/X_1 \right]\]

If the environment of reader\(^2\) executes a V-operation then the environment of reader\(^1\) also executes a V-operation.

\[\tau_{i_1,3} \wedge p_1 \wedge p_2'\]
\[= (\epsilon = e \land \ell_{r_1} \in \{0, 7, 3\} \land r = 0 \land p_1 \land p'_1 \land \Psi'_1 = \Psi_1 [1/r])\]
\[\Rightarrow (\epsilon = e \land x = 0 \land p_{r_i} \land p'_{r_i} \land \Psi'_1 = \Psi_1 [1/x]) \left[ f/X_1 \right]\]
\[= (\tau_{i,r} \land p_{r_i} \land p'_{r_i}) \left[ f/X_1 \right]\]

If the environment of reader\(^2\) executes a V-operation then the environment of reader\(^1\) also executes a V-operation.

\[\tau_{i_1,4} \wedge p_1 \wedge p_2'\]
\[= (\epsilon = e \land \ell_{r_1} \in \{0, 7, 3\} \land w = 0 \land p_2 \land p'_2 \land \Psi'_2 = \Psi_2 [1/w])\]
\[\Rightarrow (\epsilon = e \land x = 0 \land p_{r_i} \land p'_{r_i} \land \Psi'_1 = \Psi_1 [1/x]) \left[ f/X_1 \right]\]
\[= (\tau_{i,r} \land p_{r_i} \land p'_{r_i}) \left[ f/X_1 \right]\]

If the environment of reader\(^2\) executes a V-operation then the environment of reader\(^1\) also executes a V-operation.

\[\text{stut}_{i_1} \Rightarrow \text{stut}_{i_1} \left[ f/X_1 \right]\]

since \(s_{r_i}\) doesn't change.

(c) Let \(WF_{i_{r_1}} \triangleq \{ \tau_{i_{r_1}, k} \mid k \in \{4, 7\} \}\) and
\(SF_{i_{r_1}} \triangleq \{ \tau_{i_{r_1}, k} \mid k \in \{1, 2, 3, 5, 6, 8, 9, 10, 11, 12, 13, 14\} \}\) then

\[L_{i_{r_1}} \triangleq \bigwedge_{\tau \in WF_{i_{r_1}}} (\Diamond \Box En(\tau) \rightarrow \Box \Diamond \tau) \land \bigwedge_{\tau \in SF_{i_{r_1}}} (\Box \Diamond En(\tau) \rightarrow \Box \Diamond \tau)\]

70
3.5 The third development step

Let \( WF_{r_1} = \{ \tau_{i,k} \mid k \in \{2, 5\} \} \) and \( SF_{r_1} = \{ \tau_{i,k} \mid k \in \{1, 3, 4, 6, 7, 8\} \} \) then

\[
L_{r_1} = \bigwedge_{\tau \in WF_{r_1}} (\square \diamond E_n(\tau) \rightarrow \square \diamond \tau) \land \bigwedge_{\tau \in SF_{r_1}} (\square \diamond E_n(\tau) \rightarrow \square \diamond \tau)
\]

The following holds:

\[
S_2 \cap Hist(W_2) \models L_{r_2} \rightarrow L_{r_1}, \left[ \bar{f}/X_1 \right]
\]

since \( \tau_{r_1,2} \) is relatively refined by \( \tau_{r_1,4}, \tau_{r_1,5} \) is relatively refined by \( \tau_{r_1,7} \), and \( \tau_{r_1,1} \) is relatively refined by \( \tau_{r_1,6}, \tau_{r_1,2} \) and \( \tau_{r_1,5} \), and \( \tau_{r_1,2} \) is relatively refined by \( \tau_{r_1,6}, \tau_{r_1,2} \) and \( \tau_{r_1,1} \), and \( \tau_{r_1,7} \) is relatively refined by \( \tau_{r_1,2}, \tau_{r_1,10} \) and \( \tau_{r_1,11} \), and \( \tau_{r_1,8} \) is relatively refined by \( \tau_{r_1,12}, \tau_{r_1,3} \) and \( \tau_{r_1,4} \). So

\[
S_2 \cap Hist(W_2) \models L_{r_1}, \left[ \bar{f}/X_1 \right]
\]

ad (2) Analogue to the proof of (1).

ad (3) This is trivial because \( W_1 \leftrightarrow (\wedge_{i=1}^{N} W_{r_1} \land \wedge_{j=1}^{M} W_{w_j}) \).

ad (4) The following holds because \( W_2 \) doesn’t contain \( \epsilon_j \) variables, i.e., it can be put within the existential quantification, and furthermore \( W_2 = W_2[\epsilon_{2,i}/\epsilon] = W_2[\epsilon_{2,N+j}/\epsilon] \) \((i = 1, \ldots, N \) and \( j = 1, \ldots, M) \) because \( W_2 \) doesn’t constrain the \( \epsilon \) variable.

\[
(\exists \bar{\epsilon}_2. \odot B_{2}^A(\epsilon, \bar{\epsilon}_2) \land \wedge_{i=1}^{N} H_{\bar{r}_1}[\epsilon_{2,i}/\epsilon] \land \wedge_{j=1}^{M} H_{w_j}[\epsilon_{2,N+j}/\epsilon]) \land W_2
\]

\[
\rightarrow (\exists \bar{\epsilon}_2. \odot B_{2}^A(\epsilon, \bar{\epsilon}_2) \land \wedge_{i=1}^{N} (H_{\bar{r}_1} \land W_2)[\epsilon_{2,i}/\epsilon] \land \wedge_{j=1}^{M} (H_{w_j} \land W_2)[\epsilon_{2,N+j}/\epsilon])
\]

3.5 The third development step

Dijkstra’s solution to the problem of the newly introduced deadlocked sequences is as follows: record in a shared variable \( bX \) the number of components that can generate a P-operation on a semaphore \( X \) as their first coming P-operation. A component that executed a P-operation on \( X \) decreases \( bX \) by one. The component “knows” what its next P-operation is, so it increases the corresponding shared variable by one. The guards in the \textsc{Choose} segment are changed so that the correct V-branch is chosen. The initial value of \( bm \) is \( N + M \) because initially all processes have \( P(m) \) as their first coming P-operation. The initial value of \( br \) and \( bw \) is then of course 0. Like in the second step the initial value of \( m \) is 1 and that of \( ar, aw, r \) and \( w \) 0. The result of this transformation is as follows:

\begin{verbatim}
reader^3:

do true -> NCS;
P(m):=bm-1;br:=br+1;CHOOSE;
P(r):=br-1;ar:=ar+1;bm:=bm+1;CHOOSE;
READ;
\end{verbatim}
Readers/Writers Example

\[ \begin{align*}
P(m); & \; bm := bm - 1; ar := ar - 1; bm := bm + 1; \text{CHOOSE} \\
\text{od}
\end{align*} \]

writer\textsuperscript{3}:

\[ \begin{align*}
d & \; \text{true} \rightarrow \text{NCS}; \\
\text{P}(m); & \; bm := bm - 1; bw := bw + 1; \text{CHOOSE}; \\
\text{P}(w); & \; bw := bw - 1; aw := aw + 1; bm := bm + 1; \text{CHOOSE}; \\
\text{WRITE}; \\
\text{P}(m); & \; bm := bm - 1; aw := aw - 1; bm := bm + 1; \text{CHOOSE} \\
\text{od}
\end{align*} \]

with \text{CHOOSE}: if bm > 0 \rightarrow \text{V}(m)

\[ \begin{align*}
aw = 0 & \land br > 0 \rightarrow \text{V}(r) \\
aw = 0 & \land ar = 0 \land bw > 0 \rightarrow \text{V}(w)
\end{align*} \]

Syn\textsuperscript{3} : \[ \|_{i=1}^{N} \text{reader}_{i}^{3} \| \|_{j=1}^{M} \text{writer}_{j}^{3} \]

Syn\textsuperscript{3} still generates sequences that Dijkstra does not allow. These sequences are generated because \text{CHOOSE} is still non-deterministic. Suppose a reader\textsuperscript{3} can choose between a \text{V}(m) and a \text{V}(r) operation. Choosing \text{V}(m) causes that another reader\textsuperscript{3} (writer\textsuperscript{3}) can signal that it has finished executing \text{READ} (\text{WRITE}) or wants to execute \text{READ} (\text{WRITE}). A \text{V}(r) causes that a reader\textsuperscript{3} can execute \text{READ}. Choosing \text{V}(m) thus unnecessarily blocks a reader\textsuperscript{3}. So it is not a deadlocked sequence but only an inefficient sequence. The informal requirement of Syn\textsuperscript{3} is that no unnecessary blocking sequences are allowed.

In the following sections the DTL machine specifications \( S_{r,3} \) (corresponding to program reader\textsuperscript{3}) and \( S_{w,3} \) (corresponding to program writer\textsuperscript{3}), and the extra requirement \( W_{3} \), excluding inefficient computations, are given.

3.5.1 Specification \( S_{r,3} \)

The formal specification \( S_{r,3} \) \( \triangleq (B_{r,3}, H_{r,3}) \) where \( H_{r,3} \) \( \triangleq I_{r,3} \land \square T_{r,3} \land L_{r,3} \) and \( B_{r,3}, I_{r,3}, T_{r,3} \) and \( L_{r,3} \) are as follows:

1. Basis \( B_{r,3} = (\langle \text{In}_{r,3}, \text{Out}_{r,3} \rangle, \langle V_{r,3}, X_{r,3} \rangle) \)

\[ \begin{align*}
\text{In}_{r,3} & \; \triangleq \; \emptyset, \\
\text{Out}_{r,3} & \; \triangleq \; \emptyset, \\
V_{r,3} & \; \triangleq \; \{ m, bm, r, br, bw, aw, ar, s_{r} \}, \\
X_{r,3} & \; \triangleq \; \{ \ell_{r,3} \}
\end{align*} \]

- \( \ell_{r,3} = 0 \): reader\textsuperscript{3} is non-critical.
- \( \ell_{r,3} = 6 \): reader\textsuperscript{3} executed first P-action on \text{m}.
3.5 The third development step

- $\ell_{r_{1}} = 8$: reader$_{1}^{r}$ updated bm and br.
- $\ell_{r_{1}} = 7$: reader$_{1}^{r}$ executed first CHOOSE.
- $\ell_{r_{1}} = 1$: reader$_{1}^{r}$ executed P-action on r.
- $\ell_{r_{1}} = 2$: reader$_{1}^{r}$ updated br, ar and bm.
- $\ell_{r_{1}} = 3$: reader$_{1}^{r}$ is critical.
- $\ell_{r_{1}} = 4$: reader$_{1}^{r}$ executed second P-action on m.
- $\ell_{r_{1}} = 5$: reader$_{1}^{r}$ updated ar.

Let $\Psi_{3} = (m, bm, r, br, w, bw, ar, aw, s_{r_{1}}, \ell_{r_{1}})$ and

$\Psi'_{3} = (m', bm', r', br', w', bw', ar', aw', s'_{r_{1}}, \ell'_{r_{1}})$.  

2. Initial States

$L_{r_{1}} = \Psi_{3} = (1, N + M, 0, 0, 0, 0, 0, 0, 0)$

3. Transitions:

Let $CHO(X) = \vee (bm > 0 \land \Psi'_{3} = \Psi_{3}[1, X / m, \ell_{r_{1}}]$

$\vee (aw = 0 \land br > 0 \land \Psi'_{3} = \Psi_{3}[1, X / r, \ell_{r_{1}}])$

$\vee (aw = 0 \land ar = 0 \land bw > 0 \land \Psi'_{3} = \Psi_{3}[1, X / w, \ell_{r_{1}}])$

$T_{r_{1}} = \{
\begin{align*}
\tau_{r_{1}, 1} & : \quad \epsilon = i \land (\ell_{r_{1}}, m) = (0, 1) \land \Psi'_{3} = \Psi_{3}[0, 6 / m, \ell_{r_{1}}]) \\
\text{Reader}_{1}^{r} & \text{executes first P-action on m.} \\
\tau_{r_{1}, 2} & : \quad \epsilon = i \land \ell_{r_{1}} = 6 \land \Psi'_{3} = \Psi_{3}[bm - 1, br + 1, 8 / bm, br, \ell_{r_{1}}]) \\
\text{Reader}_{1}^{r} & \text{updates bm and br.} \\
\tau_{r_{1}, 3} & : \quad \epsilon = i \land \ell_{r_{1}} = 8 \land CHO(7) \\
\text{Reader}_{1}^{r} & \text{executes first CHOOSE.} \\
\tau_{r_{1}, 4} & : \quad \epsilon = i \land (\ell_{r_{1}}, r) = (7, 1) \land \Psi'_{3} = \Psi_{3}[0, 1 / r, \ell_{r_{1}}]) \\
\text{Reader}_{1}^{r} & \text{executes its P-action on r.} \\
\tau_{r_{1}, 5} & : \quad \epsilon = i \land \ell_{r_{1}} = 1 \land \Psi'_{3} = \Psi_{3}[br - 1, ar + 1, bm + 1, 2 / br, ar, bm, \ell_{r_{1}}]) \\
\text{Reader}_{1}^{r} & \text{updates br, ar and bm.} \\
\tau_{r_{1}, 6} & : \quad \epsilon = i \land \ell_{r_{1}} = 2 \land CHO(3) \land \Psi'_{3} = \Psi_{3}[1 / s_{r_{1}}] \\
\text{Reader}_{1}^{r} & \text{becomes critical.} \\
\tau_{r_{1}, 7} & : \quad \epsilon = i \land (\ell_{r_{1}}, m) = (3, 1) \land \Psi'_{3} = \Psi_{3}[0, 4 / m, \ell_{r_{1}}]) \\
\text{Reader}_{1}^{r} & \text{executes second P-action on m.} \\
\tau_{r_{1}, 8} & : \quad \epsilon = i \land \ell_{r_{1}} = 4 \land \Psi'_{3} = \Psi_{3}[ar - 1, 5 / ar, \ell_{r_{1}}]) \\
\text{Reader}_{1}^{r} & \text{updates ar.} 
\end{align*}\}

73
\( \tau_{r_i,9} \lor (\epsilon = i \land \ell_{r_i} = 5 \land CHO(0) \land \Psi_3 = \Psi_3[0/s_i]) \)

Reader_3 becomes non critical.

\( \tau_{r_i,10} \lor (\epsilon = e \land m = 1 \land \Psi_3 = \Psi_3[0/m]) \)

The environment executes a P-operation on m.

\( \tau_{r_i,11} \lor (\epsilon = e \land r = 1 \land \Psi_3 = \Psi_3[0/r]) \)

The environment executes a P-operation on r.

\( \tau_{r_i,12} \lor (\epsilon = e \land w = 1 \land \Psi_3 = \Psi_3[0/w]) \)

The environment executes a P-operation on w.

\( \tau_{r_i,13} \lor (\epsilon = e \land \ell_{r_i} \in \{0, 7, 3\} \land m = 0 \land \Psi_3 = \Psi_3[1/m]) \)

The environment executes a V-operation on m.

\( \tau_{r_i,14} \lor (\epsilon = e \land \ell_{r_i} \in \{0, 7, 3\} \land r = 0 \land \Psi_3 = \Psi_3[1/r]) \)

The environment executes a V-operation on r.

\( \tau_{r_i,15} \lor (\epsilon = e \land \ell_{r_i} \in \{0, 7, 3\} \land w = 0 \land \Psi_3 = \Psi_3[1/w]) \)

The environment executes a P-operation on w.

\( \tau_{r_i,0} \lor \text{stut}_{r_i} \)

These transitions are illustrated in figure 3.7

4. Liveness:

\( L_{r_i} \) expresses that the P- and V-operations on the semaphores m, r and w are strongly fair and all the other transitions are weakly fair.

Let \( WF_{r_i} = \{ \tau_{r_i,k} \mid k \in \{2, 5, 8\} \} \) and

\( SF_{r_i} = \{ \tau_{r_i,k} \mid k \in \{1, 3, 4, 6, 7, 9, 10, 11, 12, 13, 14, 15\} \} \) then

\[
L_{r_i} \triangleq \bigwedge_{\tau \in WF_{r_i}} (\Diamond \square En(\tau) \rightarrow \square \Diamond \tau) \land \bigwedge_{\tau \in SF_{r_i}} (\Box \Diamond En(\tau) \rightarrow \Box \Diamond \tau)
\]

3.5.2 Specification \( S_{w_j} \)

The formal specification \( S_{w_j} \triangleq (B_{w_j}, H_{w_j}) \) where \( H_{w_j} \triangleq I_{w_j} \land \Box T_{w_j} \land L_{w_j} \) and \( B_{w_j}, I_{w_j}, T_{w_j} \) and \( L_{w_j} \) are as follows:

1. Basis \( B_{w_j} = (\langle \text{In}_{w_j}, \text{Out}_{w_j}, \text{V}_{w_j}, \text{X}_{w_j} \rangle) \)

\( \text{In}_{w_j} \triangleq \emptyset, \quad \text{Out}_{w_j} \triangleq \emptyset, \quad \text{V}_{w_j} \triangleq \{m, bm, br, w, bw, ar, aw, s_{w_j}\}, \quad \text{X}_{w_j} \triangleq \{\ell_{w_j}\} \)

74
3.5 The third development step

\[
\begin{align*}
(\ell^3_{r}, s_{r}, m) &= (0, 0, 0) \\
(\ell^3_{r}, s_{r}, m) &= (0, 0, 1) \\
(\ell^3_{r}, s_{r}, m) &= (8, 0, 0) \\
(\ell^3_{r}, s_{r}, r) &= (7, 0, 1) \\
(\ell^3_{r}, s_{r}, m) &= (5, 1, 0) \\
(\ell^3_{r}, s_{r}, m) &= (4, 1, 0) \\
(\ell^3_{r}, s_{r}, m) &= (3, 1, 1) \\
(\ell^3_{r}, s_{r}, r) &= (1, 0, 0) \\
(\ell^3_{r}, s_{r}, m) &= (3, 0, 0) \\
(\ell^3_{r}, s_{r}, r) &= (2, 0, 0)
\end{align*}
\]

Figure 3.7: Transitions of reader^3:

- \(\ell_{w_j}^3 = 0\): writer^3 is non-critical.
- \(\ell_{w_j}^3 = 6\): writer^3 executed first P-action on m.
- \(\ell_{w_j}^3 = 8\): writer^3 updated bm and bw.
- \(\ell_{w_j}^3 = 7\): writer^3 executed first CHOOSE.
- \(\ell_{w_j}^3 = 1\): writer^3 executed P-action on w.
- \(\ell_{w_j}^3 = 2\): writer^3 updated bw, aw and bm.
- \(\ell_{w_j}^3 = 3\): writer^3 is critical.
- \(\ell_{w_j}^3 = 4\): writer^3 executed second P-action on m.
- \(\ell_{w_j}^3 = 5\): writer^3 updated aw.

Let \(\Psi_3 \triangleq (m, bm, r, br, w, bw, ar, aw, s_{w_j}, \ell_{w_j}^3)\) and \(\Psi'_3 \triangleq (m', bm', r', br', w', bw', ar', aw', s'_{w_j}, \ell'_{w_j}^3)\).
2. Initial States

\[ I_{w_j}^3 \triangleq \Psi_3 = (1, N + M, 0, 0, 0, 0, 0, 0, 0) \]

3. Transitions:

Let \( CHO(X) \triangleq \lor (bm > 0 \land \Psi'_3 = \Psi_3[1, X/m, \ell_{w_j}^3]) \)

\[ \lor (aw = 0 \land br > 0 \land \Psi'_3 = \Psi_3[1, X/r, \ell_{w_j}^3]) \]

\[ \lor (aw = 0 \land ar = 0 \land bw > 0 \land \Psi'_3 = \Psi_3[1, X/w, \ell_{w_j}^3]) \]

\[ T_{w_j}^3 \triangleq \]

\[ \tau_{w_j,1}^3 \quad \left( \epsilon = i \land (\ell_{w_j}^3, m) = (0,1) \land \Psi'_3 = \Psi_3[0, 6/m, \ell_{w_j}^3] \right) \]

Writer\(^3\) executes first P-action on \(m\).

\[ \tau_{w_j,2}^3 \quad \lor \quad \left( \epsilon = i \land \ell_{w_j}^3 = 6 \land \Psi'_3 = \Psi_3[0, 1/bm + 1, 8/bm, bw, \ell_{w_j}^3] \right) \]

Writer\(^3\) updates \(bm\) and \(bw\).

\[ \tau_{w_j,2}^3 \quad \lor \quad \left( \epsilon = i \land \ell_{w_j}^3 = 8 \land CHO(7) \right) \]

Writer\(^3\) executes first \(CHOOSE\).

\[ \tau_{w_j,4}^3 \quad \lor \quad \left( \epsilon = i \land (\ell_{w_j}^3, w) = (7,1) \land \Psi'_3 = \Psi_3[0, 1/w, \ell_{w_j}^3] \right) \]

Writer\(^3\) executes its P-action on \(w\).

\[ \tau_{w_j,5}^3 \quad \lor \quad \left( \epsilon = i \land \ell_{w_j}^3 = 1 \land \Psi'_3 = \Psi_3[0, 1/bw + 1, bm + 1, 2/bw, aw, bm, \ell_{w_j}^3] \right) \]

Writer\(^3\) updates \(bw\), \(aw\) and \(bm\).

\[ \tau_{w_j,6}^3 \quad \lor \quad \left( \epsilon = i \land \ell_{w_j}^3 = 2 \land CHO(3) \land \Psi'_3 = \Psi_3[1/s_{w_j}] \right) \]

Writer\(^3\) becomes critical.

\[ \tau_{w_j,7}^3 \quad \lor \quad \left( \epsilon = i \land (\ell_{w_j}^3, m) = (3,1) \land \Psi'_3 = \Psi_3[0, 4/m, \ell_{w_j}^3] \right) \]

Writer\(^3\) executes second P-action on \(m\).

\[ \tau_{w_j,8}^3 \quad \lor \quad \left( \epsilon = i \land \ell_{w_j}^3 = 4 \land \Psi'_3 = \Psi_3[0, 5/aw, \ell_{w_j}^3] \right) \]

Writer\(^3\) updates \(aw\).

\[ \tau_{w_j,9}^3 \quad \lor \quad \left( \epsilon = i \land \ell_{w_j}^3 = 5 \land CHO(0) \land \Psi'_3 = \Psi_3[0/s_{w_j}] \right) \]

Writer\(^3\) becomes non critical.

\[ \tau_{w_j,10}^3 \quad \lor \quad \left( \epsilon = e \land m = 1 \land \Psi'_3 = \Psi_3[0/m] \right) \]

The environment executes a P-operation on \(m\).

\[ \tau_{w_j,11}^3 \quad \lor \quad \left( \epsilon = e \land r = 1 \land \Psi'_3 = \Psi_3[0/r] \right) \]

The environment executes a P-operation on \(r\).

\[ \tau_{w_j,12}^3 \quad \lor \quad \left( \epsilon = e \land w = 1 \land \Psi'_3 = \Psi_3[0/w] \right) \]

The environment executes a P-operation on \(w\).
3.5 The third development step

\[ \tau_{w_3}^{i,13} \land (e \land \ell_{w_3} = 0, 7, 3 \land m = 0 \land \Psi_3[1/m]) \]

The environment executes a V-operation on \( m \).

\[ \tau_{w_3}^{i,14} \land (e \land \ell_{w_3} = 0, 7, 3 \land r = 0 \land \Psi_3[1/r]) \]

The environment executes a V-operation on \( r \).

\[ \tau_{w_3}^{i,15} \land (e \land \ell_{w_3} = 0, 7, 3 \land w = 0 \land \Psi_3[1/w]) \]

The environment executes a P-operation on \( w \).

\[ \tau_{w_3}^{i,15} \land \text{stut}_{w_3} \]

These transitions are illustrated in figure 3.8

![Diagram](image)

Figure 3.8: Transitions of writer\(_3^3\).

4. Liveness:

\( \text{L}_{w_3} \) expresses that the P- and V-operations on the semaphores \( m, r \) and \( w \) are strongly fair and all the other transitions are weakly fair.

Let \( \text{WF}_{w_3} \triangleq \{ \tau_{w_3}^{i,k} \mid k \in \{3,5,8\} \} \) and
\[ \text{Readers/Writers Example} \]

\[ \text{SF}_w^\beta = \{ \tau_{w,j,k}^\beta \mid k \in \{1, 3, 4, 6, 7, 9, 10, 11, 12, 13, 14, 15\} \} \text{ then} \]

\[ \text{In}_w^\beta = \bigwedge_{\tau \in \text{WF}_{w_j}^\beta} (\Diamond \Box \mathit{En}(\tau) \rightarrow \Box \Diamond \tau) \land \bigwedge_{\tau \in \text{SF}_{w_j}^\beta} (\Box \Diamond \mathit{En}(\tau) \rightarrow \Diamond \Box \tau) \]

### 3.5.3 Requirement \( W_3 \)

The extra requirement \( W_3 \) should exclude inefficient computations caused by the nondeterminism of \texttt{CHOOSE}. So it is natural to make \texttt{CHOOSE} more deterministic, i.e., when one can choose between a \( \mathcal{V}(\mathbf{r}) \) (\( \mathcal{V}(\mathbf{w}) \)) and a \( \mathcal{V}(\mathbf{m}) \) operation priority is given to the \( \mathcal{V}(\mathbf{r}) \) (\( \mathcal{V}(\mathbf{w}) \)) operation. Let \( q_3 \) be defined as

\[
(m = 1 \land \mathbf{br} > 0) \land (aw = 0 \land \mathbf{br} > 0) \land (aw = 0 \land \mathbf{bw} > 0)
\]

Then \( W_3 \) is as follows

\[ W_3 = \bigwedge \left( \bigwedge_{i=1}^{N} \ell_{r,i}^\beta \in \{0, 7, 3\} \Rightarrow q_3 \right) \land \left( \bigwedge_{j=1}^{M} \ell_{w,j}^\beta \in \{0, 7, 3\} \Rightarrow q_3 \right) \]

So in \texttt{CHOOSE} priority is given to \( \mathcal{V}(\mathbf{r}) \) and \( \mathcal{V}(\mathbf{w}) \) by strengthen the guard of \( \mathcal{V}(\mathbf{m}) \) with the complement of the guards of \( \mathcal{V}(\mathbf{r}) \) and \( \mathcal{V}(\mathbf{w}) \). The same construction as in the previous development step is used to write this down as a machine. Let \( p_3 = \bigwedge \left( \bigwedge_{i=1}^{N} \ell_{r,i}^\beta \in \{0, 7, 3\} \Rightarrow q_3 \right) \land \left( \bigwedge_{j=1}^{M} \ell_{w,j}^\beta \in \{0, 7, 3\} \Rightarrow q_3 \right) \) then \( W_3 = p_3 \land \Box((p_3 \land p_3^0) \lor \text{stut}_3) \). Again the liveness part of \( W_3 \) equals \textit{true}.

### 3.5.4 \( S_3 \) relatively refines \( S_2 \)

Since the semaphores \( \mathbf{m}, \mathbf{r} \) and \( \mathbf{w} \), and the shared variables \( \mathbf{ar}, \mathbf{aw}, \mathbf{br}, \mathbf{bw} \) and \( \mathbf{bm} \) are used only by the subcomponents of \( S_3 \) and the semaphores \( \mathbf{m}, \mathbf{w} \) and \( \mathbf{r} \) and the shared variables \( \mathbf{ar} \) and \( \mathbf{aw} \) only by the subcomponents of \( S_2 \), we should prove \( S_3 \models \{} \mathbf{m}, \mathbf{w}, \mathbf{r}, \mathbf{ar}, \mathbf{aw}, \mathbf{br}, \mathbf{bw}, \mathbf{bm} \} \) relatively refines \( S_2 \models \{} \mathbf{m}, \mathbf{w}, \mathbf{r}, \mathbf{ar}, \mathbf{aw} \} \). According to definition 35, 36 and theorem 8 \( S_3 \models \{} \mathbf{m}, \mathbf{w}, \mathbf{r}, \mathbf{ar}, \mathbf{aw}, \mathbf{br}, \mathbf{bw}, \mathbf{bm} \} \) relatively refines \( S_2 \models \{} \mathbf{m}, \mathbf{w}, \mathbf{r}, \mathbf{ar}, \mathbf{aw} \} \) with respect to \( (W_3, W_2) \) iff the following holds:

\[
\begin{align*}
\mathcal{D}(B_3) &= \mathcal{D}(B_2) \\
\models (\exists X_3. (G_3 \land (\epsilon = e \Rightarrow (m, r, w, ar, aw, br, bw, bm))' = (m, r, w, ar, aw, br, bw, bm))) & \rightarrow \\
(\exists X_2. (G_2 \land (\epsilon = e \Rightarrow (m, r, w, ar, aw)' = (m, r, w, ar, aw))))
\end{align*}
\]

where \( X_3 \) are the local variables from \( S_3 \), i.e., \( X_3 = \{ \ell_{r,i}^\beta \mid i = 1, \ldots, N \} \cup \{ \ell_{w,j}^\beta \mid j = 1, \ldots, M \} \cup \{ m, r, w, ar, aw, br, bw, bm \} \) and \( G_3 \) is the composition of \( S_{r,i}^\beta (i = 1, \ldots, N) \) and \( S_{w,j}^\beta (j = 1, \ldots, M) \) and \( W_3 \),

let \( \bar{c}_3 = c_{3,1}, \ldots, c_{3,N}, c_{3,N+1}, \ldots, c_{3,N+M} \), and
3.5 The third development step

Let \( B_{ij}^A \triangleq B_{r_1}^A, \ldots, B_{r_N}^A, B_{w_1}^A, \ldots, B_{w_M}^A \).

Then \( G_3 \triangleq \)

\[
(\exists \bar{\epsilon}_3. \circ \bar{g}_3^A(\epsilon, \bar{\epsilon}_3) \land \bigwedge_{i=1}^N H_{r_i}^3[\epsilon_{3,i}/\epsilon] \land \bigwedge_{j=1}^M H_{w_j}^3[\epsilon_{3,N+j}/\epsilon]) \land W_3
\]

\( X_2 \) are the local variables from \( S_2 \), i.e., \( X_2 \triangleq \{ \ell_{r_i}^3 \mid i = 1, \ldots, N \} \cup \{ \ell_{w_j}^3 \mid j = 1, \ldots, M \} \cup \{ m, r, w, ar, aw \} \) and \( G_2 \) is the composition of \( S_{\ell_{r_i}^3} \) \( (i = 1, \ldots, N) \) and \( S_{\ell_{w_j}^3} \) \( (j = 1, \ldots, M) \) and \( W_2 \).

Let \( \bar{\epsilon}_2 \triangleq \epsilon_{2,1}, \ldots, \epsilon_{2,N}, \epsilon_{2,N+1}, \ldots, \epsilon_{2,N+M} \), and

let \( B_{ij}^A \triangleq B_{r_1}^A, \ldots, B_{r_N}^A, B_{w_1}^A, \ldots, B_{w_M}^A \).

Then \( G_2 \triangleq \)

\[
(\exists \bar{\epsilon}_2. \circ \bar{g}_2^A(\epsilon, \bar{\epsilon}_2) \land \bigwedge_{i=1}^N H_{r_i}^2[\epsilon_{2,i}/\epsilon] \land \bigwedge_{j=1}^M H_{w_j}^2[\epsilon_{2,N+j}/\epsilon]) \land W_2
\]

As seen in the previous development step \( W_2 \) is \( \epsilon \)-free but can not be decomposed into sub-requirements. \( W_3 \) is \( \epsilon \)-free and can be decomposed into sub-requirements. Let \( p_{r_i} \triangleq (\ell_{r_i}^3 \in \{ 0, 7, 3 \} \rightarrow q_3) \) and \( W_{r_i}^3 \triangleq \square p_{r_i} \) and \( p_{w_j} \triangleq (\ell_{w_j}^3 \in \{ 0, 7, 3 \} \rightarrow q_3) \) and \( W_{w_j}^3 \triangleq \square p_{w_j} \) then \( W_3 = (\bigcap_{i=1}^N W_{r_i}^3) \land (\bigcap_{j=1}^M W_{w_j}^3) \). Now Lemma 9, 10 and 11 can be used for the proof, i.e.,

following proof rule can be used

\[
\begin{align*}
W_3 & \subseteq \bigcap_{i=1}^N W_{r_i}^3 \land \bigcap_{j=1}^M W_{w_j}^3 \\
S_{r_i}^3 & \xrightarrow{W_3} \text{ref} W_2^6 S_{r_i}^2 \quad W_{r_i}^3 \text{ constraining } B_{r_i}^3 \\
S_{w_j}^3 & \xrightarrow{W_3} \text{ref} W_2^6 S_{w_j}^2 \quad W_{w_j}^3 \text{ constraining } B_{w_j}^3 \\
S_3 & \xrightarrow{W_3} \text{ref} W_2^6 S_2
\end{align*}
\]

This means we have to prove for \( i = 1, \ldots, N \) and \( j = 1, \ldots, M \):

\[
\begin{align*}
(1) & \quad \left( \exists X_{r_i}^3 . (H_{r_i}^2 \land W_{r_i}^3) \right) \Rightarrow \left( \exists X_{r_i}^2 . (H_{r_i}^2 \land W_2) \right) \\
(2) & \quad \left( \exists X_{w_j}^3 . (H_{w_j}^2 \land W_{w_j}^3) \right) \Rightarrow \left( \exists X_{w_j}^2 . (H_{w_j}^2 \land W_2) \right) \\
(3) & \quad W_3 \Rightarrow (W_{r_i}^3 \land W_{w_j}^3)
\end{align*}
\]

**ad (1) Rule 3** will be used to prove (1). This means one has to prove (a), (b) and (c) below, for \( \bar{f} \) the refinement mapping from \( S_3 \) to \( S_2 \), defined as: \( \bar{f} = f_{\ell_{r_i}^3}, f_m, f_r, f_w, f_{aw}, f_{ar} \) where \( f_{\ell_{r_i}^3} \) is defined as

\[
\begin{align*}
\text{if} & \quad \ell_{r_i}^3 = 8 \quad \text{then} \quad \ell_{r_i}^3 - 2 \\
\text{if} & \quad \ell_{r_i}^3 \neq 8 \quad \text{then} \quad \ell_{r_i}^3
\end{align*}
\]

, i.e., the updating of \( bm \) and \( br \) in the first PV-section in reader\(^3\) is a stuttering step in reader\(^2\). Note: The refinement mappings for \( m, r, w, aw \) and \( ar \) are equal to the
identity mapping, so we can leave them out.

(a) \( \mathcal{S}_3 \cap \text{Hist}(W_3) \models (L_3 \land p_{r_3}) \rightarrow (L_2 \land p_2) \begin{bmatrix} \bar{f} / X_2 \end{bmatrix} \)

(b) \( \mathcal{S}_3 \cap \text{Hist}(W_3) \models T_{r_3} \land ((p_{r_3} \land p'_{r_3}) \lor \text{stut}_{r_3}) \rightarrow (T_{r_2} \land ((p_2 \land p'_2) \lor \text{stut}_2)) \begin{bmatrix} \bar{f} / X_2 \end{bmatrix} \)

(c) \( \mathcal{S}_3 \cap \text{Hist}(W_3) \models L_{r_2} \begin{bmatrix} \bar{f} / X_2 \end{bmatrix} \)

(a) **Proof 5**

\[ L_3 \land p_{r_3} \]

\[ \rightarrow \]

\[ (m, bm, r, br, w, bw, ar, aw, s_{r_3}, \ell_{r_3}) = (1, N + M, 0, 0, 0, 0, 0, 0, 0) \]

\[ \rightarrow \]

\[ (L_2 \land p_2) \begin{bmatrix} \bar{f} / X_2 \end{bmatrix} \]

(b) **Proof 6**

Since \( T_{r_2} \) is of the form \( \text{stut}_{r_2} \lor \forall \sigma (\epsilon = a_r \land \text{trans}_r) \) then \( T_{r_2} \land ((p_2 \land p'_2) \lor \text{stut}_2) \) is equal to \( \text{stut}_{r_2} \lor \forall \sigma (\epsilon = a_r \land \text{trans}_r \land p_2 \land p'_2) \). \( T_{r_3} \) is of the form \( \text{stut}_{r_3} \lor \forall \sigma (\epsilon = a_r \land \text{trans}_r) \) so \( T_{r_3} \land ((p_{r_3} \land p'_{r_3}) \lor \text{stut}_{r_3}) \) is equal to \( \text{stut}_{r_3} \lor \forall \sigma (\epsilon = a_r \land \text{trans}_r \land p_{r_3} \land p'_{r_3}) \).

\[ \tau_{r_3} \land p_{r_3} \land p'_{r_3} \]

\[ \rightarrow \]

\[ (\epsilon = i \land (\ell_{r_3} \land m) = (0, 1) \land p_{r_3} \land p'_{r_3} \land \Psi_3 = \Psi_3 \begin{bmatrix} 0, 6/m, \ell_{r_3} \end{bmatrix}) \]

\[ \rightarrow \]

\[ (\epsilon = i \land (\ell_{r_3} \land m) = (0, 1) \land p_2 \land p'_2 \land \Psi_2 = \Psi_2 \begin{bmatrix} 0, 6/m, \ell_{r_2} \end{bmatrix}) \begin{bmatrix} \bar{f} / X_2 \end{bmatrix} \]

The first P-operation of reader\(^3\) corresponds with the first P-operation of reader\(^2\).

\[ \tau_{r_2} \land p_{r_2} \land p'_{r_2} \]

\[ \rightarrow \]

\[ (\epsilon = i \land \Psi'_2 = \Psi_2) \begin{bmatrix} \bar{f} / X_2 \end{bmatrix} \]

The updating of \( \text{br} \) and \( \text{bm} \) in reader\(^3\) is an stuttering step in reader\(^2\).

\[ \tau_{r_2} \land p_{r_2} \land p'_{r_2} \]

\[ \rightarrow \]

\[ (\epsilon = i \land \ell_{r_2} = 8 \land p_{r_2} \land p'_{r_2} \land \text{CHO}(7)) \]

\[ \rightarrow \]

\[ (\epsilon = i \land \ell_{r_2} = 6 \land p_2 \land p'_2 \land \text{CHO}(7)) \begin{bmatrix} \bar{f} / X_2 \end{bmatrix} \]

\[ \rightarrow \]

\[ (\tau_{r_2} \land p_2 \land p'_2) \begin{bmatrix} \bar{f} / X_2 \end{bmatrix} \]
3.5 The third development step

The first V-operation of reader$^3_1$ corresponds to the first V-operation of reader$^2_i$.

$$
\tau_{r,i}^3 \wedge p_{r_i} \wedge p_i^r = \\
(\epsilon = i \wedge (\ell_{r,i}^3, r) = (7, 1) \wedge p_{r_i} \wedge p_i^r \wedge \Psi_3 = \Psi_3 \left[0, 1/r, \ell_{r,i}^3\right]) \\
\tau_{r,i}^3 \wedge p_{r_i} \wedge p_i^r = \\
(\tau_{r,i}^2 \wedge p_i^r) [f/X_2] \\
The second P-operation of reader$^3_1$ corresponds to the second P-operation of reader$^2_i$.

$$
\tau_{r,i}^3 \wedge p_{r_i} \wedge p_i^r = \\
(\epsilon = i \wedge \ell_{r,i}^3 = 1 \wedge p_{r_i} \\
\wedge p_i^r \wedge \Psi_3^2 = \Psi_3 [br - 1, ar + 1, bm + 1, 2/br, ar, bm, \ell_{r,i}^3]) \\
\tau_{r,i}^3 \wedge p_{r_i} \wedge p_i^r = \\
(\tau_{r,i}^2 \wedge p_i^r) [f/X_2] \\
The ar decrement step of reader$^3_1$ corresponds to the ar decrement step of reader$^2_i$.

$$
\tau_{r,i}^3 \wedge p_{r_i} \wedge p_i^r = \\
(\epsilon = i \wedge \ell_{r,i}^3 = 2 \wedge p_{r_i} \wedge p_i^r \wedge CHO(3) \wedge \Psi_3 = \Psi_3 [1/s_r]) \\
\tau_{r,i}^3 \wedge p_{r_i} \wedge p_i^r = \\
(\tau_{r,i}^2 \wedge p_i^r) [f/X_2] \\
If reader$^2_i$ becomes critical then reader$^3_1$ becomes critical.

$$
\tau_{r,i}^3 \wedge p_{r_i} \wedge p_i^r = \\
(\epsilon = i \wedge (\ell_{r,i}^3, m) = (3, 1) \wedge p_{r_i} \wedge p_i^r \wedge \Psi_3^2 = \Psi_3 \left[0, 4/m, \ell_{r,i}^3\right]) \\
\tau_{r,i}^3 \wedge p_{r_i} \wedge p_i^r = \\
(\tau_{r,i}^2 \wedge p_i^r) [f/X_2] \\
The third P-operation of reader$^3_1$ corresponds to the third P-operation of reader$^2_i$.

$$
\tau_{r,i}^3 \wedge p_{r_i} \wedge p_i^r = \\
(\epsilon = i \wedge \ell_{r,i}^3 = 4 \wedge p_{r_i} \wedge p_i^r \wedge \Psi_3^2 = \Psi_3 \left[0, 4/m, \ell_{r,i}^3\right]) \\
\tau_{r,i}^3 \wedge p_{r_i} \wedge p_i^r = \\
(\tau_{r,i}^2 \wedge p_i^r) [f/X_2] \\
The ar decrement step of reader$^3_1$ corresponds to the ar decrement step of reader$^2_i$.

$$
\tau_{r,i}^3 \wedge p_{r_i} \wedge p_i^r = \\
(\epsilon = i \wedge \ell_{r,i}^3 = 5 \wedge p_{r_i} \wedge p_i^r \wedge CHO(0) \wedge \Psi_3 = \Psi_3 [0/s_r]) \\
\tau_{r,i}^3 \wedge p_{r_i} \wedge p_i^r = \\
(\tau_{r,i}^2 \wedge p_i^r) [f/X_2] \\
The third V-operation of reader$^3_1$ corresponds to the third V-operation of reader$^2_i$.
\[
\begin{align*}
\tau_{1,0}^{3} \land p_{r_{i}} \land p'_{r_{i}} &= \left( \epsilon = e \land m = 1 \land p_{r_{i}} \land p'_{r_{i}} \land \Psi_{3}^{3} = \Psi_{3}^{3}[0/m] \right) \\
&\rightarrow \left( \epsilon = e \land m = 1 \land p_{2} \land p'_{2} \land \Psi_{2}^{3} = \Psi_{2}^{3}[0/m] \right) \left[ f/X_{2} \right] \\
&= (\tau_{1,0}^{3} \land p_{2} \land p'_{2}) \left[ f/X_{2} \right] \\
\text{If the environment of reader}_{1}^{3} \text{ executes a P-operation then the environment of reader}_{2}^{3} \text{ also executes a P-operation.}
\end{align*}
\]

\[
\begin{align*}
\tau_{1,1}^{3} \land p_{r_{i}} \land p'_{r_{i}} &= \left( \epsilon = e \land r = 1 \land p_{r_{i}} \land p'_{r_{i}} \land \Psi_{3}^{3} = \Psi_{3}^{3}[0/r] \right) \\
&\rightarrow \left( \epsilon = e \land r = 1 \land p_{2} \land p'_{2} \land \Psi_{2}^{3} = \Psi_{2}^{3}[0/r] \right) \left[ f/X_{2} \right] \\
&= (\tau_{1,1}^{3} \land p_{2} \land p'_{2}) \left[ f/X_{2} \right] \\
\text{If the environment of reader}_{1}^{3} \text{ executes a P-operation then the environment of reader}_{2}^{3} \text{ also executes a P-operation.}
\end{align*}
\]

\[
\begin{align*}
\tau_{1,2}^{3} \land p_{r_{i}} \land p'_{r_{i}} &= \left( \epsilon = e \land w = 1 \land p_{r_{i}} \land p'_{r_{i}} \land \Psi_{3}^{3} = \Psi_{3}^{3}[0/w] \right) \\
&\rightarrow \left( \epsilon = e \land w = 1 \land p_{2} \land p'_{2} \land \Psi_{2}^{3} = \Psi_{2}^{3}[0/w] \right) \left[ f/X_{2} \right] \\
&= (\tau_{1,2}^{3} \land p_{2} \land p'_{2}) \left[ f/X_{2} \right] \\
\text{If the environment of reader}_{1}^{3} \text{ executes a P-operation then the environment of reader}_{2}^{3} \text{ also executes a P-operation.}
\end{align*}
\]

\[
\begin{align*}
\tau_{1,3}^{3} \land p_{r_{i}} \land p'_{r_{i}} &= \left( \epsilon = e \land \ell_{r_{i}}^{3} \in \{ 0, 7, 3 \} \land m = 0 \land p_{r_{i}} \land p'_{r_{i}} \land \Psi_{3}^{3} = \Psi_{3}^{3}[1/m] \right) \\
&\rightarrow \left( \epsilon = e \land \ell_{r_{i}}^{3} \in \{ 0, 7, 3 \} \land m = 0 \land p_{2} \land p'_{2} \land \Psi_{2}^{3} = \Psi_{2}^{3}[1/m] \right) \left[ f/X_{2} \right] \\
&= (\tau_{1,3}^{3} \land p_{2} \land p'_{2}) \left[ f/X_{2} \right] \\
\text{If the environment of reader}_{1}^{3} \text{ executes a V-operation then the environment of reader}_{2}^{3} \text{ also executes a V-operation.}
\end{align*}
\]

\[
\begin{align*}
\tau_{1,4}^{3} \land p_{r_{i}} \land p'_{r_{i}} &= \left( \epsilon = e \land \ell_{r_{i}}^{3} \in \{ 0, 7, 3 \} \land r = 0 \land p_{r_{i}} \land p'_{r_{i}} \land \Psi_{3}^{3} = \Psi_{3}^{3}[1/r] \right) \\
&\rightarrow \left( \epsilon = e \land \ell_{r_{i}}^{3} \in \{ 0, 7, 3 \} \land r = 0 \land p_{2} \land p'_{2} \land \Psi_{2}^{3} = \Psi_{2}^{3}[1/r] \right) \left[ f/X_{2} \right] \\
&= (\tau_{1,4}^{3} \land p_{2} \land p'_{2}) \left[ f/X_{2} \right] \\
\text{If the environment of reader}_{1}^{3} \text{ executes a V-operation then the environment of reader}_{2}^{3} \text{ also executes a V-operation.}
\end{align*}
\]

\[
\begin{align*}
\tau_{1,5}^{3} \land p_{r_{i}} \land p'_{r_{i}} &= \left( \epsilon = e \land \ell_{r_{i}}^{3} \in \{ 0, 7, 3 \} \land w = 0 \land p_{r_{i}} \land p'_{r_{i}} \land \Psi_{3}^{3} = \Psi_{3}^{3}[1/w] \right) \\
&\rightarrow \left( \epsilon = e \land \ell_{r_{i}}^{3} \in \{ 0, 7, 3 \} \land w = 0 \land p_{2} \land p'_{2} \land \Psi_{2}^{3} = \Psi_{2}^{3}[1/w] \right) \left[ f/X_{2} \right] \\
&= (\tau_{1,5}^{3} \land p_{2} \land p'_{2}) \left[ f/X_{2} \right] \\
\text{If the environment of reader}_{1}^{3} \text{ executes a V-operation then the environment of reader}_{2}^{3} \text{ also executes a V-operation.}
\end{align*}
\]

\[
\text{stut}_{1}^{3} \rightarrow \text{stut}_{2}^{3} \left[ f/X_{2} \right] \\
\text{since } s_{r_{i}} \text{ doesn't change.}
\]
3.6 The fourth development step

(c) Let $W_{r,2} \triangleq \{\tau_{r,4} \mid k \in \{4, 7\}\}$ and

$S_{r,2} \triangleq \{\tau_{r,4} \mid k \in \{1, 2, 3, 5, 6, 8, 9, 10, 11, 12, 13, 14\}\}$ then

$$L_{r,2} \triangleq \bigwedge_{\tau \in W_{r,2}} (\square E n(\tau) \rightarrow \square \Diamond \tau) \land \bigwedge_{\tau \in S_{r,2}} (\Box \Diamond E n(\tau) \rightarrow \Box \Diamond \tau)$$

Let $W_{r,3} \triangleq \{\tau_{r,5} \mid k \in \{2, 5, 8\}\}$ and

$S_{r,3} \triangleq \{\tau_{r,6} \mid k \in \{1, 3, 4, 6, 7, 9, 10, 11, 12, 13, 14, 15\}\}$ then

$$L_{r,3} \triangleq \bigwedge_{\tau \in W_{r,3}} (\square E n(\tau) \rightarrow \square \Diamond \tau) \land \bigwedge_{\tau \in S_{r,3}} (\Box \Diamond E n(\tau) \rightarrow \Box \Diamond \tau)$$

The following holds:

$$S_3 \cap Hist(W_3) \models L_{r,3} \rightarrow L_{r,2} \left[ f/X_2 \right]$$

since $\tau_{r,2}$ is relatively refined by $\tau_{r,1}$ and $\tau_{r,3}$, and $\tau_{r,2}$ is relatively refined by $\tau_{r,3}$, and $\tau_{r,2}$ is relatively refined by $\tau_{r,4}$, and $\tau_{r,2}$ is relatively refined by $\tau_{r,4}$, and $\tau_{r,2}$ is relatively refined by $\tau_{r,5}$, and $\tau_{r,2}$ is relatively refined by $\tau_{r,5}$, and $\tau_{r,2}$ is relatively refined by $\tau_{r,6}$, and $\tau_{r,2}$ is relatively refined by $\tau_{r,6}$, and $\tau_{r,2}$ is relatively refined by $\tau_{r,7}$, and $\tau_{r,2}$ is relatively refined by $\tau_{r,7}$, and $\tau_{r,2}$ is relatively refined by $\tau_{r,8}$, and $\tau_{r,2}$ is relatively refined by $\tau_{r,8}$, and $\tau_{r,2}$ is relatively refined by $\tau_{r,10}$, and $\tau_{r,2}$ is relatively refined by $\tau_{r,10}$, and $\tau_{r,2}$ is relatively refined by $\tau_{r,11}$, and $\tau_{r,2}$ is relatively refined by $\tau_{r,11}$, and $\tau_{r,2}$ is relatively refined by $\tau_{r,12}$, and $\tau_{r,2}$ is relatively refined by $\tau_{r,12}$, and $\tau_{r,2}$ is relatively refined by $\tau_{r,13}$, and $\tau_{r,2}$ is relatively refined by $\tau_{r,13}$, and $\tau_{r,2}$ is relatively refined by $\tau_{r,14}$, and $\tau_{r,2}$ is relatively refined by $\tau_{r,14}$, and $\tau_{r,2}$ is relatively refined by $\tau_{r,15}$.

So

$$S_3 \cap Hist(W_3) \models L_{r,2} \left[ f/X_2 \right]$$

ad (2) Analogue to the proof of (1).

ad (3) This is trivial because $W_3 \leftrightarrow (\wedge_{i=1}^{N} W_{r,i} \land \wedge_{j=1}^{M} W_{w,j}).$

3.6 The fourth development step

We have already seen how we can prevent reader$^3$ to choose wrongly between $V(r)$ and $V(w)$. Dijkstra also updates the PV-segments in such a way that only statements that are actually executed are listed. It turns out that we do not anymore need bm. Also the guards of CHOOSE get simpler. The result of this transformation is:

reader$^4$:

```
    do true → NCS;
    P(m); br:=br+1; if aw>0 → V(m) \ aw=0 → V(r) fi;
    P(r); br, ar:=br-1, ar+1;
    if br=0 → V(m) \ br>0 → V(r) fi;
    READ;
    P(m); ar:=ar-1;
```
if \( \text{ar}>0 \lor \text{bw}=0 \rightarrow V(m) \parallel \text{ar}=0 \land \text{bw}>0 \rightarrow V(w) \) fi

\text{od}

\text{writer}_j^4:\n\quad \text{do} \text{ true} \rightarrow \text{NCS};\n\quad P(m); \text{bw}:=\text{bw}+1;\n\quad \text{if} \ \text{aw}>0 \lor \text{ar}>0 \rightarrow V(m) \parallel \text{aw}=0 \land \text{ar}=0 \rightarrow V(w) \) fi;\n\quad P(w); \text{bw}, \text{aw}:=\text{bw}-1, \text{aw}+1; V(m);\n\quad \text{WRITE};\n\quad P(m); \text{aw}:=\text{aw}-1;\n\quad \text{if} \ \text{br}=0 \land \text{bw}=0 \rightarrow V(m) \parallel \text{br}>0 \rightarrow V(r) \parallel \text{bw}>0 \rightarrow V(w) \) fi
\text{od}

\text{Syn}^4 : \parallel \mathcal{N}_{i=1}^{\text{reader}_i^4} \parallel \mathcal{M}_{j=1}^{\text{writer}_j^4}

In the following sections the DTL machine specifications \( S_{\text{r}_4^i} \) (corresponding to program \( \text{reader}_i^4 \)) and \( S_{\text{w}_4^j} \) (corresponding to program \( \text{writer}_j^4 \)) are given. It should be clear that the extra requirement \( W_4 \) is equal to \text{true} because no further requirements are imposed on \( S_4 \).

### 3.6.1 Specification \( S_{\text{r}_4^i} \)

The formal specification \( S_{\text{r}_4^i} \triangleq (B_{\text{r}_4^i}, H_{\text{r}_4^i}) \) where \( H_{\text{r}_4^i} \triangleq L_{\text{r}_4^i} \land [T_{\text{r}_4^i} \land L_{\text{r}_4^i} \land B_{\text{r}_4^i}, L_{\text{r}_4^i}, T_{\text{r}_4^i} \land L_{\text{r}_4^i} \) are as follows:

1. **Basis** \( B_{\text{r}_4^i} = ((\text{In}_{\text{r}_4^i}, \text{Out}_{\text{r}_4^i}), (V_{\text{r}_4^i}, X_{\text{r}_4^i})) \)

\[
\begin{align*}
\text{In}_{\text{r}_4^i} & \triangleq \emptyset, \\
\text{Out}_{\text{r}_4^i} & \triangleq \emptyset, \\
V_{\text{r}_4^i} & \triangleq \{ \text{m, r, w, br, bw, ar, aw, s}_{\text{r}_4^i} \}, \\
X_{\text{r}_4^i} & \triangleq \{ \ell_{\text{r}_4^i} \}
\end{align*}
\]

- \( \ell_{\text{r}_4^i} = 0 \): \( \text{reader}_i^4 \) is non critical.
- \( \ell_{\text{r}_4^i} = 6 \): \( \text{reader}_i^4 \) executes first P-action on \( \text{m} \).
- \( \ell_{\text{r}_4^i} = 7 \): \( \text{reader}_i^4 \) has updated \( \text{br} \).
- \( \ell_{\text{r}_4^i} = 8 \): \( \text{reader}_i^4 \) has left first PV-section.
- \( \ell_{\text{r}_4^i} = 1 \): \( \text{reader}_i^4 \) has executed P-action on \( \text{r} \).
- \( \ell_{\text{r}_4^i} = 2 \): \( \text{reader}_i^4 \) has updated \( \text{br} \) and \( \text{ar} \).
- \( \ell_{\text{r}_4^i} = 3 \): \( \text{reader}_i^4 \) is critical.
- \( \ell_{\text{r}_4^i} = 4 \): \( \text{reader}_i^4 \) has executed second P-action \( \text{m} \).
- \( \ell_{\text{r}_4^i} = 5 \): \( \text{reader}_i^4 \) has updated \( \text{ar} \).
3.6 The fourth development step

Let $\Psi_4 \triangleq (m, r, br, bw, ar, aw, s_r, r_s, \ell)$ and
$\Psi'_4 \triangleq (m', r', br', bw', ar', aw', s'_r, r'_s)$.

2. Initial States:

$L_4 \triangleq \Psi_4 = (1, 0, 0, 0, 0, 0, 0)$

3. Transitions:

Let $CHO1(X) \triangleq \forall (aw > 0 \land \Psi'_4 = \Psi_4[1, X/m, \ell_4])$
$\lor (aw = 0 \land \Psi'_4 = \Psi_4[1, X/r, \ell_4])$

Let $CHO2(X) \triangleq \forall (br = 0 \land \Psi'_4 = \Psi_4[1, X/m, \ell_4])$
$\lor (br > 0 \land \Psi'_4 = \Psi_4[1, X/r, \ell_4])$

Let $CHO3(X) \triangleq \forall ((ar > 0 \lor bw = 0) \land \Psi'_4 = \Psi_4[1, X/m, \ell_4])$
$\lor (ar = 0 \land bw > 0 \land \Psi'_4 = \Psi_4[1, X/w, \ell_4])$

$T_{r,4} \triangleq$

$\\tau_{r,1} \iff \left( \epsilon = i \land (\ell_r, m) = (0, 1) \land \Psi'_4 = \Psi_4[0, 6/m, \ell_4] \right)$
Reader$_4$ executes its first P-action on m.

$\\tau_{r,2} \lor \left( \epsilon = i \land \ell_r = 6 \land \Psi'_4 = \Psi_4[br + 1, 8/br, \ell_4] \right)$
Reader$_4$ updates br.

$\\tau_{r,3} \lor \left( \epsilon = i \land \ell_r = 8 \land CHO1(7) \right)$
Reader$_4$ leaves the first PV-section.

$\\tau_{r,4} \lor \left( \epsilon = i \land (\ell_r, r) = (7, 1) \land \Psi'_4 = \Psi_4[0, 1/r, \ell_4] \right)$
Reader$_4$ executes its P-action on r.

$\\tau_{r,5} \lor \left( \epsilon = i \land \ell_r = 1 \land \Psi'_4 = \Psi_4[br - 1, ar + 1, 2/br, ar, \ell_4] \right)$
Reader$_4$ updates br and ar.

$\\tau_{r,6} \lor \left( \epsilon = i \land \ell_r = 2 \land CHO2(3) \land \Psi'_4 = \Psi_4[1/s_r] \right)$
Reader$_4$ becomes critical.

$\\tau_{r,7} \lor \left( \epsilon = i \land (\ell_r, m) = (3, 1) \land \Psi'_4 = \Psi_4[0, 4/m, \ell_4] \right)$
Reader$_4$ executes its second P-action on m.

$\\tau_{r,8} \lor \left( \epsilon = i \land \ell_r = 4 \land \Psi'_4 = \Psi_4[ar - 1, 5/ar, \ell_4] \right)$
Reader$_4$ updates ar.

$\\tau_{r,9} \lor \left( \epsilon = i \land \ell_r = 5 \land CHO3(0) \land \Psi'_4 = \Psi_4[0/s_r] \right)$
Reader$_4$ becomes non critical.

$\\tau_{r,10} \lor \left( \epsilon = e \land m = 1 \land \Psi'_4 = \Psi_4[0/m] \right)$
The environment executes a P-operation on m.
These transitions are illustrated in figure 3.9.

Figure 3.9: Transitions of reader\(^4\).

4. **Liveness:**

\( L_{\ell^4} \) expresses that the P- and V-operations on the semaphores \( m, r \) and \( w \) are
3.6 The fourth development step

strongly fair and all the other transitions are weakly fair.

Let \( \text{WF}_{r_i}^4 \overset{\Delta}{=} \{ \tau_{r_i,k} \mid k \in \{2, 5, 8\} \} \) and

\[ \text{SF}_{r_i}^4 \overset{\Delta}{=} \{ \tau_{r_i,k} \mid k \in \{1, 3, 4, 6, 7, 9, 10, 11, 12, 13, 14, 15\} \} \]

then

\[ L_{r_i}^4 \overset{\Delta}{=} \bigwedge_{\tau \in \text{WF}_{r_i}^4} (\square \square \text{En}(\tau) \rightarrow \square \square \tau) \wedge \bigwedge_{\tau \in \text{SF}_{r_i}^4} (\square \square \text{En}(\tau) \rightarrow \square \square \tau) \]

3.6.2 Specification \( S_{w_j}^4 \)

The formal specification \( S_{w_j}^4 \overset{\Delta}{=} (B_{w_j}^4, H_{w_j}^4) \) where \( H_{w_j}^4 \overset{\Delta}{=} I_{w_j}^4 \wedge \square T_{w_j}^4 \wedge I_{w_j}^4 \) and \( B_{w_j}^4, I_{w_j}^4, T_{w_j}^4 \) and \( L_{w_j}^4 \) as follows:

1. **Basis** \( B_{w_j}^4 = ((I_{w_j}^4, Out_{w_j}^4), (V_{w_j}^4, X_{w_j}^4)) \)

\[
\begin{align*}
I_{w_j}^4 &\overset{\Delta}{=} \emptyset, \\
Out_{w_j}^4 &\overset{\Delta}{=} \emptyset, \\
V_{w_j}^4 &\overset{\Delta}{=} \{ m, r, br, bw, aw, aw, s_{w_j} \}, \\
X_{w_j}^4 &\overset{\Delta}{=} \{ \ell_{w_j}^4 \}
\end{align*}
\]

- \( \ell_{w_j}^4 = 0 \): writer\(^4\(_{w_j} \) is non critical.
- \( \ell_{w_j}^4 = 6 \): writer\(^4\(_{w_j} \) executes first P-action on \( m \).
- \( \ell_{w_j}^4 = 8 \): writer\(^4\(_{w_j} \) has updated \( bw \).
- \( \ell_{w_j}^4 = 7 \): writer\(^4\(_{w_j} \) has left first PV-section.
- \( \ell_{w_j}^4 = 1 \): writer\(^4\(_{w_j} \) has executed P-action on \( w \).
- \( \ell_{w_j}^4 = 2 \): writer\(^4\(_{w_j} \) has updated \( bw \) and \( aw \).
- \( \ell_{w_j}^4 = 3 \): writer\(^4\(_{w_j} \) is critical.
- \( \ell_{w_j}^4 = 4 \): writer\(^4\(_{w_j} \) has executed second P-action \( m \).
- \( \ell_{w_j}^4 = 5 \): writer\(^4\(_{w_j} \) has updated \( aw \).

Let \( \Psi_4 \overset{\Delta}{=} (m, r, br, w, bw, aw, aw, s_{w_j}, \ell_{w_j}^4) \) and

\( \Psi'_4 \overset{\Delta}{=} (m', r', br', w', bw', aw', aw', s'_{w_j}, \ell'_4) \).

2. **Initial States:**

\[ I_{w_j}^4 \overset{\Delta}{=} \Psi_4 = (1, 0, 0, 0, 0, 0, 0, 0) \]
3. Transitions:

Let $CHO1(X) \triangleq \forall \ (aw > 0 \lor ar > 0) \land \Psi_4 = \Psi_4[1, X/m, \ell_{w^4}]
\lor \ (aw = 0 \land ar = 0 \land \Psi_4 = \Psi_4[1, X/w, \ell_{w^4}])$

Let $CHO2(X) \triangleq \Psi_4 = \Psi_4[1, X/m, \ell_{w^4}]
\lor \ (br = 0 \land bw = 0 \land \Psi_4 = \Psi_4[1, X/m, \ell_{w^4}])$

Let $CHO3(X) \triangleq \forall \ (br > 0 \land \Psi_4 = \Psi_4[1, X/r, \ell_{w^4}])
\lor \ (bw > 0 \land \Psi_4 = \Psi_4[1, X/w, \ell_{w^4}])$

$T_{w^4_j} \triangleq$

$\tau_{w^4_{j3}} \quad \left( \epsilon = i \land (\ell_{w^4_j}, m) = (0, 1) \land \Psi_4 = \Psi_4[0, 6/m, \ell_{w^4_j}] \right)$

Writer $j$ executes its first P-action on m.

$\tau_{w^4_{j2}} \lor \ (\epsilon = i \land \ell_{w^4_j} = 6 \land \Psi_4 = \Psi_4[bw + 1, 8/bw, \ell_{w^4_j}])$

Writer $j$ updates bw.

$\tau_{w^4_{j3}} \lor \ (\epsilon = i \land \ell_{w^4_j} = 8 \land CHO1(7))$

Writer $j$ leaves the first PV-section.

$\tau_{w^4_{j4}} \lor \ (\epsilon = i \land (\ell_{w^4_j}, r) = (7, 1) \land \Psi_4 = \Psi_4[0, 1/w, \ell_{w^4_j}])$

Writer $j$ executes its P-action on w.

$\tau_{w^4_{j5}} \lor \ (\epsilon = i \land \ell_{w^4_j} = 1 \land \Psi_4 = \Psi_4[bw - 1, aw + 1, 2/bw, aw, \ell_{w^4_j}])$

Writer $j$ updates bw and aw.

$\tau_{w^4_{j6}} \lor \ (\epsilon = i \land \ell_{w^4_j} = 2 \land CHO2(3) \land \Psi_4 = \Psi_4[1/s_{w^4_j}])$

Writer $j$ becomes critical.

$\tau_{w^4_{j7}} \lor \ (\epsilon = i \land (\ell_{w^4_j}, m) = (3, 1) \land \Psi_4 = \Psi_4[0, 4/m, \ell_{w^4_j}])$

Writer $j$ executes its second P-action on m.

$\tau_{w^4_{j8}} \lor \ (\epsilon = i \land \ell_{w^4_j} = 4 \land \Psi_4 = \Psi_4[aw - 1, 5/aw, \ell_{w^4_j}])$

Writer $j$ updates aw.

$\tau_{w^4_{j9}} \lor \ (\epsilon = i \land \ell_{w^4_j} = 5 \land CHO3(0) \land \Psi_4 = \Psi_4[0/s_{w^4_j}])$

Writer $j$ becomes non critical.

$\tau_{w^4_{j10}} \lor \ (\epsilon = e \land m = 1 \land \Psi_4 = \Psi_4[0/m])$

The environment executes a P-operation on m.

$\tau_{w^4_{j11}} \lor \ (\epsilon = e \land r = 1 \land \Psi_4 = \Psi_4[0/r])$

The environment executes a P-operation on r.

$\tau_{w^4_{j12}} \lor \ (\epsilon = e \land w = 1 \land \Psi_4 = \Psi_4[0/w])$

The environment executes a P-operation on w.
3.6 The fourth development step

$$\tau_{w_j^4}^{13} \left\{ \epsilon \in \ell_{w_j^4} \in \{0, 7, 3\} \land m = 0 \land \Psi_4 = \Psi_4[1/m] \right\}$$
The environment executes a V-operation on m.

$$\tau_{w_j^4}^{14} \left\{ \epsilon \in \ell_{w_j^4} \in \{0, 7, 3\} \land r = 0 \land \Psi_4 = \Psi_4[1/r] \right\}$$
The environment executes a V-operation on r.

$$\tau_{w_j^4}^{15} \left\{ \epsilon \in \ell_{w_j^4} \in \{0, 7, 3\} \land w = 0 \land \Psi_4 = \Psi_4[1/w] \right\}$$
The environment executes a V-operation on w.

$$\tau_{w_j^4}^{10} \left\{ \text{stut}_{w_j^4} \right\}$$

These transitions are illustrated in figure 3.10

![Diagram of transitions](image)

Figure 3.10: Transitions of writer\(^4\).

4. Liveness:
L\(_4^w\) expresses that the P- and V-operations on the semaphores m, r and w are strongly fair and all the other transitions are weakly fair.
Let WF\(_{w_j^4}\) $\triangleq \{ \tau_{w_j^4}^{k} \mid k \in \{2, 5, 8\} \}$ and

89
\[ SF_{w_j^A} \triangleq \{ \tau_{w_j^A} \mid k \in \{1, 3, 4, 6, 7, 9, 10, 11, 12, 13, 14, 15\} \} \text{ then} \]
\[ L_{w_j^A} \triangleq \bigwedge_{\tau \in WF_{w_j^A}} (\Box \Diamond En(\tau) \rightarrow \Box \Diamond \tau) \land \bigwedge_{\tau \in SF_{w_j^A}} (\Box \Diamond En(\tau) \rightarrow \Box \Diamond \tau) \]

### 3.6.3 \(S_4\) relatively refines \(S_3\)

Since the semaphores \(m, r\) and \(w\), and the shared variables \(ar, aw, br, bw\) and \(bm\) are used only by the subcomponents of \(S_3\) and the semaphores \(m, w\) and \(r\) and the shared variables \(ar, aw, br\) and \(bw\) only by the subcomponents of \(S_4\), we should prove \(S_4 \models \{m, w, r, ar, aw, br, bw\}\) relatively refines \(S_3 \models \{m, w, r, ar, aw, bm, br, bw\}\). According to definition 35, 36 and theorem 8 \(S_4 \models \{m, w, r, ar, aw, br, bw\}\) relatively refines \(S_3 \models \{m, w, r, ar, aw, br, bw, bm\}\) with respect to \((W_4, W_3)\) iff the following holds:

\[\begin{align*}
&\exists X_3 :\left(\forall \gamma : (G_3 \land (\epsilon = e \Rightarrow (m, r, w, ar, aw, br, bw)' = (m, r, w, ar, aw, br, bw)))\right) \\
&\rightarrow (\exists X_3 : (G_3 \land (\epsilon = e \Rightarrow (m, r, w, ar, aw, br, bw, bm)' = (m, r, w, ar, aw, br, bw, bm))))
\end{align*}\]

where \(X_3\) are the local variables from \(S_3\), i.e., \(X_3 \triangleq \{\ell_{r_i} \mid i = 1, \ldots, N\} \cup \{\ell_{w_j} \mid j = 1, \ldots, M\} \cup \{m, r, w, ar, aw, br, bw\}\) and \(G_3\) is the composition of \(S_{r_i} (i = 1, \ldots, N)\) and \(S_{w_j} (j = 1, \ldots, M)\).

Let \(\bar{e}_4 \triangleq \{\epsilon_{4,1}, \ldots, \epsilon_{4,N+N+1}, \ldots, \epsilon_{4,N+M}\}\) and

\[
\begin{align*}
&\big(\exists \bar{e}_4 : \big( \circ B_4^A \land L_{i=1}^N H_{r_i^A} [\epsilon_{4,i}/\epsilon] \land L_{j=1}^M H_{w_j^A} [\epsilon_{4,N+j}/\epsilon] \big) \big)
\end{align*}
\]

\(X_3\) are the local variables from \(S_3\), i.e., \(X_3 \triangleq \{\ell_{r_i} \mid i = 1, \ldots, N\} \cup \{\ell_{w_j} \mid j = 1, \ldots, M\} \cup \{m, r, w, ar, aw, br, bm, bw\}\) and \(G_3\) is the composition of \(S_{r_i} (i = 1, \ldots, N)\) and \(S_{w_j} (j = 1, \ldots, M)\) and \(W_3\). Let \(e_3 \triangleq \{\epsilon_{3,1}, \ldots, \epsilon_{3,N}, \epsilon_{3,N+1}, \ldots, \epsilon_{3,N+M}\}\) and

\[
\begin{align*}
&\big(\exists \bar{e}_3 : \big( \circ B_3^A \land L_{i=1}^N H_{r_i^A} [\epsilon_{3,i}/\epsilon] \land L_{j=1}^M H_{w_j^A} [\epsilon_{3,N+j}/\epsilon] \big) \big) \land W_3
\end{align*}
\]

As seen in the previous development step \(W_3\) is \(\epsilon\)-free and can be decomposed into sub-requirements. Let \(p_{ri} \triangleq (\ell_{r_i} \in \{0, 7, 3\} \rightarrow q_3)\) and \(W_{r_i} \triangleq \Box p_{ri}\), and \(p_{w_j} \triangleq (\ell_{w_j} \in \{0, 7, 3\} \rightarrow q_3)\) and \(W_{w_j} \triangleq \Box p_{w_j}\) then \(W_3 = (L_{i=1}^N W_{r_i}) \land (L_{j=1}^M W_{w_j})\). Now Lemma 9, 10 and 11 can be used for the proof, i.e., following proof rule can be used

\[
\begin{align*}
&\forall i \in \{0, 7, 3\} \rightarrow q_3) \land W_{w_j} \triangleq (L_{i=1}^N W_{r_i}) \land (L_{j=1}^M W_{w_j})\end{align*}
\]

Now Lemma 9, 10 and 11 can be used for the proof, i.e., following proof rule can be used

\[
\begin{align*}
&\forall i \in \{0, 7, 3\} \rightarrow q_3) \land W_{w_j} \triangleq (L_{i=1}^N W_{r_i}) \land (L_{j=1}^M W_{w_j})\end{align*}
\]
3.6 The fourth development step

This means we have to prove for \(i = 1, \ldots, N\) and \(j = 1, \ldots, M\):

1. \((\exists x_{\tau^4}.(H_{\tau^4})) \to (\exists x_{\tau^5}.(H_{\tau^5} \land W_{\tau^5}))\)
2. \((\exists x_{\omega^3}.(H_{\omega^3})) \to (\exists x_{\omega^3}.(H_{\omega^3} \land W_{\omega^3}))\)
3. \((W_{\tau^5} \land W_{\omega^3}) \to W_3\)

\textbf{ad (1)} Rule 3 will be used to prove (1). This means one has to prove (a), (b) and (c) below, for \(\bar{f}\) the refinement mapping from \(\mathcal{S}_4\) to \(\mathcal{S}_3\), defined as:

\[
\bar{f} = f_{\tau^4}, f_{m}, f_r, f_w, f_{ar}, f_{br}, f_{bw}, f_{bm} \text{ where } f_{bm} \text{ is defined as }
\]

\[
N + M - \text{br} - \text{bw}
\]

, i.e., \(\text{bm}\) can be expressed in terms of \(\text{br}\) and \(\text{bw}\). Note: the refinement mappings for \(\text{itr}4\), \(\text{r}, \text{w}, \text{aw}, \text{ar}, \text{br}\) and \(\text{bw}\) are equal to the identity mapping, so we can leave them out.

(a) \textbf{Proof 7}

\[
\begin{align*}
\mathcal{S}_4 &\models L_{\tau^4} \to (L_{\tau^5} \land p_{ri}) \left[\bar{f}/X_3\right] \\
\mathcal{S}_3 &\models T_{\tau^4} \to (T_{\tau^5} \land ((p_{ri} \land p'_{ri}) \lor \text{stut}_{\tau^3})) \left[\bar{f}/X_3\right] \\
\mathcal{S}_3 &\models L_{\tau^5} \left[\bar{f}/X_3\right]
\end{align*}
\]

(b) \textbf{Proof 8}

\(T_{\tau^3}\) is of the form \(\text{stut}_{\tau^3} \lor \vee \chi (\epsilon = a_r \land \text{trans}_{r})\) so \(T_{\tau^3} \land ((p_{ri} \land p'_{ri}) \lor \text{stut}_{\tau^3})\) is equal to \(\text{stut}_{\tau^3} \lor \vee \chi (\epsilon = a_r \land \text{trans}_{r} \land p_{ri} \land p'_{ri})\).

\[
\begin{align*}
\tau_{\tau^3} &= \left(\epsilon = i \land (\ell_{\tau^4}, \text{m}) = (0, 1) \land \Psi_4 = \Psi_4[0, 6/m, \ell_{\tau^4}]\right) \\
&\to \left(\epsilon = i \land (\ell_{\tau^3}, \text{m}) = (0, 1) \land p_{ri} \land p'_{ri} \land \Psi_3 = \Psi_3[0, 6/m, \ell_{\tau^3}]\right) \left[\bar{f}/X_3\right] \\
&= \left(\tau_{\tau^3} \land p_{ri} \land p'_{ri}\right) \left[\bar{f}/X_3\right]
\end{align*}
\]

The first \(P\)-operation of \(\text{reader}^4_{\tau}\) corresponds to the first \(P\)-operation of \(\text{reader}^4_{\tau}\).
The updating of reader\textsuperscript{1} corresponds to the updating of br and bm reader\textsuperscript{3}.

The first V-operation of reader\textsuperscript{4} corresponds to the first V-operation of reader\textsuperscript{3}.

The second P-operation of reader\textsuperscript{4} corresponds to the second P-operation of reader\textsuperscript{3}.

The ar decrement step of reader\textsuperscript{4} corresponds to the ar decrement step of reader\textsuperscript{3}.

If reader\textsuperscript{4} becomes critical then reader\textsuperscript{3} becomes critical.

The third P-operation of reader\textsuperscript{4} corresponds to the third P-operation of reader\textsuperscript{3}.
3.6 The fourth development step

\[ \tau_{i,4}^4 \]
\[ = (\epsilon = i \land \ell_{i,4} = 4 \land \Psi_4 = \Psi_4 \left[ ar - 1, 5/ar, \ell_{i,4} \right]) \]
\[ \rightarrow (\epsilon = i \land \ell_{i,4} = 4 \land p_{ri} \land p_{ri} \land \Psi_3 = \Psi_3 \left[ ar - 1, 5/ar, \ell_{i,4} \right]) \left[ f/X_3 \right] \]
\[ = (\tau_{i,8}^4 \land p_{ri} \land p_{ri} \land \Psi_3) \left[ f/X_3 \right] \]
The ar decrement step of reader\(^4_i\) corresponds to the ar decrement step of reader\(^3_i\).

\[ \tau_{i,4}^5 \]
\[ = (\epsilon = i \land \ell_{i,4} = 5 \land CHO3(0) \land \Psi_4 = \Psi_4 \left[ 0/s_{ri} \right]) \]
\[ \rightarrow (\epsilon = i \land \ell_{i,4} = 5 \land p_{ri} \land p_{ri} \land CHO(0) \land \Psi_3 = \Psi_3 \left[ 0/s_{ri} \right]) \left[ f/X_3 \right] \]
\[ = (\tau_{i,10}^4 \land p_{ri} \land p_{ri} \land \Psi_3) \left[ f/X_3 \right] \]
The third V-operation of reader\(^4_i\) corresponds to the third V-operation of reader\(^3_i\).

\[ \tau_{i,4}^6 \]
\[ = (\epsilon = e \land m = 1 \land \Psi_4 = \Psi_4 \left[ 0/m \right]) \]
\[ \rightarrow (\epsilon = e \land m = 1 \land p_{ri} \land p_{ri} \land \Psi_3 = \Psi_3 \left[ 0/m \right]) \left[ f/X_3 \right] \]
\[ = (\tau_{i,10}^4 \land p_{ri} \land p_{ri} \land \Psi_3) \left[ f/X_3 \right] \]
If the environment of reader\(^4_i\) executes a P-operation then the environment of reader\(^3_i\) also executes a P-operation.

\[ \tau_{i,4}^7 \]
\[ = (\epsilon = e \land r = 1 \land \Psi_4 = \Psi_4 \left[ 0/r \right]) \]
\[ \rightarrow (\epsilon = e \land r = 1 \land p_{ri} \land p_{ri} \land \Psi_3 = \Psi_3 \left[ 0/r \right]) \left[ f/X_3 \right] \]
\[ = (\tau_{i,11}^4 \land p_{ri} \land p_{ri} \land \Psi_3) \left[ f/X_3 \right] \]
If the environment of reader\(^4_i\) executes a P-operation then the environment of reader\(^3_i\) also executes a P-operation.

\[ \tau_{i,4}^8 \]
\[ = (\epsilon = e \land w = 1 \land \Psi_4 = \Psi_4 \left[ 0/w \right]) \]
\[ \rightarrow (\epsilon = e \land w = 1 \land p_{ri} \land p_{ri} \land \Psi_3 = \Psi_3 \left[ 0/w \right]) \left[ f/X_3 \right] \]
\[ = (\tau_{i,12}^4 \land p_{ri} \land p_{ri} \land \Psi_3) \left[ f/X_3 \right] \]
If the environment of reader\(^4_i\) executes a P-operation then the environment of reader\(^3_i\) also executes a P-operation.

\[ \tau_{i,4}^9 \]
\[ = (\epsilon = e \land \ell_{i,4} \in \{ 0, 7, 3 \} \land m = 0 \land \Psi_4 = \Psi_4 \left[ 1/m \right]) \]
\[ \rightarrow (\epsilon = e \land \ell_{i,4} \in \{ 0, 7, 3 \} \land m = 0 \land p_{ri} \land p_{ri} \land \Psi_3 = \Psi_3 \left[ 1/m \right]) \left[ f/X_3 \right] \]
\[ = (\tau_{i,13}^4 \land p_{ri} \land p_{ri} \land \Psi_3) \left[ f/X_3 \right] \]
If the environment of reader\(^4_i\) executes a V-operation then the environment of reader\(^3_i\) also executes a V-operation.

93
Readers/Writers Example

- \( \tau^{4,4} \)
  \[ = \left( \epsilon = e \land \ell^{4} \in \{0, 7, 3\} \land r = 0 \land \Psi^{4} = \Psi_{4}[1/r^{4}] \right) \]
  \[ \rightarrow \left( \epsilon = e \land \ell^{5} \in \{0, 7, 3\} \land r = 0 \land p_{r} \land p'_{r} \land \Psi^{5} = \Psi_{5}[1/r^{4}] \right) \left[ f/X_{3} \right] \]
  \[ = (\tau^{5,4} \land p_{r} \land p'_{r}) \left[ f/X_{3} \right] \]

If the environment of reader\(^{4}\) executes a V-operation then the environment of reader\(^{3}\) also executes a V-operation.

- \( \tau^{4,5} \)
  \[ = \left( \epsilon = e \land \ell^{4} \in \{0, 7, 3\} \land w = 0 \land \Psi^{4} = \Psi_{4}[1/w] \right) \]
  \[ \rightarrow \left( \epsilon = e \land \ell^{5} \in \{0, 7, 3\} \land w = 0 \land p_{r} \land p'_{r} \land \Psi^{5} = \Psi_{5}[1/w] \right) \left[ f/X_{3} \right] \]
  \[ = (\tau^{5,5} \land p_{r} \land p'_{r}) \left[ f/X_{3} \right] \]

If the environment of reader\(^{4}\) executes a V-operation then the environment of reader\(^{3}\) also executes a V-operation.

- \( \text{stut}^{4} \rightarrow \text{stut}^{3} \left[ f/X_{3} \right] \)

since \( r_{v} \) doesn't change.

(c) Let \( \text{WF}^{4}_{r_{i}} \overset{\Delta}{=} \{ \tau^{4}_{r_{i}} \mid k \in \{2, 5, 8\} \} \) and
\( \text{SF}^{4}_{r_{i}} \overset{\Delta}{=} \{ \tau^{4}_{r_{i,k}} \mid k \in \{1, 3, 4, 6, 7, 9, 10, 11, 12, 13, 14, 15\} \} \) then
\[ \text{L}^{4}_{r_{i}} \overset{\Delta}{=} \bigwedge_{\tau \in \text{WF}^{4}_{r_{i}}} (\Diamond \Box En(\tau) \rightarrow \Box \Diamond En(\tau)) \land \bigwedge_{\tau \in \text{SF}^{4}_{r_{i}}} (\Diamond \Box En(\tau) \rightarrow \Box \Diamond En(\tau)) \]

Let \( \text{WF}^{3}_{r_{i}} \overset{\Delta}{=} \{ \tau^{3}_{r_{i,k}} \mid k \in \{2, 5, 8\} \} \) and
\( \text{SF}^{3}_{r_{i}} \overset{\Delta}{=} \{ \tau^{3}_{r_{i,k}} \mid k \in \{1, 3, 4, 6, 7, 9, 10, 11, 12, 13, 14, 15\} \} \) then
\[ \text{L}^{3}_{r_{i}} \overset{\Delta}{=} \bigwedge_{\tau \in \text{WF}^{3}_{r_{i}}} (\Diamond \Box En(\tau) \rightarrow \Box \Diamond En(\tau)) \land \bigwedge_{\tau \in \text{SF}^{3}_{r_{i}}} (\Diamond \Box En(\tau) \rightarrow \Box \Diamond En(\tau)) \]

The following holds:
\[ \mathcal{S}_{4} \models \text{L}^{4}_{r_{i}} \rightarrow \text{L}^{3}_{r_{i}} \left[ f/X_{3} \right] \]

since \( \tau^{3}_{r_{i,k}} \) is relatively refined by \( \tau^{4}_{r_{i}} \) for \( k = 0, \ldots, 15 \).

So
\[ \mathcal{S}_{4} \models \text{L}^{3}_{r_{i}} \left[ f/X_{3} \right] \]

(ad 2) Analogue to the proof of (1).

(ad 3) This is trivial because \( W_{3} \leftrightarrow (X_{i=1}^{N} W_{r_{i}}^{3} \land X_{j=1}^{M} W_{w_{j}}^{3}) \).
Chapter 4

Stable Storage Example

This chapter first introduces in sect. 4.1 a general methodology for proving fault tolerant systems correct. This general methodology uses the relative refinement concept of sect. 2.3.2. The remaining sections of this chapter give an illustration of this general methodology by applying it to a fault tolerant system consisting of a number of disks implementing stable storage. Section 4.2 introduces this application. In sections 4.3, 4.4, 4.5 and 4.6 the four steps of this general methodology are applied to the stable storage example [Cri85, Sch91].

4.1 The General Methodology

The general methodology consists of four steps. In the first step one gives the abstract specification $S \triangleq (B, H)$ where $H$ is a DTL formula specifying the fault tolerant system. In this specification no faults are visible, hence they don’t occur as observables. The designer’s task is to give an implementation of this system under the assumption that only faults from certain classes can occur. These faults are called anticipated faults. These are faults which may affect the implementation in that they may give rise to errors in the state of the implementation, resulting subsequently in failures of that implementation. In step 2, 3 and 4 of the methodology a fault-tolerant system is developed.

The second step identifies the anticipated faults which can affect an implementation $S_P \triangleq (B_P, H_P)$. This implementation serves as first approximation to the final implementation of $S$. It should be clear that $S_P$ is not a refinement of $S$ because of the possible occurrences of anticipated faults. $S_P$ is only a refinement when these faults do not occur, i.e., $S_P$ is a relative refinement of $S$. So in step 2 we must prove:

$$S_P \text{ ref } S$$

In the third step one specifies how these anticipated faults are detected, i.e., one has to specify a detection layer $S_{D_s}$ for these faults. This layer is added in bottom-up fashion to the implementation $S_P$ of the second step and stops upon detection of the first error, i.e., $S_{D_s}$ is a fail-stop implementation. So the second approximation to the final implementation consists of the composition of $S_P$ and $S_{D_s}$. This approximation is clearly not a refinement because when in $S_P$ a fault occurs, and $S_{D_s}$ detects the corresponding error, the whole
approximation stops. One would like to have (eventually) an approximation that doesn’t stop, i.e., the physical disk isn’t affected by faults and the detection layer should detect no error. Let $\mathbf{W} \triangleq (W_{D_s}, W_P)$ where $W_P$ expresses that no faults occur and $W_{D_s}$ expresses that no error is detected. Then we must prove the following:

$$(2) \quad S_{D_s} \upharpoonright \mathbf{W} \upharpoonright S_P \mathsf{ref} W_P S_P.$$  

From (1), (2) and the transitivity of relative refinement relation follows:

$$S_{D_s} \upharpoonright \mathbf{W} \upharpoonright S_P \mathsf{ref} S.$$  

In the fourth step one specifies the corrective action to be undertaken after detection of an error. This means in general that one needs redundancy, i.e., several copies of $S_P$ and $S_D$ components, because when a detection layer $S_D$ detects an error, the state before that error has to be recovered and that can only be done by accessing another copy of $S_P$ through its corresponding detection layer $S_D$. Note that the $S_D$ component doesn’t stop anymore on the detection of an error but merely waits for the corrective action to be undertaken. Say, we need $N$ copies of $S_P$ and $S_D$. The final implementation consists then of those $N$ copies of $S_P$ and $S_D$ plus a recovery layer $S_R$. Let $W_R$ express which kind of errors can be recovered. If the following holds:

$$(3) \quad \|_{i=1}^{N} (S_P, \| S_{D_i}) \| S_R W_R \mathsf{ref} S_{D_s} \upharpoonright \mathbf{W} \upharpoonright S_P$$  

then from (1), (2), (3) and the transitivity of relative refinement follows the desired result, i.e.:

$$\|_{i=1}^{N} (S_P, \| S_{D_i}) \| S_R W_R \mathsf{ref} S$$  

This ends our exposition of the general methodology. In the next sections this methodology will be applied to a stable storage example.

4.2 Application: Introduction

Stable storage is defined as follows. A disk is used to store and retrieve data. During these operations some faults can occur in the underlying hardware. To make the disk more reliable one introduces layers for the detection and correction of errors, due to these faults. The system with these detection and correction layers is called “stable storage”. This stable storage is a fault tolerant system because it stores and retrieves data in a reliable way under the assumption that faults from a certain class are recovered (corrected). This class consists of two kinds of faults. The first one consists of faults that damage the disk surface— the contents of the disk are said to be corrupted by these faults. The second one consists of faults that affect the disk control system, and results into the contents of the disk being read from or written to the wrong location. Notice that other kinds of faults, such as power failure or physical destruction of the whole stable storage system, are not taken into account. I.e., stable storage should function correctly provided such latter faults do not occur.
4.3 First Step: Stable Storage

4.3.1 Introduction

In this section we give a specification of a stable storage system as we ideally would like to have it. So no faults are observed. If they occur internally, they should be repaired by the system without leaving any observable trace. For that is the meaning of ‘stable’ here!

4.3.2 Specification

The abstract specification of the stable storage specifies the following: The user signals with a read request event that he wants to read the contents of some location of stable storage. Stable storage will then respond by sending the requested contents. The user signals with a write request event that some data has to be written on some location of stable storage, with a response event the stable storage signals that the write has been performed. Note: we have a very simple stable storage that can handle only one request at a time. The formal specification $\mathcal{S} = (B, H)$ where $H \triangleq I \land T \land L$ and $B$, $I$, $T$ and $L$ are as follows:

1. **Basis** $B = ((\text{In}, \text{Out}), (V, X))$

   $\text{In} \triangleq \{\text{Rreq, Wreq}\}$,
   $\text{Out} \triangleq \{\text{Rres, Wres}\}$,
   $V \triangleq \emptyset$,
   $X \triangleq \{\ell, r, s, M[n] \mid n \in SN\}$

   where $SN$ is the set of sector numbers: $[1, ..., Z]$. Let $Inf$ be the set of information items that could be stored and retrieved by stable storage but that will not be further specified. For $n \in SN$ and $c, d \in Inf$:

   - **Rreq?($n$)**: the request to read sector $n$.
   - **Rres!(c)**: the response to the previous read request where $c$ are the contents of requested sector.
   - **Wreq?($d$)**: write information item $d$ onto sector $n$.
   - **Wres!**: previous write has been performed.
   - $\ell$: local variable indicating the status of the stable storage; $\ell = 0$ means no requests are issued, $\ell = 1$ means a read request has been issued, and $\ell = 2$ means a write request has been issued.
   - $r$: local variable indicating the requested sector.
   - $s$: local variable indicating the contents of the requested sector or the to be written data.
   - $M[n]$: the physical sector $n$.

   Let $\Psi_0 \triangleq (\ell, r, s, M[1], ..., M[Z])$ and $\Psi'_0 \triangleq (\ell', r', s', M'[1], ..., M'[Z])$.  

97
2. Initial States:

\[ I \models \ell = 0 \wedge \bigwedge_{i \in SN} M[i] = dflt \]

Where \( dflt \in Inf \) is some default information item.

3. Transitions:

\[ T \models \]

\[ \tau_1 \quad \left( \epsilon = Rreq? (n) \wedge \ell = 0 \wedge \Psi_0 = \Psi_0 [1, n/\ell, r] \right) \]

The user requests the contents of sector \( n \).

\[ \tau_2 \quad \vee \left( \epsilon = Rres (M[r]) \wedge \ell = 1 \wedge \Psi_0 = \Psi_0 [0/\ell] \right) \]

Stable storage responds with the contents of the requested sector.

\[ \tau_3 \quad \vee \left( \epsilon = Wreq? (n, d) \wedge \ell = 0 \wedge \Psi_0 = \Psi_0 [2, n, d/\ell, r, s] \right) \]

The user requests that \( d \) should be written onto sector \( n \).

\[ \tau_4 \quad \vee \left( \epsilon = Wres! \wedge \ell = 2 \wedge \Psi_0 = \Psi_0 [0, s/\ell, M[r]] \right) \]

Stable storage responds with a signal that requested write is performed.

\[ \tau_0 \quad \vee \text{stut} \]

These transitions are illustrated in figure 4.1

![Diagram of transitions](image)

**Figure 4.1:** Transitions of stable storage.

4. Liveness condition:

The liveness condition expresses that the communication transitions are strongly fair.

Let \( SF = \{ \tau_i \mid i \in \{1, 2, 3, 4\} \} \) then

\[ L \models \bigwedge_{\tau \in SF} (\Box \Diamond En(\tau) \rightarrow \Box \Diamond \tau) \]
4.4 Second Step: Physical Disk

4.4.1 Introduction

In this step, which is the first stage in our task to develop a fault tolerant system, we give the specification of a physical disk. This specification is a first approximation to our fault tolerant system, i.e., it acts as bottom layer of our desired implementation and because the other layers haven’t been developed yet it is the only layer we have at this moment. In this specification we must specify, because this is the first stage of our development, which are the anticipated faults our system, i.e., we have to specify which are the faults of our interest that could affect a physical disk. These faults are represented as events in our formalism. This first approximation of stable storage is not a correct one because of these anticipated faults (the physical disk doesn’t anticipate on these faults at all!). But under the assumption that these faults don’t occur this first implementation is a refinement of stable storage.

4.4.2 Specification

We must specify a physical disk, the anticipated faults and their impact on the physical disk. We take as anticipated faults the following ones (cf. [Cri85, Sch91]):

- Damages of the disk surface causing corruption of the contents of a physical sector.
- Disk control faults causing the contents of a particular physical sector to be read or written at a wrong location.

These two faults are described using two events: the \textit{dam} event standing for a damage to the disk surface and the \textit{csf} event standing for a disk control system fault. As in the specification of stable storage, the user requests with $\text{Rreq}(n)$ that it wants to read the contents of physical sector $n$. The physical disk then responds with $\text{Rres}(c)$ delivering the requested contents. With $\text{Wreq}(n, d)$ the user signals that $d$ should be written onto sector $n$. The physical disk responds with $\text{Wres}$ that the requested information has been written. The formal specification $S_P = (B_P, \Pi_P)$ where $H_P \triangleq I_P \land T_P \land L_P$ and $B_P, I_P, T_P$ and $L_P$ are as follows:

1. **Basis** $B_P = ((\text{In}_P, \text{Out}_P), (V_P, X_P))$

   \begin{align*}
   \text{In}_P & \triangleq \{ \text{Rreq}, \text{Wreq} \}, \\
   \text{Out}_P & \triangleq \{ \text{Rres}, \text{Wres} \}, \\
   V_P & \triangleq \emptyset, \\
   X_P & \triangleq \{ \ell_P, r_P, s_P, M_P[n], F[n] \mid n \in PN \}
   \end{align*}

where $PN$ is the set of physical sector numbers: $\{1, \ldots, Y\}$. Let $Ph_y$ be the set of information items that could be stored and retrieved by the physical disk but that will not be further specified. The special information item $\odot$ is introduced to model disk surface damage faults. For $n \in PN$ and $c, d \in Ph_y$.
Stable Storage Example

- \textbf{Rreq}?(n): the request to read sector \(n\).
- \textbf{Rres}!(c): the response to the previous request where \(c\) are the contents of requested sector.
- \textbf{Wreq}?(d): write information item \(d\) onto sector \(n\).
- \textbf{Wres}!: response that previous write has been performed.
- \(\ell_P\): local variable indicating the status of the physical disk.
- \(r_P\): local variable indicating the requested physical sector.
- \(s_P\): local variable indicating the requested contents or the data to be written.
- \(M_P[n]\): the physical sector \(n\).
- \(F\): the control system, i.e., the control system maps sector \(n\) to sector \(F[n]\).

Let \(\Psi_1 \triangleq (\ell_P, r_P, s_P, M_P[1], \ldots, M_P[Y], F[1], \ldots, F[Y])\) and \(\Psi'_1 \triangleq (\ell'_P, r_P, s'_P, M'_P[1], \ldots, M'_P[Y], F'[1], \ldots, F'[Y])\).

2. Initial States:

\[
1_P \triangleq \ell_P = 0 \land \bigwedge_{i \in PN} (M_P[i] = dflt \land F[i] = i)
\]

All sectors contain the default data item \(dflt\) and the control system has not been affected by control system faults.

3. Transitions:

\[
T_P \triangleq \begin{align*}
\tau_{P,1} & \quad \left( \epsilon = \text{Rreq}?(n) \land \ell_P = 0 \land \Psi'_1 = \Psi_1[1, n/\ell_P, r_P] \right) \\
& \quad \text{The user requests the contents of sector } n.
\tau_{P,2} & \quad \lor \left( \epsilon = \text{Rres}!(M_P[F[r_P]]) \land \ell_P = 1 \land \Psi'_1 = \Psi_1[0/\ell_P] \right) \\
& \quad \text{The physical disk responds with the contents of the requested sector.}
\tau_{P,3} & \quad \lor \left( \epsilon = \text{Wreq}?(n, d) \land \ell_P = 0 \land \Psi'_1 = \Psi_1[2, n, d/\ell_P, r_P, s_P] \right) \\
& \quad \text{The user requests that } d \text{ should be written onto sector } n.
\tau_{P,4} & \quad \lor \left( \epsilon = \text{Wres}!(n, d) \land \ell_P = 2 \land \Psi'_1 = \Psi_1[0, s_P/\ell_P, M_P[F[r_P]]] \right) \\
& \quad \text{The physical disk responds with a signal that requested write is performed.}
\tau_{P,5} & \quad \lor \left( \epsilon = i \land n \neq j \land \Psi'_1 = \Psi_1[j/F[n]] \right) \\
& \quad \text{Due to control system fault the sector } n \text{ is mapped to sector } j.
\tau_{P,6} & \quad \lor \left( \epsilon = i \land \Psi'_1 = \Psi_1[j/M_P[n]] \right) \\
& \quad \text{Due to disk surface fault the contents of sector } n \text{ are replaced by corrupted data } \overline{\circ}.
\tau_{P,0} & \quad \lor \text{stat}_P
\end{align*}
\]

These transitions are illustrated in figure 4.2 where \(fault\) is either a control system fault or a disk surface fault.
4.4 Second Step: Physical Disk

![Diagram of transitions of the physical disk]

Figure 4.2: Transitions of the physical disk.

4. **Liveness condition:**
   The liveness condition expresses that the communication transitions are strongly fair. Let \( SF_p = \{ \tau_{Pi} \mid i \in \{1, 2, 3, 4\} \} \) then
   \[
   L_P = \bigwedge_{\tau \in SF_P} (\square \diamond En(\tau) \rightarrow \square \diamond \tau)
   \]

4.4.3 **Requirement** \( W_P \)

The requirement \( W_P \) expresses that the control system and disk surface faults never occur.

\[
W_P = \square (\bigwedge_{i \in PN} (M_P[i] \neq \varnothing \land F[i] = i))
\]

This corresponds to the following machine: Let

\[
\begin{align*}
p & \triangleq \bigwedge_{i \in PN} (M_P[i] \neq \varnothing \land F[i] = i) \\
p' & \triangleq \bigwedge_{i \in PN} (M_P'[i] \neq \varnothing \land F'[i] = i)
\end{align*}
\]

then \( W_P \) is equal to the machine \((p \land \square ((p \land p') \lor std_P)).\)

4.4.4 **\( S_P \) relatively refines \( S \)**

We should prove \( S_P \) relatively refines \( S \). Let the external requirement for the system \( S \) be \textbf{true} (i.e., no extra requirement is imposed). According to theorem 8 \( S_P \) relatively refines \( S \) with respect to \((W_P, W)\) iff the following holds:

\[
\mathcal{D}(B_P) = \mathcal{D}(B) \quad \text{and} \quad \models (\exists X_P . (G_P)) \rightarrow (\exists X . (G))
\]
where $X_P$ are the local variables from $\mathcal{S}_P$, i.e., $X_P \triangleq \{ \ell_P, r_P, s_P, M_P[n], F[n] \mid n \in PN \}$ and $G_P$ is defined as

$$I_P \land \Box T_P \land I_P \land W_P$$

$X$ are the local variables from $\mathcal{S}$, i.e., $X \triangleq \{ \ell, r, s, M[n] \mid n \in SN \}$ and $G$ is defined as

$$I \land \Box T \land I$$

Rule 3 will be used to prove

$$\models (\exists X_P. (G_P)) \land (\exists X. (G))$$

This means one has to prove (a), (b) and (c) below, for $\bar{f}$ the refinement mapping from $\mathcal{S}_P$ to $\mathcal{S}$, defined as: $\bar{f} = f_t, f_r, f_s, f_{M[n]} \ (n \in SN)$. We will assume that the set of sector numbers $SN$ is equal to the set of physical sector numbers $PN$. The refinement mappings are defined as:

$$f_\ell \triangleq \ell_P$$
$$f_r \triangleq r_P$$
$$f_s \triangleq s_P$$
$$f_{M[n]} \triangleq M_P[n]$$

(a) Proof 9

$$I_P \land p = \% \text{ Def. } I_P, p$$
$$\ell_P = 0 \land \land_{i \in PN}(M_P[i] = dflt \land F[i] = i)$$
$$\land \land_{i \in PN}(M_P[i] \neq \odot \land F[i] = i)$$
$$\rightarrow \% \text{ Def. bar f}$$
$$\land (\ell = 0 \land \land_{i \in SN} M[i] = dflt) [\bar{f}/X]$$
$$\models I [\bar{f}/X]$$

(b) Proof 10

Since $T_P$ is of the form $\text{stut}_P \lor \forall_\ell (\epsilon = a_\ell \land \text{trans}_\ell)$ then $T_P \land ((p \land p') \lor \text{stut}_P)$ is equal to $\text{stut}_P \lor \forall_\ell (\epsilon = a_\ell \land \text{trans}_\ell \land p \land p')$.

$$\tau_{P,1} \land p \land p'$$
$$= \left( \epsilon = \text{Rreq}(n) \land \ell_P = 0 \land p \land p' \land \Psi_1 = \Psi_1[1, n/\ell_P, r_P] \right)$$
$$\rightarrow \left( \epsilon = \text{Rreq}(n) \land \ell = 0 \land \Psi_0 = \Psi_0[1, n/\ell, r] \right) [\bar{f}/X]$$
$$\tau_1 [\bar{f}/X]$$

The read request at the physical disk level corresponds to the read request at the abstract level.
4.4 Second Step: Physical Disk

\[- \tau_{P,3} \land p \land p' \]
\[= (\epsilon = \text{Res}(M_P[F[r_P]]) \land \ell_P = 1 \land p \land p' \land \Psi_1 = \Psi_1[0/\ell_P]) \]
\[\rightarrow (\epsilon = \text{Res}(M[r]) \land \ell = 1 \land \Psi_0 = \Psi_0[0/\ell]) \left[ j/X \right] \]
\[= \tau_2 \left[ j/X \right] \]

The read response at the physical disk level corresponds to the read response at the abstract level.

\[- \tau_{P,4} \land p \land p' \]
\[= (\epsilon = \text{Wreq?}(n, d) \land \ell_P = 0 \land p \land p' \land \Psi_0 = \Psi_0[2, n, d/\ell, r_p, s_p]) \]
\[\rightarrow (\epsilon = \text{Wreq?}(n, d) \land \ell = 0 \land \Psi_0 = \Psi_0[2, n, d/\ell, r, s]) \left[ j/X \right] \]
\[= \tau_3 \left[ j/X \right] \]

The write request at the physical disk level corresponds to the write request at the abstract level.

\[- \tau_{P,5} \land p \land p' \]
\[= (\epsilon = \text{Res!}(M_P[F[r_P]]) \land \ell_P = 2 \land p \land p' \land \Psi_0 = \Psi_0[0, s_p/\ell_P, M_P[F[r_P]]) \]
\[\rightarrow (\epsilon = \text{Res!}(M[r]) \land \ell = 2 \land \Psi_0 = \Psi_0[0, s/\ell, M[r]]) \left[ j/X \right] \]
\[= \tau_4 \left[ j/X \right] \]

The write response at the physical disk level corresponds to the write response at the abstract level.

Due to the external requirement the disk control fault transition can not be taken, i.e., is equal to false and from false everything can be inferred.

\[- \tau_{P,6} \land p \land p' \]
\[= (\epsilon = i \land n \neq j \land p \land p' \land \Psi_1 = \Psi_1[j/F[n]]) \]
\[= \text{false} \]
\[\rightarrow T \left[ j/X \right] \]

Due to the external requirement the disk surface fault transition can not be taken, i.e., is equal to false and from false everything can be inferred.

\[- \text{stut}_P \rightarrow \text{stut} \left[ j/X \right] \]

(c) Let SF\(_P = \{ \tau_{P,i} \mid i \in \{1, 2, 3, 4\} \} \) then

\[L = \bigwedge_{\tau \in SF_P} (\Box \Diamond En(\tau) \rightarrow \Box \Diamond \tau) \]

. Let SF = \{ \tau_i \mid i \in \{1, 2, 3, 4\} \} then

\[L = \bigwedge_{\tau \in SF} (\Box \Diamond En(\tau) \rightarrow \Box \Diamond \tau). \]
Then the following holds

\[ S_P \cap Hist(W_P) = L_P \rightarrow L [f/X] \]

since \( \tau_i \) is relatively refined by \( \tau_{P_i} \) for \( i = 1, \ldots, 4 \). So

\[ S_P \cap Hist(W_P) = L [f/X] \]

holds.

4.5 Third Step: Fail-Stop Detection Layer

4.5.1 Introduction

In this step, the second stage in our development of the fault tolerant system, we specify in bottom-up fashion on top of the physical disk that has been specified in Section 4.4, the layer that detects the faults that we assumed could affect the physical disk (the anticipated faults). The detection layer acts as a sort of "interface" between the user and the physical disk. It stops when an anticipated fault is detected by the detection mechanism, i.e., the whole system (detection layer plus physical disk) stops when such a fault occurs. It also informs the user which kind of anticipated fault has occurred. This second implementation is "better" than the first one because now the user is certain, under the assumption that the detection mechanism detects all the anticipated faults, that the retrieved data is reliable. The implementation of the detection layer is such that as soon as a fault is detected the system stops. This is called a fail-stop implementation [LA90]. As seen above, there are two classes of anticipated faults. Consequently there are two kinds of detection mechanisms. The first one checks whether the contents read from the physical disk are corrupted, i.e., detects errors due to damage of the disk surface. This is done with a cyclic redundancy mechanism [LA90]. The second one checks whether the contents of read from the physical disk originate from the right location. This is done with an address checking mechanism [LA90] which encodes the location of the contents of the physical disk in the contents itself.

4.5.2 Specification

The detection layer consists of three parts: the first part checks whether the data retrieved from the physical disk is affected by a corrupt data fault (the fault that damages the disk surface). This is done with a cyclic redundancy check (CRC) mechanism [LA90]. The second part checks whether the data retrieved from the physical disk is from the correct physical location, i.e., whether it is affected by a disk control system fault. This is done with an address checking (ADR) mechanism [LA90]. The third part prevents further access by the user of the physical disk when one of these two mechanisms detects a fault. This can be easily done because the detection layer acts as "interface" between the user and the physical disk, the detection layer then refuse to communicate with the user and the physical disk. Furthermore this part then gives a message to inform the user which anticipated fault has occurred.
4.5 Third Step: Fail-Stop Detection Layer

The protocol of this interface between user and physical disk is as follows. The user read requests the contents of some physical sector by issuing a \textbf{Rreq}(n) event to the detection disk layer. This detection disk layer issues after receipt of this event a \textbf{Rreqp}(m) event to the physical disk. The physical disk then responds with a \textbf{Rresp}(c) event delivering the requested contents of that physical sector. The detection layer then responds after checking the contents with a \textbf{Rresp}(cd) event delivering either the requested contents or an error message. The user write requests that \textit{d} should be written on sector \textit{n} by issuing a \textbf{Wreq}(n, \textit{d}) event to the detection layer. The detection layer then issues a \textbf{Wreqp}(m, \textit{dd}) event to the physical disk requesting that \textit{dd} is written on sector \textit{m}. The physical disk then responds with a \textbf{Wresp} event that the requested information is written. The detection layer then responds to the user with a \textbf{Wres} event that the information is written.

Logical sector numbers are introduced now, but are used in the next step to correct disk surface damage faults, i.e., when the detection layer detects that data from a physical sector number is affected by a disk surface damage fault, the correct data will be written to another physical sector number. In order to retrieve these contents from this new location logical sector numbers are introduced. When contents are stored at a new physical sector the logical sector number will be pointing to this new sector. So actually the data are retrieved from their logical sector number. In this step however, the mapping between the logical sector numbers and the physical sector numbers will be the identity mapping because they are not needed here. The detection layer is described more formally by the following specification: \( S_{D_s} = (B_{D_s}, H_{D_s}) \) where \( H_{D_s} \triangleq I_{D_s} \wedge \square T_{D_s} \wedge L_{D_s} \), and \( B_{D_s}, I_{D_s}, T_{D_s}, \) and \( L_{D_s} \) are as follows:

1. \textbf{Basis} \( B_{D_s} = ((I_{D_s}, \text{Out}_{D_s}), (V_{D_s}, X_{D_s})) \)

\[
\begin{align*}
\text{In}_{D_s} & \triangleq \{ \text{Rreq, Wreq, Rresp, Wresp} \} \\
\text{Out}_{D_s} & \triangleq \{ \text{Rres, Wres, Rreqp, Wreqp}, \}
\end{align*}
\]
\[
\begin{align*}
V_{D_s} & \triangleq \emptyset, \\
X_{D_s} & \triangleq \{ \ell_{D_s}, T_{D_s}, s_{D_s}, L_{D_s}[i] | i \in LN \}
\end{align*}
\]

where \( LN \) is the set of logical sector numbers: \([1, \ldots, Y]\). Let \( Lg \) the set of data items that the user wants to store on or to retrieve from the physical disk and \( Phy \) the set information items that can be stored on or retrieved from the physical disk (Note: an item from \( Phy \) is an crc-encoded and address-encoded item of \( Lg \).) For \( n \in LN, c, d \in Lg, m \in P N \) and \( cd, dd \in Phy \):

- \textbf{Rreq}(n): the request from the user to read logical sector \textit{n}.
- \textbf{Rres!(c)}: the response of the detection layer to the previous request where \textit{c} are the crc-decoded and address-decoded contents of the requested logical sector \textit{n}.
- \textbf{Wreq!(n, d)}: write information item \textit{d} onto logical sector \textit{n}.
- \textbf{Wres!}: response that previous write has been performed.
- \textbf{Rreqp!(m)}: the request from the detection layer to read physical sector \textit{m}.
- \textbf{Rresp!(cd)}: the response of the physical disk to the previous request where \textit{c} are the crc-encoded and address-encoded contents of requested physical.

105
Stable Storage Example

- $\text{Wreqp}(m, dd)$: write information item $dd$ onto physical sector $m$.
- $\text{Wresp}$: response that previous write has been performed.
- $\ell_{Ds}$: local variable indicating the status of the detection layer; $\ell_{Ds} = 0$: the detection layer is waiting for a request, $\ell_{Ds} = 1$: the user has issued a read request, $\ell_{Ds} = 2$: the detection layer has issued a read request, $\ell_{Ds} = 3$: the physical has responded to a read request with correct data, $\ell_{Ds} = 4$: the physical disk has responded to a read request with incorrect data, $\ell_{Ds} = 5$: the detection layer has responded to a read request with an error message (stop status), $\ell_{Ds} = 6$: the user has issued a write request, $\ell_{Ds} = 7$: the detection layer has issued a write request, $\ell_{Ds} = 8$: the physical disk has responded to a write request.
- $r_{Ds}$: local variable indicating the requested sector.
- $s_{Ds}$: local variable indicating the requested information or the data to be written.
- $LS_{Ds}[i]$: the physical sector mapped to logical sector $i$.

Let $\Psi_1 = \langle \ell_{Ds}, r_{Ds}, s_{Ds}, LS_{Ds}[1], \ldots, LS_{Ds}[Y] \rangle$ and $\Psi_2 = \langle \ell'_{Ds}, r'_{Ds}, s'_{Ds}, LS'_{Ds}[1], \ldots, LS'_{Ds}[Y] \rangle$.

2. Initial states:

$$I_{Ds} \triangleq \ell_{Ds} = 0 \land \bigwedge_{i \in LN} LS_{Ds}[i] = i$$

3. Transitions:

To describe the two detecting mechanisms as transitions the following functions are needed: (see [LA90] for more information about this CRC-coding)

- $\text{CC} : \text{Phy} \rightarrow \text{Bool}$
  (Crc-Check) Is used to check whether data from the physical disk is damaged by a disk surface fault.
- $\text{CD} : \text{Phy} \rightarrow (\text{Lg} \times \text{PN})$
  (Crc-Decode) Is used to decode the CRC-coded physical data into address format.
- $\text{CE} : (\text{Lg} \times \text{PN}) \rightarrow \text{Phy}$
  (Crc-Encode) Is used to encode data in address format into physical CRC format.
- $\text{AC} : (\text{Lg} \times \text{PN} \times \text{PN}) \rightarrow \text{Bool}$
  (Adr-Check) Is used to check whether data is read from the correct physical location.
- $\text{AD} : (\text{Lg} \times \text{PN}) \rightarrow \text{Lg}$
  (Adr-Decode) Is used to decode data in address format into user format.
4.5 Third Step: Fail-Stop Detection Layer

- \( AE : (LN \times Lg) \rightarrow (Lg \times PN) \)
  (Adr-Encode) Is used to encode a physical sector number and a information item given by the user into address format.

Let

\[
\begin{align*}
\text{Good} & \triangleq CC(cd) \land AC(CD(cd), LSDs[rDs]) \\
\text{A.err} & \triangleq CC(cd) \land \neg AC(CD(cd), LSDs[rDs]) \\
\text{C.err} & \triangleq \neg CC(cd) \\
c & \triangleq AD(CD(cd)) \\
m & \triangleq LSDs[rDs] \\
dd & \triangleq CE(AE(rDs, sDs)) \\
c_1 & \triangleq \text{address error} \\
c_2 & \triangleq \text{crc error}
\end{align*}
\]

\[T_{Ds} \triangleq\]

\[
\begin{align*}
\tau_{Ds,1} \quad & \left( \epsilon = Rreq?(n) \land \ell_{Ds} = 0 \land \Psi'_2 = \Psi_2[1, n/\ell_{Ds}, r_{Ds}] \right) \\
& \text{The user requests the contents of logical sector } n.
\end{align*}
\]

\[
\begin{align*}
\tau_{Ds,2} \quad & \lor \left( \epsilon = Rreqp!(m) \land \ell_{Ds} = 1 \land \Psi'_2 = \Psi_2[2/\ell_{Ds}] \right) \\
& \text{The detection layer requests the to logical sector } r_{Ds} \text{ mapped physical sector.}
\end{align*}
\]

\[
\begin{align*}
\tau_{Ds,3} \quad & \lor \left( \epsilon = Resp?(cd) \land \ell_{Ds} = 2 \land Good \land \Psi'_2 = \Psi_2[3, c/\ell_{Ds}, r_{Ds}] \right) \\
& \text{The physical disk responds with the contents of the requested sector and the detection layer detects no error in them.}
\end{align*}
\]

\[
\begin{align*}
\tau_{Ds,4} \quad & \lor \left( \epsilon = Resp?(cd) \land \ell_{Ds} = 2 \land A.err \land \Psi'_2 = \Psi_2[4, c_1/\ell_{Ds}, s_{Ds}] \right) \\
& \text{The physical disk responds with the contents of the requested sector and the detection layer detects a control system error.}
\end{align*}
\]

\[
\begin{align*}
\tau_{Ds,5} \quad & \lor \left( \epsilon = Resp?(cd) \land \ell_{Ds} = 2 \land C.err \land \Psi'_2 = \Psi_2[4, c_2/\ell_{Ds}, s_{Ds}] \right) \\
& \text{The physical disk responds with the contents of the requested sector and the detection layer detects a disk surface damage error.}
\end{align*}
\]

\[
\begin{align*}
\tau_{Ds,6} \quad & \lor \left( \epsilon = Resp!(s_{Ds}) \land \ell_{Ds} = 3 \land \Psi'_2 = \Psi_2[0/\ell_{Ds}] \right) \\
& \text{The detection layer responds with the contents of the user requested sector.}
\end{align*}
\]

\[
\begin{align*}
\tau_{Ds,7} \quad & \lor \left( \epsilon = Resp!(s_{Ds}) \land \ell_{Ds} = 4 \land \Psi'_2 = \Psi_2[5/\ell_{Ds}] \right) \\
& \text{The detection layer responds with an error message and then stops.}
\end{align*}
\]
\( \tau_{D,s,8} \lor (\epsilon = \mathbf{Wreq}^2(n, d) \land \ell_{D,s} = 0 \land \Psi'_2 = \Psi_2[6, n, d/\ell_{D,s}, \nu_{D,s}, s_{D,s}] ) \)

The user requests that \( d \) should be written onto logical sector \( n \).

\( \tau_{D,s,9} \lor (\epsilon = \mathbf{Wreqp}(m, dd) \land \ell_{D,s} = 6 \land \Psi'_2 = \Psi_2[7/\ell_{D,s}] ) \)

The detection requests that \( dd \) should be written onto physical sector \( m \).

\( \tau_{D,s,10} \lor (\epsilon = \mathbf{Wresp}? \land \ell_{D,s} = 7 \land \Psi'_2 = \Psi_2[8/\ell_{D,s}] ) \)

The physical disk responds with a signal to the detection layer that requested write is performed.

\( \tau_{D,s,11} \lor (\epsilon = \mathbf{Wres!} \land \ell_{D,s} = 8 \land \Psi'_2 = \Psi_2[0/\ell_{D,s}] ) \)

The detection layer responds with a signal to the user that the requested write is performed.

\( \tau_{D,s,0} \lor \mathbf{stut}_{D,s} \)

These transitions are illustrated in figure 4.3

![Figure 4.3: Transitions of the fail-stop detection layer.](image)

4. **Liveness condition:**

The liveness condition expresses that the communication transitions are strongly fair. Let \( \text{SF}_{D,s} = \{ \tau_{D,s,i} \mid i \in \{1, \ldots, 11\} \} \) then

\[
L_{D,s} = \bigwedge_{\tau \in \text{SF}_{D,s}} (\Box \Diamond E_{n}(\tau) \rightarrow \Box \Diamond \tau)
\]

108
4.5 Third Step: Fail-Stop Detection Layer

4.5.3 Requirement \( W_{D_s} \)

The requirement \( W_{D_s} \) should express that no errors are detected.

\[
W_P = \Box (\epsilon = \text{Rresp}(cd) \rightarrow \text{Good})
\]

4.5.4 \( S_{D_s} \parallel S_P \) relatively refines \( S_P \)

The communication channels of \( S_P \) in \( S_{D_s} \parallel S_P \) should be renamed in order to compose \( S_{D_s} \) with \( S_P \), i.e., instead of \( H_P \) as specification we should take

\[
H_P [\text{Rreqp}, \text{Wreqp}, \text{Rresp}, \text{Wresp}/\text{Rreq}, \text{Wreq}, \text{Res}, \text{Wres}] .
\]

Let \( S_{P1} \) be the specification with this renaming. According to theorem 8 \( S_{D_s} \mid \Box \mid S_{P1} \) relatively refines \( S_P \) with respect to \((\text{true}, W_P)\) iff the following holds:

\[
\begin{align*}
\forall (B_{D_s,P1}) & = \forall (B_P) \text{ and} \\
\models (\exists X_{D_s,P1} \cdot (G_{D_s,P1})) & \rightarrow (\exists X_P \cdot (G_P))
\end{align*}
\]

where \( X_{D_s,P1} \) are the local variables from \( S_{D_s} \parallel S_{P1} \), i.e.,

\[
X_{D_s,P1} \triangleq \{ \ell_{D_s}, r_{D_s}, s_{D_s}, L_{SD_s}[i] \mid i \in LN \} \cup \{ \ell_{P1}, r_{P1}, s_{P1}, M_{P1}[n], F[n] \mid n \in PN \}
\]

and \( G_{D_s,P1} \) is defined as

\[
\exists \epsilon_1, \epsilon_2 \cdot B_{D_s} \cup B_{P1} \cdot (\text{H}_{D_s} \land W_{D_s})[\epsilon_1/\epsilon] \land (\text{H}_{P1} \land W_{P1})[\epsilon_2/\epsilon]
\]

This can be rewritten to following machine specification of \( S_2 : S_2 = (B_2, H_2) \) where \( H_2 \triangleq I_2 \land \Box T_2 \land L_2 \) and \( B_2, I_2, T_2 \) and \( L_2 \) are as follows:

1. **Basis** \( B_2 = ((\text{In}_2, \text{Out}_2), (V_2, X_2)) \)

   \[
   \begin{align*}
   \text{In}_2 & \triangleq \{ \text{Rreq}, \text{Wreq} \} \\
   \text{Out}_2 & \triangleq \{ \text{Res}, \text{Wres} \}, \\
   V_2 & \triangleq 0, \\
   X_2 & \triangleq \{ \ell_{D_s}, r_{D_s}, s_{D_s}, L_{SD_s}[i] \mid i \in LN \} \\
   & \cup \{ \ell_{P1}, r_{P1}, s_{P1}, M_{P1}[n], F[n] \mid n \in PN \}
   \end{align*}
   \]

   Let

   \[
   \begin{align*}
   \Psi_{I2} & \triangleq (\ell_{D_s}, r_{D_s}, s_{D_s}, L_{SD_s}[i], \ldots, L_{SD_s}[Y], \\
   & \ell_{P1}, r_{P1}, s_{P1}, M_{P1}[i], \ldots, M_{P1}[Y], F[i], \ldots, F[Y]) \\
   \Psi'_{I2} & \triangleq (\ell'_{D_s}, r'_{D_s}, s'_{D_s}, L'_{SD_s}[i], \ldots, L'_{SD_s}[Y], \\
   & \ell'_{P1}, r'_{P1}, s'_{P1}, M'_{P1}[i], \ldots, M'_{P1}[Y], F'[i], \ldots, F'[Y])
   \end{align*}
   \]

2. **Initial states**:

   \[
   I_2 \triangleq \ell_{D_s} = 0 \land \ell_{P1} = 0 \land \bigwedge_{i \in LN} L_{SD_s}[i] = i \land \bigwedge_{i \in PN} (M_{P1}[i] = \text{dflt} \land F[i] = i)
   \]

\[109\]
3. Transitions:

Let

\[
\begin{align*}
    c & \triangleq AD(CD(M_{P1}[F_{P1}])) \\
    m & \triangleq LS_{D_s}[r_{D_s}] \\
    dd & \triangleq CE(AE(r_{D_s}, s_{D_s}))
\end{align*}
\]

the address- and crc-decoded contents

physical sector

address- and crc-encoded contents

\[
T_2 \triangleq \epsilon
\]

\[
\begin{align*}
    \tau_{2,1} & \triangleq (\epsilon = \text{Rreq}(n)) \land \ell_{D_s} = 0 \land \ell_{P1} = 0 \land \Psi'_{12} = \Psi_{12}[1, n/\ell_{D_s}, r_{D_s}] \\
    \tau_{2,2} & \triangledown (\epsilon = i \land \ell_{D_s} = 1 \land \ell_{P1} = 0 \land \Psi'_{12} = \Psi_{12}[2, 1, m/\ell_{D_s}, \ell_{P1}, r_{P1}]) \\
    \tau_{2,3} & \triangledown (\epsilon = i \land \ell_{D_s} = 2 \land \ell_{P1} = 1 \land \Psi'_{12} = \Psi_{12}[3, c, 0/\ell_{D_s}, s_{D_s}, \ell_{P1}]) \\
    \tau_{2,4} & \triangledown (\epsilon = \text{Wres!(}s_{D_s}) \land \ell_{D_s} = 3 \land \ell_{P1} = 0 \land \Psi'_{12} = \Psi_{12}[0/\ell_{D_s}]) \\
    \tau_{2,5} & \triangledown (\epsilon = \text{Wreq?(}n, d) \land \ell_{D_s} = 0 \land \ell_{P1} = 0 \land \Psi'_{12} = \Psi_{12}[6, n, d/\ell_{D_s}, \ell_{P1}, r_{P1}]) \\
    \tau_{2,6} & \triangledown (\epsilon = i \land \ell_{D_s} = 6 \land \ell_{P1} = 0 \land \Psi'_{12} = \Psi_{12}[7, 2, m, dd/\ell_{D_s}, \ell_{P1}, r_{P1}, s_{P1}]) \\
    \tau_{2,7} & \triangledown (\epsilon = i \land \ell_{D_s} = 7 \land \ell_{P1} = 2 \land \Psi'_{12} = \Psi_{12}[8, 0, sp_1/\ell_{D_s}, \ell_{P1}, M_{P1}[F_{P1}]) \\
    \tau_{2,8} & \triangledown (\epsilon = \text{Wres!} \land \ell_{D_s} = 8 \land \ell_{P1} = 0 \land \Psi'_{12} = \Psi_{12}[0/\ell_{D_s}]) \\
    \tau_{2,9} & \triangledown stut_1
\end{align*}
\]

Figure 4.4 illustrates the transitions of the relative composed system $S_{D_s} \mid [\psi] \mid S_{P1}$. Due to $B_{\psi} \circ B_{\psi} (\epsilon, \epsilon_1, \epsilon_2)$ the communications events with the physical disk are transformed into i events and due to $W_{D_s}$ and $W_{P1}$ no faults occur and no errors are detected.

4. Liveness condition:

The liveness condition expresses that all non-stutter transitions are strongly fair. Let $SF_{D_s} = \{ \tau_{2,i} \mid i \in \{1, \ldots, 8\} \}$ then

\[
L_2 \triangleq \bigwedge_{\tau \in SF_2} (\Box \Diamond En(\tau) \rightarrow \Box \Diamond \tau)
\]
4.5 Third Step: Fail-Stop Detection Layer

Figure 4.4: Transitions of the relative composed system.

$X_P$ are the local variables from $S_P$, i.e., $X_P \triangleq \{\ell_P, r_P, s_P, M_P[n], F[n] \mid n \in PN\}$ and $G_P$ is defined as

$I_P \land \Box T_P \land L_P \land W_P$

Rule 3 will be used to prove

$\vdash (\exists X_2.(G_2)) \rightarrow (\exists X_P.(G_P))$.

This means one has to prove (a), (b) and (c) below, for $\bar{f}$ the refinement mapping from $S_2$ to $S_P$, defined as: $\bar{f} = f_{\ell_P}, f_{r_P}, f_{s_P}, f_{M_P[n]}, f_{F_P[n]} \ (n \in SN)$. The refinement mappings are defined as:

$$f_{\ell_P}$$

\begin{align*}
\text{if } & \ell_{D_3} = 0 \text{ then } \ell_{D_3} \\
& \ell_{D_3} = 1 \text{ then } \ell_{D_3} \\
& \ell_{D_3} = 2 \text{ then } \ell_{D_3} - 1 \\
& \ell_{D_3} = 3 \text{ then } \ell_{D_3} - 2 \\
& \ell_{D_3} = 6 \text{ then } \ell_{D_3} - 4 \\
& \ell_{D_3} = 7 \text{ then } \ell_{D_3} - 5 \\
& \ell_{D_3} = 8 \text{ then } \ell_{D_3} - 6
\end{align*}
Stable Storage Example

\[
\begin{align*}
  f_{r_P} & \triangleq r_{D_3} \\
  f_{s_P} & \triangleq s_{D_3} \\
  f_{M_P[n]} & \triangleq AD(CD(M_P[1][L_S[n]])) \\
  f_{P_P[n]} & \triangleq F_P[1][L_S[n]]
\end{align*}
\]

(a) \textbf{Proof 11}

\[
\begin{align*}
  I_2 & = \% \text{ Def. } I_1 \\
  & \left( \ell_{D_3} = 0 \land \ell_{P_1} = 0 \land \Lambda_i \in L_N \ L_S[D_3[i]] = i \land \Lambda_i \in P_N(M_P[i] = \text{ dfit} \land F[i] = i) \right) \\
  & \rightarrow \% \text{ Def. } \tilde{f}, \ p \\
  & \left( \ell_P = 0 \land \Lambda_i \in P_N(M_P[i] = \text{ dfit} \land F[i] = i) \land P \right) \left[ \tilde{f}/X_P \right] \\
  & = \% \text{ Def. } I_P \\
  & \left( I_P \land P \right) \left[ \tilde{f}/X_P \right]
\end{align*}
\]

(b) \textbf{Proof 12}

Since \( T_P \) is of the form \( \text{stut}_P \lor \forall \epsilon \left( \epsilon = a_r \land \text{trans}_r \right) \) then \( T_P \land ((P \land P') \lor \text{stut}_P) \) is equal to \( \text{stut}_P \lor \forall \epsilon \left( \epsilon = a_r \land \text{trans}_r \land P \land P' \right) \).

\[
\begin{align*}
  \tau_{2,1} & = \left( \epsilon = \text{ Rreq?}(n) \land \ell_{D_3} = 0 \land \ell_{P_1} = 0 \land \Psi_{12} = \Psi_{12}[1, n/\ell_{D_3}, r_{D_3}] \right) \\
  & \rightarrow \left( \epsilon = \text{ Rreq?}(n) \land \ell_P = 0 \land P \land P' \land \Psi_1 = \Psi_1[1, n/\ell_P, r_P] \right) \left[ \tilde{f}/X_P \right] \\
  & = \left( \tau_{P_1, P} \land P \land P' \right) \left[ \tilde{f}/X_P \right]
\end{align*}
\]

The user read request at the second level corresponds to the user read request at the first level.

\[
\begin{align*}
  \tau_{2,2} & = \left( \epsilon = i \land \ell_{D_3} = 1 \land \ell_{P_1} = 0 \land \Psi_{12} = \Psi_{12}[2, m/\ell_{D_3}, t_{P_1}, r_{P_1}] \right) \\
  & \rightarrow \left( \epsilon = i \land \Psi_1 = \Psi_1 \right) \left[ \tilde{f}/X_P \right] \\
  & \rightarrow \text{stut}_P \left[ \tilde{f}/X_P \right]
\end{align*}
\]

The read request to the physical disk at the second level corresponds to stutter step of the physical disk at the first level.

\[
\begin{align*}
  \tau_{2,3} & = \left( \epsilon = i \land \ell_{D_3} = 2 \land \ell_{P_1} = 1 \land \Psi_{12} = \Psi_{12}[3, c, 0/\ell_{D_3}, s_{D_3}, t_{P_1}] \right) \\
  & \rightarrow \left( \epsilon = i \land \Psi_1 = \Psi_1 \right) \left[ \tilde{f}/X_P \right] \\
  & \rightarrow \text{stut}_P \left[ \tilde{f}/X_P \right]
\end{align*}
\]

The read response of the physical disk at the second level corresponds to the stutter step of the physical disk at the first level.
4.5 Third Step: Fail-Stop Detection Layer

\[ \tau_{2.4} \]
\[ = \left( \epsilon = \text{Rres}(s_{D_3}) \land \ell_{D_3} = 3 \land \ell_{P_1} = 0 \land \Psi_{12} = \Psi_{12}[0/\ell_{D_3}] \right) \]
\[ \rightarrow \left( \epsilon = \text{Rres}(M_{P}[F[r_{P}]]) \land \ell_{P} = 1 \land p \land p' \land \Psi_{1} = \Psi_{1}[0/\ell_{P}] \right) \left[ \bar{f}/X_{P} \right] \]
\[ = (\tau_{P,2} \land p \land p') \left[ \bar{f}/X_{P} \right] \]

The read response of the detection layer at the second level corresponds to the read response of the physical disk at the first level.

\[ \tau_{2.5} \]
\[ = \left( \epsilon = \text{Wreq}(n, d) \land (\ell_{D_3}, \ell_{P_1}) = (0, 0) \right) \]
\[ \land \Psi_{12} = \Psi_{12}[6, n, d/\ell_{D_3}, r_{D_3}, s_{D_3}] \]
\[ \rightarrow \left( \epsilon = \text{Wreq}(n, d) \land \ell_{P} = 0 \land p \land p' \right) \]
\[ \land \Psi_{1} = \Psi_{1}[2, n, d/\ell_{P}, r_{P}, s_{P}] \left[ \bar{f}/X_{P} \right] \]
\[ = (\tau_{P,3} \land p \land p') \left[ \bar{f}/X_{P} \right] \]

The user write request at the second level corresponds to the user write request at the first level.

\[ \tau_{2.6} \]
\[ = \left( \epsilon = i \land \ell_{D_3} = 6 \land \ell_{P_1} = 0 \land \Psi_{12} = \Psi_{12}[7, 2, m, d/d_{D_3}, \ell_{P_1}, r_{P_1}, s_{P_1}] \right) \]
\[ \rightarrow \left( \epsilon = i \land \Psi_{1} = \Psi_{1} \right) \left[ \bar{f}/X_{P} \right] \]
\[ \rightarrow \text{stut}_{P} \left[ \bar{f}/X_{P} \right] \]

The write request to the physical disk at the second level corresponds to the stutter step of the physical disk at the first level.

\[ \tau_{2.7} \]
\[ = \left( \epsilon = i \land \ell_{D_3} = 7 \land \ell_{P_1} = 2 \land \Psi_{12} = \Psi_{12}[8, 0/\ell_{D_3}, \ell_{P_1}] \right) \]
\[ \rightarrow \left( \epsilon = i \land \Psi_{1} = \Psi_{1} \right) \left[ \bar{f}/X_{P} \right] \]
\[ \rightarrow \text{stut}_{P} \left[ \bar{f}/X_{P} \right] \]

The write response of the physical disk at the second level corresponds to the stutter step of the physical disk at the first level.

\[ \tau_{2.8} \]
\[ = \left( \epsilon = \text{Wres}! \land \ell_{D_3} = 8 \land \ell_{P_1} = 0 \land \Psi_{12} = \Psi_{12}[0/\ell_{D_3}] \right) \]
\[ \rightarrow \left( \epsilon = \text{Wres}! \land \ell_{P} = 2 \land p \land p' \land \Psi_{1} = \Psi_{1}[0, s_{P}/\ell_{P}, M_{P}[F[r_{P}]]] \right) \left[ \bar{f}/X_{P} \right] \]
\[ = (\tau_{P,4} \land p \land p') \left[ \bar{f}/X_{P} \right] \]

The write response of the physical detection layer at the second level corresponds to the write response of the physical disk at the first level.

\[ \text{stut}_{2} \rightarrow \text{stut}_{P} \left[ \bar{f}/X_{P} \right] \]

(c) Let \( SF_{P} = \{ \tau_{P,i} \mid i \in \{1, 2, 3, 4\} \} \) then

\[ L_{P} \overset{\Delta}{=} \bigwedge_{\tau \in SF_{P}} \left( \Box E n(\tau) \rightarrow \Box \tau \right) \]
. Let $SF_i = \{\tau_i \mid i \in \{1, \ldots, 8\}\}$ then

\[
L_2 \triangleq \bigwedge_{\tau \in SF_2} (\Box \Diamond En(\tau) \rightarrow \Box \Diamond \tau).
\]

Then the following holds

\[
S_2 \models L_2 \rightarrow L_P \left[ \overline{f}/X_P \right]
\]

So

\[
S_2 \models L_P \left[ \overline{f}/X_P \right]
\]

holds.

4.6 Fourth Step: Error Recovery Layer

4.6.1 Introduction

In this step the error recovery layer is specified. This is the layer that tries to correct the errors detected by the detection layer. The technique used for error recovery is that of the \textit{mirrored disk concept} [LA90]. This mirror disk concept is as follows: instead of one physical disk and corresponding detection layer $N$ physical disks with identical contents and $N$ corresponding detection layers ($N > 1$) are maintained. In case some information can no longer be retrieved from a disk, the information is still available on another one. The user requests some contents from the error recovery layer. The error recovery layer selects a disk from which it can retrieve these contents. Then it requests these contents from the corresponding detection layer of that disk. The detection layer requests then the contents from the physical disk and checks whether the contents are correct. The detection layer then signals if the contents are correct and if not it will signal which error has it has detected. If the contents are correct the error recovery layer will send them to the user and is then ready for new requests from the user. As seen before the detection layer can detect two kinds of errors: (1) errors due to disk surface damage fault and (2) errors due to disk control system faults. The error recovery layer will react as follows on these errors:

\textbf{ad (1)} First, the error recovery layer selects another disk from which it can retrieve the requested contents and when the corresponding detection layer signals that the contents are correct, the error recovery layer will write these contents to another location of the affected disk. In order to retrieve these contents from this new location \textit{logical} locations are introduced. When contents are stored at a new physical location the logical location will be pointing to this new location. So actually the data are retrieved from their logical location. Subsequently the error recovery layer will send the contents to the user and is ready to receive new requests from the user. When the detection-layer of the second disk also reports an error the error recovery layer will react as described in \textit{ad(1)} and \textit{ad(2)} depending on the kind of error detected.
4.6 Fourth Step: Error Recovery Layer

ad (2) First, the error recovery layer disables the faulty disk and then it will select another disk from which it can retrieve the requested contents and when the corresponding detection layer signals that the contents are correct the error recovery layer will them to the user. When the detection-layer of the second disk also reports an error the error recovery layer will react as described in ad(1) and ad(2) depending on the kind of error detected.

This error recovery process only works if the following assumptions are made:

- In order to store the contents on a new physical location enough spare locations should be available on an affected disk.

- Furthermore, the following must always hold in order to recover the ad(1)-type of error on a disk or to retrieve the contents from a logical location: for all logical locations there exists at least one non-disabled physical disk that has correct data stored on that logical location. This condition guarantees that always, each logical location contains correct data (on which disk we don't know, but it is a non-disabled one and it is not the disk whose type 1 error has to be repaired).

4.6.2 Specification of the Recovery Layer

The error recovery layer acts as interface between the user and the N detection layers of the N physical disks. The user requests with a \texttt{Rreq}(n) event the contents of some logical sector \(n\). The error recovery layer requests these contents, on receipt of this event, by issuing a \texttt{Rreqd}_j(n) event to one of the non-disabled detection layers. This detection layer responds with an \texttt{Rresd}_i(d) event. As seen in the third step there are three possibilities:

1. If this event delivers a message saying that the, to this detection layer corresponding, physical disk has been affected by a disk control system fault then this detection layer will be disabled and the error-recovery layer will send a \texttt{Rreqd}_j(n) event to another non-disabled detection layer.

2. If this event delivers a message that the, to this detection layer corresponding, physical disk has been affected by a disk surface damage fault then the error recovery layer requests the contents with a \texttt{Rreqd}_j(n) from another non-disabled detection layer until it finds a detection layer that responds with the correct contents. Then the error recovery layer can "repair" the physical disks that has been affected by a disk surface damage fault at the same logical sector by generating a \texttt{Wreqd}_j write request event with the correct data to the same logical sector number of the corresponding detection layers of those physical disks. The detections layers will respond with a \texttt{Wresd} indicating that the affected physical disks has been repaired. The design decision we make is that the detection layer has to find the spare physical sector to which these contents can be written. After that, the error recovery layer responds with a \texttt{Rres}(c) event to deliver the requested contents.

3. If this event delivers normal data the error recovery layer will respond with a \texttt{Rres}(c) event delivering the requested contents.
The user requests with a \textbf{Wreq}(n, d) event that \(d\) has to be written onto a logical sector \(n\). The error recovery layer requests with a \textbf{Wreqd}(n, d) event to all non-disabled detection layers that \(d\) has to be written on logical sector \(n\) to ensure that the corresponding physical disks have identical contents on their logical sectors. The detection layers then respond to these requests with a \textbf{Wresd} event. The error recovery layer then responds with a \textbf{Wres} event to the user that the write operation has been performed.

The stable storage layer is described by the following specification: \(S_R = (B_R, H_R)\) where \(H_R = I_R \land \square T_R \land L_R\) and \(B_R, I_R, T_R\) and \(L_R\) are as follows:

1. **Basis** \(B_R = ((\text{In}_R, \text{Out}_R), (V_R, X_R))\)

   \[
   \begin{align*}
   \text{In}_R &\triangleq \{\text{Rreq}, \text{Wreq}, \text{Rresd}_i, \text{Wresd} ; i \in N d\}, \\
   \text{Out}_R &\triangleq \{\text{Rres}, \text{Wres}, \text{Wreqd}_i, \text{Wresd}_i ; i \in N d\}, \\
   V_R &\triangleq 0, \\
   X_R &\triangleq \{\ell_R, r_R, s_R, t_R, G, A, W\}
   \end{align*}
   \]

- \textbf{Rreq}?(n): the request from the user to read logical sector \(n\).
- \textbf{Rresd}!(c): the response of the error recovery layer to the previous request where \(c\) are the crc-decoded and address-decoded contents of the requested logical sector.
- \textbf{Wreq}?(n, d): user request to write information item \(d\) onto logical sector \(n\).
- \textbf{Wres}!: write response to the user that the requested information is written.
- \textbf{Rreqd}!(n): the read request from the error recovery layer towards detection layer \(i\).
- \textbf{Rresd}?!(c): the read response from detection layer \(i\) to the previous request where \(c\) are the contents of the requested logical sector.
- \textbf{Wreqd}!(n, d): the write request from the error recovery layer to detection layer \(i\) to write information item \(d\) onto logical sector \(n\).
- \textbf{Wresd}?!: response from detection layer \(i\) to the error recovery layer that the requested information has been written.
- \(\ell_R\): local variable indicating the status of the error recovery layer; \(\ell_R = 0\): the error recovery layer is waiting for a request, \(\ell_R = 1\): the user has issued a read request or the detection layer responded to a read request with affected data, \(\ell_R = 2\): the error recovery layer has issued a read request, \(\ell_R = 3\): the detection responded to a read request with correct data or all affected disk are repaired, \(\ell_R = 4\): the the detection responded to a read request with correct data and there are affected disks, \(\ell_R = 5\): the error recovery layer has issued a write request to repair an affected disk and there are still affected disks to be repaired, \(\ell_R = 6\): the error recovery layer has issued a write request to repair an affected disk and there are no more affected disks, \(\ell_R = 7\): the user has issued a write request, \(\ell_R = 8\): the error recovery layer has issued a write request and there are still to be written disks, \(\ell_R = 9\): the error recovery layer has issued a write request and there are no more to be written disks, \(\ell_R = 10\): the detection layer of the last to be written disk responded to a write request.
4.6 Fourth Step: Error Recovery Layer

- $r_R$: local variable indicating the requested sector.
- $s_R$: local variable indicating the requested contents or the requested contents to be written.
- $t_R$: local variable indicating the index of the disk to which a request has been issued.
- $G$: local variable indicating the set of indexes of non-disabled disks.
- $A$: local variable indicating the set of indexes of by control system faults affected disks.
- $W$: local variable indicating the set of indexes on which data should be written.

Let $\Psi_3 \triangleq (\ell_R, r_R, s_R, t_R, G, A, W)$ and $\Psi'_3 \triangleq (\ell'_R, r'_R, s'_R, t_R, G', A', W')$.

2. Initial States:

$$ I_R \triangleq \ell_R = 0 \land G = \{1, \ldots, N\} \land A = \emptyset $$

The error recovery layer is waiting for requests from the user and all the $N$ disks are non-disabled.

3. Transitions:

Let

- $c_1 \triangleq$ address error
  address error message
- $c_2 \triangleq$ crc error
  crc error message
- $Good1 \triangleq \ell = r_R \land d_c \neq c_1 \land \ell \neq c_2 \land A = \emptyset$
  data is not affected by faults and the number of affected disks is zero
- $Good2 \triangleq \ell = r_R \land d_c \neq c_1 \land d_c \neq c_2 \land A \neq \emptyset$
  data is not affected by faults and the number of affected disks is non-zero
- $A.er \triangleq i = t_R \land d_c = c_1$
  data is affected by control system fault
- $C.er \triangleq i = t_R \land d_c = c_2$
  data is affected by disk surface damage
- $G^- \triangleq G \setminus \{i\}$
  set of good disks minus $i$
- $A^- \triangleq A \setminus \{i\}$
  set of affected disks minus $i$
- $A^+ \triangleq A \cup \{i\}$
  set of affected disks plus $i$
- $W^- \triangleq W \setminus \{i\}$
  set of to be written disks minus $i$
\[ T_R \triangleq \]

\[ \tau_{R,1} \quad (\epsilon = \mathbf{Rreq?}(n) \land \ell_R = 0 \land \Psi_3' = \Psi_3[1, n/\ell_R, r_R]) \]

The user requests the contents of logical sector \( n \).

\[ \tau_{R,2} \quad \lor (\epsilon = \mathbf{Rreq!}_i(\ell_R) \land \ell_R = 1 \land C1 \land \Psi_3 = \Psi_3[2, i/\ell_R, r_i]) \]

The error recovery layer requests the contents of logical sector \( r_R \) from an enabled detection layer.

\[ \tau_{R,3} \quad \lor (\epsilon = \mathbf{Rresd?}_i(cd) \land \ell_R = 2 \land \text{Good1} \land \Psi_3 = \Psi_3[3, i/\ell_R, s_i]) \]

The detection layer responds with the contents of the requested sector and the detection layer has detected no error in them.

\[ \tau_{R,4} \quad \lor (\epsilon = \mathbf{Rresd?}_i(cd) \land \ell_R = 2 \land \text{A.err} \land \Psi_3' = \Psi_3[1, G^-/\ell_R, G]) \]

The detection layer responds with the contents of the requested sector and the detection layer has detected a control system error, so this detection layer will be disabled.

\[ \tau_{R,5} \quad \lor (\epsilon = \mathbf{Rresd?}_i(cd) \land \ell_R = 2 \land \text{C.err} \land \Psi_3' = \Psi_3[1, A^+/\ell_R, A]) \]

The detection layer responds with the contents of the requested sector and the detection layer detects an disk surface damage error, so disk \( i \) has to be repaired.

\[ \tau_{R,6} \quad \lor (\epsilon = \mathbf{Rresd?}_i(cd) \land \ell_R = 2 \land \text{Good2} \land \Psi_3' = \Psi_3[4, cd/\ell_R, s_i]) \]

A correct disk has been found so the error recovery layer can repair the affected disks.

\[ \tau_{R,7} \quad \lor (\epsilon = \mathbf{Wreq!}_i(\ell_R, s_i) \land \ell_R = 4 \land C2 \land \Psi_3'[5, i, A^-/\ell_R, A]) \]

An affected disk is being repaired and there are still un repaired disk.

\[ \tau_{R,8} \quad \lor (\epsilon = \mathbf{Wresd?}_i \land \ell_R = 5 \land i = t_R \land \Psi_3' = \Psi_3[4/\ell_R]) \]

The affected disk is repaired.

\[ \tau_{R,9} \quad \lor (\epsilon = \mathbf{Wreq!}_i(\ell_R, s_i) \land \ell_R = 4 \land C3 \land \Psi_3'[6, A^-/\ell_R, A]) \]

An affected disk is being repaired and there are no un repaired disks.

\[ \tau_{R,10} \quad \lor (\epsilon = \mathbf{Wresd?}_i \land \ell_R = 6 \land i = t_R \land \Psi_3' = \Psi_3[3/\ell_R]) \]

All affected disk are repaired, so the user requested contents can be sent.
4.6 Fourth Step: Error Recovery Layer

The error recovery layer responds with the requested contents.

\[ \tau_{R,11} \lor (\epsilon = \text{Res}_! (s_R) \land \ell_R = 3 \land \Psi_3 = \Psi_3 [0/\ell_R]) \]

The error recovery layer responds with the requested contents.

\[ \tau_{R,12} \lor (\epsilon = \text{Wreq}! (n, d) \land \ell_R = 0 \land \Psi_3 = \Psi_3 [7, n, d, G/\ell_R, t_R, s_R, W]) \]

The user requests that \( d \) should be written on to logical sector \( n \).

\[ \tau_{R,13} \lor (\epsilon = \text{Wreqd}! (t_R, s_R) \land \ell_R = 7 \land C4 \land \Psi_3 = \Psi_3 [9, i, W^-/\ell_R, t_R, W]) \]

The requested information is being written to a disk and there are still disks which haven't written them.

\[ \tau_{R,14} \lor (\epsilon = \text{Wres}! (n, d) \land \ell_R = 8 \land i = t_R \land \Psi_3 = \Psi_3 [7/\ell_R]) \]

The requested information is written onto disk \( i \).

\[ \tau_{R,15} \lor (\epsilon = \text{Wreqd}! (m_R, s_R) \land \ell_R = 7 \land C5 \land \Psi_3 = \Psi_3 [9, i, W^-/\ell_R, t_R, W]) \]

The requested information is being written to a disk and there are no disks which haven't written them.

\[ \tau_{R,16} \lor (\epsilon = \text{Wres}! (n, d) \land \ell_R = 9 \land i = t_R \land \Psi_3 = \Psi_3 [10/\ell_R]) \]

The requested information is written onto all disks.

\[ \tau_{R,17} \lor (\epsilon = \text{Wres}! (n, d) \land \ell_R = 10 \land \Psi_3 = \Psi_3 [0/\ell_R]) \]

The error recovery layer responds with a signal to the user that requested write is performed.

\[ \tau_{R,1} \lor \text{stat}_R \]

These transitions are illustrated in figure 4.5

4. Liveness Condition:
The liveness condition expresses that the communication transitions are strongly fair.

Let \( SF_R = \{ \tau_{R,i} : i \in \{1, \ldots, 17\} \} \) then

\[ L_R = \underset{\tau \in SF_R}{\bigwedge} (\square[\square \triangleleft E_{\ell}(\tau) \rightarrow \square \triangleleft \tau]) \]

4.6.3 Specification of the Detection Layer

The detection layer is nearly the same as the fail-stop detection layer the only difference is that when error due to a disk surface fault has been detected the detection layer waits for the corrective action to be undertaken, i.e., a write request of the correct data to the to be repaired logical sector. It therefore selects a spare physical sector and maps the logical sector to it. It then issues a write request to this new physical sector. The physical disk then responds to this write request. The detection layer responds that the disk has been repaired.

The detection layer is described more formally by the following specification: \( S_D = (B_D, H_D) \) where \( H_D = I_D \land [\square \triangleleft T_D \land L_D] \) and \( B_D, I_D, T_D \) and \( L_D \) are as follows:

119
Figure 4.5: Transitions of the error recovery layer.

1. Basis $B_D = (\langle \text{In}_D, \text{Out}_D \rangle, \langle \text{V}_D, \text{X}_D \rangle)$

$$\text{In}_D \triangleq \{ \text{Rreqd, Wreqd, Rresp, Wresp} \},$$
$$\text{Out}_D \triangleq \{ \text{Rresd, Wresd, Rreqp, Wreqp} \},$$
$$\text{V}_D \triangleq \emptyset,$$
$$\text{X}_D \triangleq \{ \ell_D, r_D, s_D, \text{LS}_D[i] \mid i \in LN \}$$

where $LN$ is the set of logical sector numbers: $\{1, ..., Y\}$. Let $Lg$ be the set of data items that the user wants to store on or retrieve from the physical disk and $Phy$ be the set of information items that can be stored on or retrieved from the physical disk (Note: an item from $Phy$ is an crc-encoded and address-encoded item of $Lg$.)

For $n \in LN$, $c, d \in Lg$, $m \in PN$ and $cd, dd \in Phy$:

- **Rreqd**($n$): the request from the user to read logical sector $n$.
- **Rresd**($c$): the response of the detection layer to the previous request where $c$ are the crc-decoded and address-decoded contents of the requested logical sector $n$.
- **Wreqd**($n, d$): write information item $d$ onto logical sector $n$.
- **Wresd**: response that previous write has been performed.
4.6 Fourth Step: Error Recovery Layer

- \( \text{Rreqp}(m) \): the request from the detection layer to read physical sector \( m \).
- \( \text{Rresp}(cd) \): the response of the physical disk to the previous request where \( c \) are the crc-encoded and address-encoded contents of requested physical.
- \( \text{Wreqp}(m, dd) \): write information item \( dd \) onto physical sector \( m \).
- \( \text{Wresp} \): response that previous write has been performed.

- \( \ell_D \): local variable indicating the status of the detection layer; \( \ell_D = 0 \): the detection layer is waiting for a request, \( \ell_D = 1 \): the user has issued a read request, \( \ell_D = 2 \): the detection layer has issued a read request, \( \ell_D = 3 \): the physical has responded to a read request with correct data, \( \ell_D = 4 \): the physical disk has responded to a read request with incorrect data, \( \ell_D = 5 \): the detection layer has responded to a read request with an address error message (stop status), \( \ell_D = 6 \): the user has issued a write request, \( \ell_D = 7 \): the detection layer has issued a write request, \( \ell_D = 8 \): the physical disk has responded to a write request, \( \ell_D = 9 \): the detection layer has responded to a read request with a crc error message (can be repaired), \( \ell_D = 9 \): the user has issued a write request in order to repair the corresponding disk.

- \( r_D \): local variable indicating the requested sector.
- \( s_D \): local variable indicating the requested information or the data to be written.
- \( \text{LS}_D[i] \): the physical sector mapped to logical sector

Let \( \Psi_2 \triangleq (\ell_D, r_D, s_D, \text{LS}_D[1], \ldots, \text{LS}_D[Y]) \) and
\( \Psi'_2 \triangleq (\ell'_D, r'_D, s'_D, \text{LS}'_D[1], \ldots, \text{LS}'_D[Y]) \).

2. Initial states:

\[ I_D \triangleq \ell_D = 0 \bigwedge_{i \in LN} \text{LS}_D[i] = i \]

3. Transitions:
The same detection mechanism as the fail-stop detection layer is used. Let \( spare \) be
a function that returns a spare physical sector number. Let

\[ \text{Good} \triangleq CC(cd) \land AC(CD(cd), LS_D[r_D]) \]

data has not been affected by faults

\[ \text{A.er} \triangleq CC(cd) \land \neg AC(CD(cd), LS_D[r_D]) \]

data has been affected by a control system fault

\[ \text{C.er} \triangleq \neg CC(cd) \]

data has been affected by a disk surface damage

\[ c \triangleq AD(CD(cd)) \]

the address- and crc-decoded contents

\[ m \triangleq LS_D[r_D] \]

physical sector

\[ dd \triangleq CE(AE(r_D, s_D)) \]

address- and crc-encoded contents

\[ x \triangleq \text{sparing} \]

spare physical sector

\[ c_1 \triangleq \text{address error} \]

address error message

\[ c_2 \triangleq \text{crc error} \]

crc error message

\[ T_D \triangleq \]

\[ \tau_{D,1} \quad (\epsilon = \text{Rreqd}(n) \land \ell_D = 0 \land \Psi_2 = \Psi_2[1, n/\ell_D, r_D]) \]

The user requests the contents of logical sector \( n \).

\[ \tau_{D,2} \lor (\epsilon = \text{Rreqp!}(m) \land \ell_D = 1 \land \Psi_2 = \Psi_2[2/\ell_D]) \]

The detection layer requests the to logical sector \( r_D \) mapped physical sector.

\[ \tau_{D,3} \lor (\epsilon = \text{Rresp?}(cd) \land \ell_D = 2 \land \text{Good} \land \Psi_2 = \Psi_2 [3, c_1/\ell_D, s_D]) \]

The physical disk responds with the contents of the requested sector and the detection layer detects no error in them.

\[ \tau_{D,4} \lor (\epsilon = \text{Rresp?}(cd) \land \ell_D = 2 \land \text{A.er} \land \Psi_2 = \Psi_2 [4, c_1/\ell_D, s_D]) \]

The physical disk responds with the contents of the requested sector and the detection layer detects an control system error.

\[ \tau_{D,5} \lor (\epsilon = \text{Rresp?}(cd) \land \ell_D = 2 \land \text{C.er} \land \Psi_2 = \Psi_2 [4, c_2/\ell_D, s_D]) \]

The physical disk responds with the contents of the requested sector and the detection layer detects an disk surface damage error.

\[ \tau_{D,6} \lor (\epsilon = \text{Rresd!}(s_D) \land \ell_D = 3 \land \Psi_2 = \Psi_2 [0/\ell_D]) \]

The detection layer responds with the contents of the user requested sector.

\[ \tau_{D,7} \lor (\epsilon = \text{Rresd!}(s_D) \land \ell_D = 4 \land s_D = c_1 \land \Psi_2 = \Psi_2 [5/\ell_D]) \]

In case of an address error the detection layer responds with the corresponding error message and then stops.
4.6 Fourth Step: Error Recovery Layer

\( \tau_{D,8} \vee (\epsilon = \text{Rresd}(s_D) \wedge \ell_D = 4 \wedge s_D = c_1 \wedge \Psi'_2 = \Psi_2[9/\ell_D]) \)

The detection layer responds with an error message and waits for the corrective action.

\( \tau_{D,9} \vee (\epsilon = \text{Wreqd}(n, d) \wedge \ell_D = 9 \wedge \Psi'_2 = \Psi_2[10, n, d, x/\ell_D, r_D, s_D, \text{LSD}[n]]) \)

The user requests that \( d \) should be written on a spare physical sector.

\( \tau_{D,10} \vee (\epsilon = \text{Wreqd}(m, d) \wedge \ell_D = 10 \wedge \Psi'_2 = \Psi_2[7/\ell_D]) \)

The detection layer requests that \( dd \) should be written onto physical sector \( m \).

\( \tau_{D,11} \vee (\epsilon = \text{Wreqd}(n, d) \wedge \ell_D = 0 \wedge \Psi'_2 = \Psi_2[6, n, d/\ell_D, r_D, s_D]) \)

The user requests that \( d \) should be written onto logical sector \( n \).

\( \tau_{D,12} \vee (\epsilon = \text{Wreqd}(m, d) \wedge \ell_D = 6 \wedge \Psi'_2 = \Psi_2[7/\ell_D]) \)

The detection layer requests that \( dd \) should be written onto physical sector \( m \).

\( \tau_{D,13} \vee (\epsilon = \text{Wresp}! \wedge \ell_D = 7 \wedge \Psi'_2 = \Psi_2[8/\ell_D]) \)

The physical disk responds with a signal to the detection layer that requested write is performed.

\( \tau_{D,14} \vee (\epsilon = \text{Wresd}! \wedge \ell_D = 8 \wedge \Psi'_2 = \Psi_2[0/\ell_D]) \)

The detection layer responds with a signal to the user that requested write is performed.

\( \tau_{D,0} \vee \text{stut}_{D} \)

These transitions are illustrated in figure 4.6

4. Liveness conditions:

The liveness condition expresses that the communication transitions are strongly fair. Let \( SF_D = \{\tau_{D,i} \mid i \in \{1, \ldots, 14\}\} \) then

\[
L_D = \bigwedge_{\tau \in SF_D} (\Box \Diamond En(\tau) \rightarrow \Box \Diamond \tau)
\]

4.6.4 Requirement \( W_R \)

The error recovery requirement should express that for all logical locations there exists at least one non-disabled disk that has correct data stored on that logical location and enough spare locations should be available on an affected disk.

\[
W_R \triangleq \Box(\forall i \in G, CC_i(M_{P_i}[\text{LSD}_i[n]]) \wedge AC_i(CD_i(M_{P_i}[\text{LSD}_i[n]], \text{LSD}_i[n])))
\]

This corresponds to the following machine: Let

\[
p_3 \triangleq (\forall i \in G, CC_i(M_{P_i}[\text{LSD}_i[n]]) \wedge AC_i(CD_i(M_{P_i}[\text{LSD}_i[n]], \text{LSD}_i[n])))
\]

\[
p'_3 \triangleq (\forall i \in G', CC_i(M'_{P_i}[\text{LSD}_i'[n]]) \wedge AC_i(CD_i(M'_{P_i}[\text{LSD}_i'[n]], \text{LSD}_i'[n])))
\]

then \( W_R \) is equal to the machine \( p_3 \land \Box((p_3 \land p'_3) \lor \text{stut}_3) \).
First we construct the system $\mathcal{S}_3 \triangleq \bigwedge_{i=1}^{N} (\mathcal{S}_{Di} \parallel \mathcal{S}_{Pi}) \parallel \mathcal{S}_R$ relatively refines $\mathcal{S}_{Ds} \parallel \mathcal{S}_P$ then according to theorem 8 $\mathcal{S}_3$ relatively refines $\mathcal{S}_2 \triangleq (\mathcal{S}_{Ds} \parallel \mathcal{S}_{P1})$ with respect to $(\mathcal{W}_R, \mathbf{true})$ iff the following holds:

$$\mathcal{O}(B_3) = \mathcal{O}(B_2) \text{ and } \models (\exists X_3. (G_3)) \rightarrow (\exists X_2. (G_2))$$

where $X_2$ are the local variables from $\mathcal{S}_{Ds} \parallel \mathcal{S}_{P1}$, i.e., $X_2 \triangleq \{\ell_{Ds}, r_{Ds}, s_{Ds}, LS_{Ds}[i] \mid i \in LN\} \cup \{\ell_{P1}, r_{P1}, s_{P1}, M_{P1}[n], F[n] \mid n \in PN\}$ and $G_2$ is defined as

$$\exists \epsilon_1, \epsilon_2, B_{\epsilon_{1,2}}^{A_1} \circ B_{\epsilon_{1,2}}^{A_1} (\epsilon, \epsilon_1, \epsilon_2) \land (H_{Ds} \land W_{Ds})[\epsilon_1/\epsilon] \land (H_{P1} \land W_{P1})[\epsilon_2/\epsilon]$$

This can be rewritten to following machine specification of $\mathcal{S}_2$: $\mathcal{S}_2 = (B_2, H_2)$ where $H_2 \triangleq I_2 \land \square T_2 \land L_2$ and $B_1, I_2, T_2$ and $L_2$ are defined in section 4.5. $X_3$ are the local variables from $\mathcal{S}_3$, i.e., $X_3 \triangleq \bigcup_{i=1}^{N} (\ell_{Dj}, r_{Dj}, s_{Dj}, LS_{Dj}[i] \mid i \in LN \} \cup \{\ell_{Pj}, r_{Pj}, s_{Pj}, M_{Pj}[n], F[n] \mid n \in PN\}) \cup \{\ell_{R}, r_{R}, s_{R}, r_{R}, G, A, W\}$. Let $\bar{\epsilon} \triangleq \epsilon_{1,1}, \ldots, \epsilon_{1,N}, \epsilon_{2,1}, \ldots, \epsilon_{2,N}, \epsilon_3$ and let $B_{\bar{\epsilon}}^{A_1} \triangleq B_{\ell_{D1}}^{A_1}, \ldots, B_{\ell_{DN}}^{A_1}, B_{\ell_{P1}}^{A_1}, \ldots, B_{\ell_{PN}}^{A_1}, B_{\ell_{R}}^{A_1}$ then $G_3$ is defined as

$$(\exists \bar{\epsilon}. \circ B_{\bar{\epsilon}}^{A_1} (\bar{\epsilon}, \bar{\epsilon})(\bigwedge_{j=1}^{N}(H_{Dj})) [\epsilon_{1,j}/\epsilon] \land (H_{Pj})[\epsilon_{2,j}/\epsilon] \land H_{R}[\epsilon_3/\epsilon] \land W_R$$

Figure 4.6: Transitions of the detection layer.
4.6 Fourth Step: Error Recovery Layer

The \((\exists e. \circ \mathcal{D}_4 (e, e)(\bigwedge_{j=1}^{N} (H_{Dj}[\epsilon_{1,j}] \land (H_{Pj}[\epsilon_{2,j}] \land H_{R}[\epsilon_{3,j}])) \land \mathcal{H}_3[e])\) part can be rewritten to following machine specification \(\mathcal{S}_3 = (B_3, H_3)\) where \(H_3 = I_3 \land T_3 \land I_3\) and \(B_3, I_3, T_3\) and \(L_3\) are as follows:

1. **Basis** \(B_3 = ((I_3, Out_3), (V_3, X_3))\)

   \[
   \begin{align*}
   I_3 & \triangleq \{ \text{Rreq, Wreq} \} \\
   Out_3 & \triangleq \{ \text{Rres, Wres} \}, \\
   V_3 & \triangleq \emptyset, \\
   X_3 & \triangleq \text{as above}
   \end{align*}
   \]

   Let

   \[
   \Psi_3 \triangleq \big((\ell_{Dj}, r_{Dj}, s_{Dj}, LS_{Dj}[1], \ldots, LS_{Dj}[Y], \\
   \ell_{Pj}, r_{Pj}, s_{Pj}, M_{Pj}[1], \ldots, M_{Pj}[Y], F_{j}[1], \ldots, F_{j}[Y])_{j=1,\ldots,N}, \\
   t_R, s_R, t_R, G, A, W)\]

   \[
   \Psi'_3 \triangleq \big((\ell'_{Dj}, r'_{Dj}, s'_{Dj}, LS'_{Dj}[1], \ldots, LS'_{Dj}[Y], \\
   \ell'_{Pj}, r'_{Pj}, s'_{Pj}, M'_{Pj}[1], \ldots, M'_{Pj}[Y], F'_{j}[1], \ldots, F'_{j}[Y])_{j=1,\ldots,N} \\
   t'_R, s'_R, t'_R, G', A', W')\]

2. **Initial states:**

   \[
   I_3 \triangleq \bigwedge_{j=1}^{N} (I_{Dj} \land I_{Pj}) \land I_R
   \]

3. **Transitions:**

   Let

   - \(c_1 \triangleq \text{address error}\)
     address error message
   - \(c_2 \triangleq \text{crc error}\)
     crc error message
   - \(\text{Good1} \triangleq i = t_R \land dc \neq c_1 \land dc \neq c_2 \land A = \emptyset\)
     data is not affected by faults and the number of affected disks is zero
   - \(\text{Good2} \triangleq i = t_R \land dc \neq c_1 \land dc \neq c_2 \land A \neq \emptyset\)
     data is not affected by faults and the number of affected disks is non-zero
   - \(\text{A.er} \triangleq i = t_R \land dc = c_1\)
     data is affected by control system fault
   - \(\text{C.er} \triangleq i = t_R \land dc = c_2\)
     data is affected by disk surface damage
   - \(G^- \triangleq G \setminus \{i\}\)
     set of good disks minus \(i\)
Stable Storage Example

- $A^- \triangleq A \setminus \{i\}$
  set of affected disks minus $i$
- $A^+ \triangleq A \cup \{i\}$
  set of affected disks plus $i$
- $W^- \triangleq W \setminus \{i\}$
  set of to be written disks minus $i$
- $C1 \triangleq i \in G \land i \notin A$
  disk $i$ is good and not affected
- $C2 \triangleq i \in A \land A^- \neq \emptyset$
  disk $i$ is affected and the number of affected disks is greater than 1
- $C3 \triangleq i \in A \land A^- = \emptyset$
  disk $i$ is the only affected disk
- $C4 \triangleq i \in W \land W^- \neq \emptyset$
  disk $i$ should be written onto and the number of to be written disks is greater than 1
- $C5 \triangleq i \in W \land W^- = \emptyset$
  disk $i$ is the only disk to be written onto
- $q \triangleq (\ell_{Di}, \ell_{Pi}, \ell_R)$
  status of detection layer $i$ and physical disk $i$ and the error recovery layer.

$T_3 \triangleq$

$\tau_{3,1} \quad (\epsilon = Rreq?(n) \land \ell_R = 0 \land \Psi_3 = \Psi_3[1, n/\ell_R, r_R])$

The user requests the contents of logical sector $n$.

$\tau_{3,2} \quad \lor \ (\epsilon = i \land \ell_R = 1 \land C1 \land \Psi_5 = \Psi_3[2, i, 1, r_R/\ell_R, v_R, \ell_{Di}, \ell_{Di}])$

The error recovery layer requests the contents of logical sector $r_R$ from an enabled detection layer $i$.

$\tau_{3,3} \quad \lor \ (\epsilon = i \land q = (1, 0, 2) \land \Psi_3 = \Psi_3[2, 1, m/\ell_{Di}, \ell_{Di}, r_{Di}, r_{Di}])$

The detection layer $i$ requests the to logical sector $r_{Di}$ mapped physical sector from physical disk $i$.

$\tau_{3,4} \quad \lor \ (\epsilon = i \land q = (2, 1, 2) \land \text{Good} \land \Psi_5 = \Psi_3[3, c, 0/\ell_{Di}, s_{Di}, \ell_{Di}, \ell_{Di}])$

The physical disk $i$ responds with the contents of the requested sector and the detection layer $i$ detects no error in them.

$\tau_{3,5} \quad \lor \ (\epsilon = i \land q = (2, 1, 2) \land A.err \land \Psi_5 = \Psi_3[4, c_1, 0/\ell_{Di}, s_{Di}, \ell_{Di}, \ell_{Di}])$

The physical disk $i$ responds with the contents of the requested sector and the detection layer $i$ detects a control system error.

$\tau_{3,6} \quad \lor \ (\epsilon = i \land q = (2, 1, 2) \land C.err \land \Psi_2 = \Psi_2[4, c_2, 0/\ell_{Di}, s_{Di}, \ell_{Di}, \ell_{Di}])$

The physical disk $i$ responds with the contents of the requested sector and the detection layer $i$ detects a disk surface damage error.
4.6 Fourth Step: Error Recovery Layer

\( \forall \left( \epsilon = i \land q = (3, 0, 2) \land Good 1 \land \Psi_3 = \Psi_3[0, 3, s_{Di}/\ell_{Di}, \ell_R, s_R] \right) \)

The detection layer \( i \) responds with the contents of the requested sector and the detection layer \( i \) has detected no error in them.

\( \forall \left( \epsilon = i \land q = (4, 0, 2) \land A.err; \land \Psi_3 = \Psi_3[5, 1, G^-/\ell_{Di}, \ell_R, G] \right) \)

The detection layer \( i \) responds with the contents of the requested sector and the detection layer \( i \) has detected an control system error, so this detection layer will be disabled.

\( \forall \left( \epsilon = i \land q = (4, 0, 2) \land C.err; \land \Psi_3 = \Psi_3[9, 1, A^+/\ell_{Di}, \ell_R, A] \right) \)

The detection layer \( i \) responds with the contents of the requested sector and the detection layer \( i \) detects an disk surface damage error, so physical disk \( i \) has to be repaired.

\( \forall \left( \epsilon = i \land q = (3, 0, 2) \land Good 2 \land \Psi_3 = \Psi_3[0, 4, s_{Di}/\ell_{Di}, \ell_R, s_R] \right) \)

A correct disk \( i \) has been found so the error recovery layer can repair the affected disks.

\( \forall \left( \epsilon = i \land q = (9, 0, 4) \land C2 \land \Psi_3 = \Psi_3[10, r_R, s_R, x_i, 5, i, A^-/\ell_{Di}, r_{Di}, s_{Di}, LS_{Di}, \ell_R, t_R, A] \right) \)

An affected disk \( i \) is being repaired and there are still unrepaired disks.

\( \forall \left( \epsilon = i \land q = (10, 0, 5) \land \Psi_3 = \Psi_3[7, 2, m_i, dd_i/\ell_{Di}, \ell_{Pi}, r_{Pi}, s_{Pi}] \right) \)

An affected disk \( i \) is being repaired and there are still unrepaired disks.

\( \forall \left( \epsilon = i \land q = (7, 2, 5) \land \Psi_3 = \Psi_3[8, 0, s_{Pi}/\ell_{Di}, \ell_{Pi}, M_{Pi}[F_{Pi}[r_{Pi}]]] \right) \)

An affected disk \( i \) is being repaired and there are still unrepaired disks.

\( \forall \left( \epsilon = i \land q = (8, 0, 5) \land i = t_R \land \Psi_3 = \Psi_3[0, 4/\ell_{Di}, \ell_R] \right) \)

The affected disk is repaired.

\( \forall \left( \epsilon = i \land q = (9, 0, 4) \land C3 \land \Psi_3 = \Psi_3[10, 6, r_R, s_R, x_i, A^-/\ell_{Di}, r_{Di}, s_{Di}, LS_{Di}, \ell_R, A] \right) \)

An affected disk is being repaired and there are no further unrepaired disks.

\( \forall \left( \epsilon = i \land q = (10, 0, 6) \land \Psi_3 = \Psi_3[7, 2, m_i, dd_i/\ell_{Di}, \ell_{Pi}, r_{Pi}, s_{Pi}] \right) \)

An affected disk \( i \) is being repaired and there are no further unrepaired disks.

\( \forall \left( \epsilon = i \land q = (7, 2, 6) \land \Psi_3 = \Psi_3[8, 0, s_{Pi}/\ell_{Di}, \ell_{Pi}, M_{Pi}[F_{Pi}[r_{Pi}]]] \right) \)

An affected disk \( i \) is being repaired and there are no further unrepaired disks.

\( \forall \left( \epsilon = i \land q = (8, 0, 6) \land i = t_R \land \Psi_3 = \Psi_3[0, 3/\ell_{Di}, \ell_R] \right) \)

All affected disk are repaired, so the user requested contents can be sent.

\( \forall \left( \epsilon = Res!(s_{Pi}) \land \ell_R = 3 \land \Psi_3 = \Psi_3[0, \ell_R] \right) \)

The error recovery layer responds with the requested contents.

\( \forall \left( \epsilon = Wreq?(n, d) \land \ell_R = 0 \land \Psi_3 = \Psi_3[7, n, d, G/\ell_{Di}, r_{Di}, s_{Di}, W] \right) \)

The user requests that \( d \) should be written onto logical sector \( n \).

\( \forall \left( \epsilon = i \land q = (0, 0, 7) \land C4 \land \Psi_3 = \Psi_3[8, 6, r_R, s_R, i, W^-/\ell_{Di}, r_{Di}, s_{di}, t_R, W] \right) \)

The requested information is being written to a disk and there are still unwritten disks.
Stable Storage Example

\( \tau_{3,12} \quad \forall \left( \epsilon = i \land q = (6, 0, 8) \land \Psi_3 = \Psi_3 [7, 2, m_1, dd_i / \ell_D, \ell_P, r_P, s_P] \right) \)

The requested information is being written to a disk and there are still unwritten disks.

\( \tau_{3,23} \quad \forall \left( \epsilon = i \land q = (7, 2, 8) \land \Psi_3 = \Psi_3 [8, 0, s_P / \ell_D, \ell_P, M_P [F_P [r_P]]] \right) \)

The requested information is being written to a disk and there are still unwritten disks.

\( \tau_{3,24} \quad \forall \left( \epsilon = i \land q = (8, 0, 8) \land i = t_R \land \Psi_3 = \Psi_3 [0, 7 / \ell_D, \ell_R] \right) \)

The requested information is written onto disk \( i \).

\( \tau_{3,25} \quad \forall \left( \epsilon = i \land q = (0, 0, 7) \land C 5 \wedge \Psi_3 = \Psi_3 [9, 6, r_R, s_R, i, W^-, / \ell_R, / \ell_D, r_D, s_D, t_R, W] \right) \)

The requested information is being written to a disk and there are no further unwritten disks.

\( \tau_{3,26} \quad \forall \left( \epsilon = i \land q = (6, 0, 9) \land \Psi_3 = \Psi_3 [7, 2, m_1, dd_i / \ell_D, \ell_P, r_P, s_P] \right) \)

The requested information is being written to a disk and there are no further unwritten disks.

\( \tau_{3,27} \quad \forall \left( \epsilon = i \land q = (7, 2, 9) \land \Psi_3 = \Psi_3 [8, 0, s_P / \ell_D, \ell_P, M_P [F_P [r_P]]] \right) \)

The requested information is being written to a disk and there are no further unwritten disks.

\( \tau_{3,28} \quad \forall \left( \epsilon = i \land q = (8, 0, 9) \land i = t_R \land \Psi_3 = \Psi_3 [0, 10 / \ell_D, \ell_R] \right) \)

The requested information is written onto all disks.

\( \tau_{3,29} \quad \forall \left( \epsilon = \text{Wres}! \land \ell_R = 10 \land \Psi_3 = \Psi_3 [0 / \ell_R] \right) \)

The error recovery layer responds with a signal to the user that requested write is performed.

\( \tau_{3,0} \quad \forall \text{stut}_3 \)

These transitions are illustrated in figure 4.7 with the transitions for the physical disk omitted.

4. Liveness Condition:
The liveness condition expresses that all non-stuttering transitions are strongly fair. Let \( SF_3 = \{ \tau_{3,i} \mid i \in \{ 1, \ldots, 29 \} \} \) then

\[
I_3 = \bigwedge_{\tau \in SF_3} (\Box \Diamond En(\tau) \rightarrow \Box \Diamond \tau)
\]

Rule 3 will be used to prove

\[
\models (\exists X_3 \cdot (G_3)) \rightarrow (\exists X_2 \cdot (G_2))
\]

128
This means one has to prove (a), (b) and (c) below, for \( f \) the refinement mapping from \( S_3 \) to \( S_1 \), defined as: \( f = f_{t_d}, f_{r_d}, f_{s_L}, f_{S_0, [m]}, \ldots, f_{LS, [v]}, f_{T_1, f_{P_1, f_{s_L}, f_{P_1}, f_{M_0, [v]}, f_{P_1, [v]}}} \) \((n \in SN, m \in LN)\). The refinement mappings are defined as:

\[
\begin{align*}
if & \quad \ell_R = 0 & then & \ell_R \\
& \quad \ell_R = 1 & then & \ell_R - 1 \\
& \quad \ell_R = 2 \wedge \ell_{P_1} = 0 & then & \ell_R - 2 \\
& \quad \ell_R = 2 \wedge \ell_{P_1} = 1 & then & \ell_R - 1 \\
& \quad \ell_R = 3 & then & \ell_R - 3 \\
& \quad \ell_R = 4 & then & \ell_R - 4 \\
& \quad \ell_R = 5 & then & \ell_R - 5 \\
& \quad \ell_R = 6 & then & \ell_R - 6 \\
& \quad \ell_R = 7 & then & \ell_R - 7 \\
& \quad \ell_R = 8 \wedge \ell_{P_1} = 0 & then & \ell_R - 8 \\
& \quad \ell_R = 8 \wedge \ell_{P_1} = 2 & then & \ell_R - 6 \\
& \quad \ell_R = 9 \wedge \ell_{P_1} = 0 & then & \ell_R - 9 \\
& \quad \ell_R = 9 \wedge \ell_{P_1} = 2 & then & \ell_R - 7 \\
& \quad \ell_R = 10 & then & \ell_R - 10
\end{align*}
\]
Stable Storage Example

\[ f_{load} \]

\[
\begin{align*}
\text{if} & \quad \ell_R = 0 & \text{then} & \ell_R \\
\text{if} & \quad \ell_R = 1 & \text{then} & \ell_R \\
\text{if} & \quad \ell_R = 2 \land \ell_{P_i} = 0 & \text{then} & \ell_R - 1 \\
\text{if} & \quad \ell_R = 2 \land \ell_{P_i} = 1 & \text{then} & \ell_R - 1 \\
\text{if} & \quad \ell_R = 3 & \text{then} & \ell_R \\
\text{if} & \quad \ell_R = 4 & \text{then} & \ell_R - 2 \\
\text{if} & \quad \ell_R = 5 & \text{then} & \ell_R - 3 \\
\text{if} & \quad \ell_R = 6 & \text{then} & \ell_R - 4 \\
\text{if} & \quad \ell_R = 7 & \text{then} & \ell_R - 1 \\
\text{if} & \quad \ell_R = 8 & \text{then} & \ell_R - 1 \\
\text{if} & \quad \ell_R = 9 & \text{then} & \ell_R - 2 \\
\text{if} & \quad \ell_R = 10 & \text{then} & \ell_R - 2
\end{align*}
\]

For \( i \in G \land i \not\in A \) (physical \( i \) is not affected by any fault)

\[
\begin{align*}
\ell_{P_i} & \overset{\Delta}{=} r_{P_i} \\
\ell_{S_i} & \overset{\Delta}{=} s_{P_i} \\
\ell_{M_{P_i}[n]} & \overset{\Delta}{=} M_{P_i}[n] \\
\ell_{F_{P_i}[n]} & \overset{\Delta}{=} F_{P_i}[n] \\
\ell_{D_i} & \overset{\Delta}{=} r_{D_i} \\
\ell_{S_D} & \overset{\Delta}{=} s_{D_i} \\
\ell_{L_{S_D}[m]} & \overset{\Delta}{=} L_{S_D}[m]
\end{align*}
\]

(a) \( S_3 \cap \text{Hist}(W_R) \models (I_3 \land p_3) \rightarrow (I_2) \left[ \bar{f}/X_2 \right] \)

(b) \( S_3 \cap \text{Hist}(W_R) \models (T_3 \land ((p_3 \land p'_3) \lor \text{stut}_3)) \rightarrow (T_2) \left[ \bar{f}/X_2 \right] \)

(c) \( S_3 \cap \text{Hist}(W_R) \models I_2 \left[ \bar{f}/X_2 \right] \)

(a) **Proof 13**

\[
I_3 \land p_3 \\
\rightarrow \text{Def. } I_3, p_3, \bar{f}, I_2 \\
(1_2) \left[ \bar{f}/X_2 \right]
\]

(b) **Proof 14**

Since \( T_3 \) is of the form \( \text{stut}_3 \lor \lor_\epsilon (\epsilon = a_\tau \land \text{trans}_\tau) \) then \( T_3 \land ((p_3 \land p'_3) \lor \text{stut}_3) \) is equal to \( \text{stut}_3 \lor \lor_\epsilon (\epsilon = a_\tau \land \text{trans}_\tau \land p_3 \land p'_3) \).

\[
\tau_{3,1} \land p_3 \land p'_3 \\
\rightarrow (\tau_{2,1}) \left[ \bar{f}/X_2 \right]
\]

The user read request at the third level corresponds to the user read request at the second level.
4.6 Fourth Step: Error Recovery Layer

\[ \tau_{3,1} \land p_3 \land p' \]
\[ \rightarrow \text{stut}_1 \left[ \bar{f}/X_2 \right] \]

The read request to the detection layer \( i \) at the third level corresponds to the stutter step at the second level.

\[ \tau_{3,3} \land p_3 \land p' \]
\[ \rightarrow (\tau_{2,1}) \left[ \bar{f}/X_2 \right] \]

The read request to physical disk \( i \) at the third level corresponds to the read request to the physical disk at the second level.

\[ \tau_{3,4} \land p_3 \land p' \]
\[ \rightarrow (\tau_{2,3}) \left[ \bar{f}/X_2 \right] \]

The read response of the physical disk \( i \) at the third level corresponds to the read response of the physical disk at the second level because no errors are detected.

\[ \tau_{3,5} \land p_3 \land p' \]
\[ \rightarrow \text{stut}_2 \left[ \bar{f}/X_2 \right] \]

The read response of the physical disk \( i \) at the third level corresponds to the stutter step at the second level because a control system error is detected.

\[ \tau_{3,6} \land p_3 \land p' \]
\[ \rightarrow \text{stut}_2 \left[ \bar{f}/X_2 \right] \]

The read response of the physical disk \( i \) at the third level corresponds to the stutter step at the second level because a disk surface error is detected.

\[ \tau_{3,7} \land p_3 \land p' \]
\[ \rightarrow \text{stut}_2 \left[ \bar{f}/X_2 \right] \]

The read response of the detection \( i \) at the third level corresponds to the stutter step at the second level.

\[ \tau_{3,8} \land p_3 \land p' \]
\[ \rightarrow \text{stut}_2 \left[ \bar{f}/X_2 \right] \]

The read response of the detection \( i \) at the third level corresponds to the stutter step at the second level.

\[ \tau_{3,9} \land p_3 \land p' \]
\[ \rightarrow \text{stut}_2 \left[ \bar{f}/X_2 \right] \]

The read response of the detection \( i \) at the third level corresponds to the stutter step at the second level.

\[ \tau_{3,10} \land p_3 \land p' \]
\[ \rightarrow \text{stut}_2 \left[ \bar{f}/X_2 \right] \]

The read response of the detection \( i \) at the third level corresponds to the stutter step at the second level.

\[ \tau_{3,j} \land p_3 \land p' \]
\[ \rightarrow \text{stut}_2 \left[ \bar{f}/X_2 \right] \]

For \( j = 11, \ldots, 18 \)
The correction step at the third level corresponds to the stutter step at the second level.

- \( \tau_{3,19} \land p_3 \land p'_3 \)
  \( \rightarrow (\tau_{2,4}) \left[ f/X_2 \right] \)

The read response to the user at the third level corresponds to read response to the user at the second level.

- \( \tau_{3,20} \land p_3 \land p'_3 \)
  \( \rightarrow (\tau_{2,4}) \left[ f/X_2 \right] \)

The write request of the user at the third level corresponds to the write request of the user at the second level.

- For \( j = 21, 25 \)
  \( \tau_{3,j} \land p_3 \land p'_3 \)
  \( \rightarrow \text{stut}_2 \left[ f/X_2 \right] \)

The write request to the detection layer \( i \) at the third level corresponds to the stutter step at the second level.

- For \( j = 22, 26 \)
  \( \tau_{3,j} \land p_3 \land p'_3 \)
  \( \rightarrow (\tau_{2,6}) \left[ f/X_2 \right] \)

The write request to the physical disk \( i \) at the third level corresponds to the write request to the physical disk at the second level.

- For \( j = 23, 27 \)
  \( \tau_{3,j} \land p_3 \land p'_3 \)
  \( \rightarrow (\tau_{2,7}) \left[ f/X_2 \right] \)

The write response of the physical disk \( i \) at the third level corresponds to the write response of the physical disk at the second level.

- For \( j = 24, 28 \)
  \( \tau_{3,j} \land p_3 \land p'_3 \)
  \( \rightarrow \text{stut}_2 \left[ f/X_2 \right] \)

The write response of the detection layer \( i \) at the third level corresponds to the stutter step at the second level.

- \( \tau_{3,29} \land p_3 \land p'_3 \)
  \( \rightarrow (\tau_{2,8}) \left[ f/X_2 \right] \)

The write response to the user at the third level corresponds to the write response to the user at the second level.

- \( \text{stut}_3 \rightarrow \text{stut}_2 \left[ f/X_2 \right] \)
4.6 Fourth Step: Error Recovery Layer

(c) Let \( SF_3 = \{ \tau_{3,i} \mid i \in \{1, \ldots, 29\} \} \) then

\[
L_3 \triangleq \bigwedge_{\tau \in SF_3} (\square \diamond En(\tau) \to \square \diamond \tau)
\]

. Let \( SF_2 = \{ \tau_{2,i} \mid i \in \{1, \ldots, 8\} \} \) then

\[
L_2 \triangleq \bigwedge_{\tau \in SF_2} (\square \diamond En(\tau) \to \square \diamond \tau).
\]

Then the following holds

\[
\mathcal{S}_3 \cap Hist(W_3) \models L_3 \to L_2 \left[ \bar{f}/X_3 \right]
\]

So

\[
\mathcal{S}_3 \cap Hist(W_3) \models L_2 \left[ \bar{f}/X_2 \right]
\]

holds.
Stable Storage Example
Bibliography


BIBLIOGRAPHY


BIBLIOGRAPHY


<table>
<thead>
<tr>
<th>Reference</th>
<th>Title and Authors</th>
</tr>
</thead>
</table>
Appendix A

Proofs of Dense Model Theorems

A.1 Proof of Theorem 1

Theorem 1 (Relationship between histories and infinite sequences)
Let $h \in \mathcal{H}/ \simeq_h$ then $(\text{sel}_i)_{i \geq 0} \in \text{SEQ}/ \simeq_s$ where $\text{sel}_i$ is as follows:
if $nn(h) < \infty$:

$$\begin{align*}
\text{sel}_{i+1} &= h(i) & 0 \leq i < nn(h) \\
\text{sel}_{2i} &= \lim_{t \rightarrow t_1} h(t) & 0 \leq i \leq nn(h) \\
\text{sel}_{2i+1} &= \lim_{k \rightarrow t_1} h(t) & k = nn(h) \land k \leq i \\
\text{sel}_{2i} &= \lim_{k \rightarrow t_1} h(t) & k = nn(h) \land k \leq i
\end{align*}$$

if $nn(h) = \infty$:

$$\begin{align*}
\text{sel}_{i+1} &= h(i) & 0 \leq i \\
\text{sel}_{2i} &= \lim_{t \rightarrow t_1} h(t) & 0 \leq i
\end{align*}$$

Let $\text{seq} = (\text{sel}_i)_{i \geq 0} \in \text{SEQ}/ \simeq_s$ then $h \in \mathcal{H}/ \simeq_h$ where $h$ is as follows:
if $ns(\text{seq}) < \infty$:

$$\begin{align*}
h(0) &= \text{sel}_0 \\
h(t) &= \text{sel}_{2i-1} & t \in \mathbb{N} \land 0 < t \leq ns(\text{seq}) \\
h(t) &= \text{sel}_{2i} & t \in \mathbb{N} \land t > ns(\text{seq}) \\
h(t) &= \text{sel}_{2i} & i < t < i + 1
\end{align*}$$

if $ns(\text{seq}) = \infty$:

$$\begin{align*}
h(0) &= \text{sel}_0 \\
h(t) &= \text{sel}_{2i-1} & t \in \mathbb{N} \\
h(t) &= \text{sel}_{2i} & i < t < i + 1
\end{align*}$$

Proof 15 Let $h \in \mathcal{H}/ \simeq_h$ then $h$ is of the form $h_1 \circ \text{di}(h_1)$ for some $h_1 \in \mathcal{H}$. According to Def. 10 $h$ is then of the form that at discrete points the non-stutter steps and at all other points the stutter steps occur. The construction of seq above is such that at odd points the
non-stutter steps (or λ-steps if number of non-stutter steps is finite) and at even points
the λ-step occur, i.e., a sequence from $\text{SEQ/} \simeq_s$.

Let $\text{seq} = (\text{seq}_i)_{i \geq 0} \in \text{SEQ/} \simeq_s$ then, according to Def. 11, seq is such that at odd
points non-stutter steps occur (or λ-steps if the number of non-stutter step is finite) and
at the even points λ steps. The construction of h above is such that at discrete points
greater than zero the non-stutter steps occur (or λ steps if the number of non-stutter steps
is finite) and at all other points the λ steps, i.e. a history from $\mathcal{H}/ \simeq_h$.

A.2 Proof of Lemma 1

Lemma 1

Given machine $M = (B, I, T)$ then

$\text{Comp}(M)$ is a safety set.

Proof 16 One has to prove that $\text{Comp}(M)$ is closed, i.e., $\mathcal{H} \setminus \text{Comp}(M)$ is an open set,
i.e., $\mathcal{H} \setminus \text{Comp}(M) \in \tau_d$ ($\tau_d$ is the topological space defined in Def. 14).

\begin{align*}
\mathcal{H} \setminus \text{Comp}(M) &\in \tau_d \\
&\quad \% \text{ Def. 14 } \tau_d \\
&\forall h : \exists \varepsilon > 0 : \forall h_1 : \\
&\quad (h \in \mathcal{H} \setminus \text{Comp}(M) \land d(h, h_1) < \varepsilon) \rightarrow h_1 \in \mathcal{H} \setminus \text{Comp}(M) \\
&\quad \% \text{ Contraposition} \\
&\forall h : \exists \varepsilon > 0 : \forall h_1 \in \text{Comp}(M) \land d(h, h_1) < \varepsilon) \rightarrow h \in \text{Comp}(M) \\
&\quad \% \text{ Def. 14 } d(h, h_1) \\
&\forall h : \forall t : \forall h_1 : (h \setminus t = h_1 \setminus t \land h_1 \in \text{Comp}(M)) \rightarrow h \in \text{Comp}(M) \\
&\quad \% \text{ Pred. Calc.} \\
&\forall h : \forall h_1 : (h = h_1 \land h_1 \in \text{Comp}(M)) \rightarrow h \in \text{Comp}(M) \\
&\quad \% \text{ Pred. Calc.} \\
&\text{true}
\end{align*}

A.3 Proof of Theorem 2

Theorem 2

Let $\text{rexp}$ be a rigid expression, $\exp$ be an expression, $\text{evexp}$ an event expression and $p$ a

a $\forall t, h_0, h_1 : h_0 \simeq_{\delta h} h_1 \rightarrow ((h_0, t) \models \text{rexp} = (h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) \models \text{rexp})$
b $\forall t, h_0, h_1 : h_0 \simeq_{\delta h} h_1 \rightarrow ((h_0, t) \models \exp = (h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) \models \exp)$
c $\forall t, h_0, h_1 : h_0 \simeq_{\delta h} h_1 \rightarrow ((h_0, t) \models \text{evexp} = (h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) \models \text{evexp})$
d $\forall t, h_0, h_1 : h_0 \simeq_{\delta h} h_1 \rightarrow ((h_0, t) \models p \iff (h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) = p)$
A.3 Proof of Theorem 2

Proof 17

\[ a \quad \forall t, h_0, h_1 : h_0 \simeq_{\delta_t} h_1 \rightarrow ((h_0, t) \models \text{rexp} = (h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) \models \text{rexp}) \]

The proof proceeds by induction on the structure of rexp

- \text{rexp} = \mu:
  \[
  (h_0, t) \models \mu \\
  = \quad % h_0 = h_1 \circ \text{di}(h_1) \circ \text{di}^{-1}(h_0) \\
  (h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) \models \mu
  \]

- \text{rexp} = n:
  \[
  (h_0, t) \models n \\
  = \quad % Def. 20 \\
  \theta_{h_0}(0)(n) \\
  = \quad % h_0 = h_1 \circ \text{di}(h_1) \circ \text{di}^{-1}(h_0), \text{di}(h_1) \circ \text{di}^{-1}(h_0)(0) = 0 \\
  \theta_{h_1}(\text{di}(h_1) \circ \text{di}^{-1}(h_0)(0))(n) \\
  = \quad % Def. 20 \\
  (h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) \models n
  \]

- \text{exp} = n':
  \[
  (h_0, t) \models \!\!\!\!\! n' \\
  = \quad % Def. 20 \\
  \theta_{h_0}(0)(n) \\
  = \quad % h_0 = h_1 \circ \text{di}(h_1) \circ \text{di}^{-1}(h_0), \text{di}(h_1) \circ \text{di}^{-1}(h_0)(0) = 0 \\
  \theta_{h_1}(\text{di}(h_1) \circ \text{di}^{-1}(h_0)(0))(n) \\
  = \quad % Def. 20 \\
  (h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) \models \!\!\!\!\! n
  \]

- \text{exp} = \text{exp}_1 + \text{exp}_2:
  \[
  (h_0, t) \models \text{exp}_1 + \text{exp}_2 \\
  = \quad % Def. 20 \\
  (h_0, t) \models \text{exp}_1 + (h_0, t) \models \text{exp}_2 \\
  = \quad % Induction \\
  (h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) \models \text{exp}_1 + (h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) \models \text{exp}_2 \\
  = \quad % Def. 20 \\
  (h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) \models \text{exp}_1 + \text{exp}_2
  \]

141
The proof proceeds by induction on structure of $\exp$:

- $\exp = \exp$:
  
  $$(h_0, t) \models \exp$$
  
  $\%$ Theorem 2a
  
  $$(h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) \models \exp$$

- $\exp = \nu$:
  
  $$(h_0, t) \models \nu$$
  
  $\%$ Def. 20
  
  $$\theta_{h_0}(t)(\nu)$$
  
  $\%$ $h_0 = h_1 \circ \text{di}(h_1) \circ \text{di}^{-1}(h_0)$
  
  $$\theta_{h_1}(\text{di}(h_1) \circ \text{di}^{-1}(h_0)(t))(\nu)$$
  
  $\%$ Def. 20
  
  $$(h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) \models \nu$$

- $\exp = \nu'$:
  
  $$(h_0, t) \models \nu'$$
  
  $\%$ Def. 20
  
  $$\lim_{t \leftarrow t_1} \theta_{h_0}(t_1)(\nu)$$
  
  $\%$ $h_0 = h_1 \circ \text{di}(h_1) \circ \text{di}^{-1}(h_0)$
  
  $$\lim_{t \leftarrow t_1} \theta_{h_1}(\text{di}(h_1) \circ \text{di}^{-1}(h_0)(t_1))(\nu)$$
  
  $\%$ $t_2 = \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t_1)$
  
  $$\lim_{\text{di}(h_1) \circ \text{di}^{-1}(h_0)(t) \leftarrow t_2} \theta_{h_1}(t_2)(\nu)$$
  
  $\%$ Def. 20
  
  $$(h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) \models \nu'$$

- $\exp = \check{\nu}$:
  
  $t = 0$
  
  $$(h_0, 0) \models \check{\nu}$$
  
  $\%$ Def. 20
  
  $$\theta_{h_0}(0)(\nu)$$
  
  $\%$ $h_0 = h_1 \circ \text{di}(h_1) \circ \text{di}^{-1}(h_0)$
  
  $$\theta_{h_1}(\text{di}(h_1) \circ \text{di}^{-1}(h_0)(0))(\nu)$$
  
  $\%$ Def. 20, $\text{di}(h_1) \circ \text{di}^{-1}(h_0)(0) = 0$
  
  $$(h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(0)) \models \check{\nu}$$
Let $t > 0$

\[
\begin{align*}
(h_0, t) & \models \bar{v} \\
& = \% \text{ Def. } 20 \\
\lim_{t_1 \to t} \theta_{h_0}(t_1)(\bar{v}) \\
& = \% h_0 = h_1 \circ \text{di}^{-1}(h_0) \\
\lim_{t_1 \to t} \theta_{h_1}(\text{di}(h_1) \circ \text{di}^{-1}(h_0)(t_1))(\bar{v}) \\
& = \% t_2 = \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t_1) \\
\lim_{t_2 \to \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)} \theta_{h_1}(t_2)(\bar{v}) \\
& = \% \text{ Def. } 20 \\
(h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) & \models \bar{v}
\end{align*}
\]

1. $\text{exp} = x$

\[
\begin{align*}
(h_0, t) & \models x \\
& = \% \text{ Def. } 20 \\
\theta_{h_0}(t)(x) \\
& = \% h_0 = h_1 \circ \text{di}(h_1) \circ \text{di}^{-1}(h_0) \\
\theta_{h_1}(\text{di}(h_1) \circ \text{di}^{-1}(h_0)(t))(x) \\
& = \% \text{ Def. } 20 \\
(h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) & \models x
\end{align*}
\]

2. $\text{exp} = x'$

\[
\begin{align*}
(h_0, t) & \models x' \\
& = \% \text{ Def. } 20 \\
\lim_{t \to t_1} \theta_{h_0}(t)(x) \\
& = \% h_0 = h_1 \circ \text{di}(h_1) \circ \text{di}^{-1}(h_0) \\
\lim_{t \to t_1} \theta_{h_1}(\text{di}(h_1) \circ \text{di}^{-1}(h_0)(t))(x) \\
& = \% t_2 = \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t_1) \\
\lim_{t_2 \to \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t) \to t_2} \theta_{h_1}(t_2)(x) \\
& = \% \text{ Def. } 20 \\
(h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) & \models x'
\end{align*}
\]

3. $\text{exp} = \bar{x}$

\[
\begin{align*}
(h_0, 0) & \models \bar{x} \\
& = \% \text{ Def. } 20 \\
\theta_{h_0}(0)(x) \\
& = \% h_0 = h_1 \circ \text{di}(h_1) \circ \text{di}^{-1}(h_0) \\
\theta_{h_1}(\text{di}(h_1) \circ \text{di}^{-1}(h_0)(0))(x) \\
& = \% \text{ Def. } 20, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(0) = 0 \\
(h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(0)) & \models \bar{x}
\end{align*}
\]
proofs of dense model theorems

t > 0

(h₀, t) \models \neg x

= \% Def. 20
\lim_{t₁ \to t} \theta_{h₀}(t₁)(x)

= \% h₀ = h₁ \circ di(h₁) \circ di⁻¹(h₀)
\lim_{t₁ \to t} \theta_{h₁}(di(h₁) \circ di⁻¹(h₀)(t₁))(x)

= \% t₂ = di(h₁) \circ di⁻¹(h₀)(t₁)
\lim_{t₂ \to di(h₁) \circ di⁻¹(h₀)(t₁)} \theta_{h₁}(t₂)(x)

= \% Def. 20
(h₁, di(h₁) \circ di⁻¹(h₀)(t₁)) \models \neg x

• exp = exp₁ + exp₂:

(h₀, t) \models exp₁ + exp₂

= \% Def. 20
(h₀, t) \models exp₁ + (h₀, t) \models exp₂

= \% Induction
(h₁, di(h₁) \circ di⁻¹(h₀)(t₁)) \models exp₁ + (h₁, di(h₁) \circ di⁻¹(h₀)(t₁)) \models exp₂

= \% Def. 20
(h₁, di(h₁) \circ di⁻¹(h₀)(t₁)) \models exp₁ + exp₂

\forall t, h₀, h₁ : h₀ \succeq_{h₁} h₁ \to ((h₀, t) \models evexp = (h₁, di(h₁) \circ di⁻¹(h₀)(t₁)) \models evexp)

The proof proceeds by induction on structure of evexp:

• evexp = a?:

(h₀, t) \models a?

= \% Def. 20
a?

= \% Def. 20
(h₁, di(h₁) \circ di⁻¹(h₀)(t₁)) \models a?

• evexp = a!:

(h₀, t) \models a!

= \% Def. 20
a!

= \% Def. 20
(h₁, di(h₁) \circ di⁻¹(h₀)(t₁)) \models a!
A.3 Proof of Theorem 2

- $\text{evexp} = i$:

  
  \[
  (h_0, t) \models i
  \]
  
  \[
  = \quad \text{Def. 20}
  \]
  
  \[
  i
  \]
  
  \[
  = \quad \text{Def. 20}
  \]
  
  \[
  (h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) \models i
  \]

- $\text{evexp} = e$:

  
  \[
  (h_0, t) \models e
  \]
  
  \[
  = \quad \text{Def. 20}
  \]
  
  \[
  e
  \]
  
  \[
  = \quad \text{Def. 20}
  \]
  
  \[
  (h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) \models e
  \]

- $\text{evexp} = \lambda$:

  
  \[
  (h_0, t) \models \lambda
  \]
  
  \[
  = \quad \text{Def. 20}
  \]
  
  \[
  \lambda
  \]
  
  \[
  = \quad \text{Def. 20}
  \]
  
  \[
  (h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) \models \lambda
  \]

- $\text{evexp} = \epsilon$:

  
  \[
  (h_0, t) \models \epsilon
  \]
  
  \[
  = \quad \text{Def. 20}
  \]
  
  \[
  \psi_{h_0}(t)(\epsilon)
  \]
  
  \[
  = \quad \text{Def. 20}
  \]
  
  \[
  h_0 = h_1 \circ \text{di}(h_1) \circ \text{di}^{-1}(h_0)
  \]
  
  \[
  \psi_{h_1}(\text{di}(h_1) \circ \text{di}^{-1}(h_0)(t))(\epsilon)
  \]
  
  \[
  = \quad \text{Def. 20}
  \]
  
  \[
  (h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) \models \epsilon
  \]

- $\text{evexp} = \epsilon'$:

  
  \[
  (h_0, t) \models \epsilon'
  \]
  
  \[
  = \quad \text{Def. 20}
  \]
  
  \[
  \lim_{t \to t_1} \psi_{h_0}(t_1)(\epsilon)
  \]
  
  \[
  = \quad \text{Def. 20}
  \]
  
  \[
  h_0 = h_1 \circ \text{di}(h_1) \circ \text{di}^{-1}(h_0)
  \]
  
  \[
  \lim_{t \to t_1} \psi_{h_1}(\text{di}(h_1) \circ \text{di}^{-1}(h_0)(t_1))(\epsilon)
  \]
  
  \[
  = \quad \text{Def. 20}
  \]
  
  \[
  t_2 = \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t_1)
  \]
  
  \[
  \lim_{\text{di}(h_1) \circ \text{di}^{-1}(h_0)(t) \to t_2} \psi_{h_1}(t_2)(\epsilon)
  \]
  
  \[
  = \quad \text{Def. 20}
  \]
  
  \[
  (h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) \models \epsilon'
  \]
Proofs of Dense Model Theorems

- \( e v s p = ^*c: \)
  
  \( t = 0 \)
  
  \[
  (h_0, 0) \models ^*c
  \]
  
  = \% Def. 20
  
  \[
  \psi_{h_0}(0)(\epsilon)
  \]
  
  = \% \( h_0 = h_1 \circ \text{di}(h_1) \circ \text{di}^{-1}(h_0) \)
  
  \[
  \psi_{h_1}(\text{di}(h_1) \circ \text{di}^{-1}(h_0)(0))(\epsilon)
  \]
  
  = \% Def. 20, \( \text{di}(h_1) \circ \text{di}^{-1}(h_0)(0) = 0 \)
  
  \[
  (h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(0)) \models ^*c
  \]
  
  \( t > 0 \)
  
  \[
  (h_0, t) \models ^*c
  \]
  
  = \% Def. 20
  
  \[
  \lim_{t_i \to t} \psi_{h_0}(t_1)(\epsilon)
  \]
  
  = \% \( h_0 = h_1 \circ \text{di}(h_1) \circ \text{di}^{-1}(h_0) \)
  
  \[
  \lim_{t_i \to t} \theta_{h_1}(\text{di}(h_1) \circ \text{di}^{-1}(h_0)(t_1))(\epsilon)
  \]
  
  = \% \( t_2 = \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t_1) \)
  
  \[
  \lim_{t \to \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)} \psi_{h_1}(t_2)(\epsilon)
  \]
  
  = \% Def. 20
  
  \[
  (h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) \models ^*c
  \]

\[
\forall t, h_0, h_1 : h_0 \simeq_{\text{d}} h_1 \to ((h_0, t) \models p \iff (h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) \models p)
\]

*The proof proceeds by induction on structure of \( p \):

- \( p = \text{true}: \)
  
  \[
  (h_0, t) \models \text{true}
  \]
  
  = \% Def. 20
  
  \[
  \text{true}
  \]
  
  = \% Def. 20
  
  \[
  (h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) \models \text{true}
  \]

- \( p = (\text{exp}_1 = \text{exp}_2): \)
  
  \[
  (h_0, t) \models \text{exp}_1 = \text{exp}_2
  \]
  
  = \% Def. 20
  
  \[
  (h_0, t) \models \text{exp}_1 = (h_0, t) \models \text{exp}_2
  \]
  
  = \% Theorem 2b
  
  \[
  (h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) \models \text{exp}_1 = (h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) \models \text{exp}_2
  \]
  
  = \% Def. 20
  
  \[
  (h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) \models \text{exp}_1 = \text{exp}_2
  \]
A.3 Proof of Theorem 2

- \( p = (\text{exp}_1 < \text{exp}_2): \)

\[
(h_0, t) \models \text{exp}_1 < \text{exp}_2 \\
= \quad \% \; \text{Def. 20} \\
(h_0, t) \models \text{exp}_1 < \text{exp}_2 \\
= \quad \% \; \text{Theorem 2b} \\
(h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) \models \text{exp}_1 < (h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) \models \text{exp}_2 \\
= \quad \% \; \text{Def. 20} \\
(h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) \models \text{exp}_1 < \text{exp}_2
\]

- \( p = (\text{evexp}_1 = \text{evexp}_2): \)

\[
(h_0, t) \models \text{evexp}_1 = \text{evexp}_2 \\
= \quad \% \; \text{Def. 20} \\
(h_0, t) \models \text{evexp}_1 = (h_0, t) \models \text{evexp}_2 \\
= \quad \% \; \text{Theorem 2c} \\
(h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) \models \text{evexp}_1 = (h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) \models \text{evexp}_2 \\
= \quad \% \; \text{Def. 20} \\
(h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) \models \text{evexp}_1 = \text{evexp}_2
\]

- \( p = \neg p_0: \)

\[
(h_0, t) \models \neg p_0 \\
= \quad \% \; \text{Def. 20} \\
\text{not} \; (h_0, t) \models p_0 \\
= \quad \% \; \text{Induction} \\
\text{not} \; (h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) \models p_0 \\
= \quad \% \; \text{Def. 20} \\
(h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) \models \neg p_0
\]

- \( p = p_1 \lor p_2: \)

\[
(h_0, t) \models p_1 \lor p_2 \\
= \quad \% \; \text{Def. 20} \\
(h_0, t) \models p_1 \lor (h_0, t) \models p_2 \\
= \quad \% \; \text{Induction} \\
(h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) \models p_1 \lor (h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) \models p_2 \\
= \quad \% \; \text{Def. 20} \\
(h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) \models p_1 \lor p_2
\]
• $p = p_1 \Uppsi p_2$:

\[
(h_0, t) \models p_1 \Uppsi p_2 \\
= \quad \text{% Def. 20} \\
\exists t : (h_0, t_0) \models p_2 \text{ and } \forall t_1 \in (t, t_0) : (h_0, t_1) \models p_1 \\
= \quad \text{% Induction} \\
\exists t_0 > t : (h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t_0)) \models p_2 \\
\quad \text{and } \forall t_1 \in (t, t_0) : (h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t_1)) \models p_1 \\
= \quad \text{% Def. 20} \\
\exists t_2 > \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t) : (h_1, t_2) \models p_2 \\
\quad \text{and } \forall t_3 \in (t_2, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) : (h_1, t_3) \models p_1 \\
= \quad \text{% Def. 20} \\
(h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) \models p_1 \Uppsi p_2
\]

• $p = p_1 \check{\check{S}} p_2$:

\[
(h_0, t) \models p_1 \check{\check{S}} p_2 \\
= \quad \text{% Def. 20} \\
\exists t : (h_0, t_0) \models p_2 \text{ and } \forall t_1 \in (t, t_0) : (h_0, t_1) \models p_1 \\
= \quad \text{% Induction} \\
\exists t_0 < t : (h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t_0)) \models p_2 \\
\quad \text{and } \forall t_1 \in (t_0, t) : (h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t_1)) \models p_1 \\
= \quad \text{% Def. 20} \\
\exists t_2 < \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t) : (h_1, t_2) \models p_2 \\
\quad \text{and } \forall t_3 \in (t_2, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) : (h_1, t_3) \models p_1 \\
= \quad \text{% Def. 20} \\
(h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) \models p_1 \check{\check{S}} p_2
\]

• $p = \exists x. p_0$:

\[
(h_0, t) \models \exists x. p_0 \\
= \quad \text{% Def. 20} \\
\exists h_2 : h_2 \text{ x-variant of } h_0 \land (h_2, t) \models p_0 \\
= \quad \text{% Def. 20} \\
\exists h_3 : h_3 \text{ x-variant of } h_1 \land (h_3, h_2) \models p_0 \\
= \quad \text{% Def. 20} \\
(h_1, \text{di}(h_1) \circ \text{di}^{-1}(h_0)(t)) \models \exists x. p_0
\]
A.4 Proof of Lemma 2

\[ p = \exists c.p_0: \]

\[ (h_0, t) \models \exists c.p_0 \]
\[ = \quad \% \text{Def. 20} \]
\[ \exists h_2 : h_2 \text{ e-variant of } h_0 \land (h_2, t) \models p_0 \]
\[ = \quad \% \quad \% \text{Def. 20} \]
\[ \exists h_2 : h_2 \text{ e-variant of } h_1 \circ di(h_1) \circ di^{-1}(h_0) \land (h_2, t) \models p_0 \]
\[ = \quad \% \quad \% \text{Def. 20} \]
\[ \exists h_2 : h_2 \text{ e-variant of } h_1 \circ di(h_1) \circ di^{-1}(h_0) \land (h_2, t) \models p_0 \]
\[ = \quad \% \quad \% \text{Def. 20} \]
\[ \exists h_3 : h_3 \text{ e-variant of } h_1 \land (h_3, di(h_1) \circ di^{-1}(h_0)(t)) \models p_0 \]

\[ p = \exists n.p_0: \]

\[ (h_0, t) \models \exists n.p_0 \]
\[ = \quad \% \text{Def. 20} \]
\[ \exists h_3 : h_2 \text{ n-variant of } h_0 \land (h_2, t) \models p_0 \]
\[ = \quad \% \quad \% \text{Def. 20} \]
\[ \exists h_3 : h_2 \text{ n-variant of } h_1 \circ di(h_1) \circ di^{-1}(h_0) \land (h_2, t) \models p_0 \]
\[ = \quad \% \quad \% \text{Def. 20} \]
\[ \exists h_3 : h_2 \text{ n-variant of } h_1 \circ di(h_1) \circ di^{-1}(h_0) \land (h_2, t) \models p_0 \]
\[ = \quad \% \quad \% \text{Def. 20} \]
\[ \exists h_3 : h_2 \text{ n-variant of } h_1 \circ di(h_1) \circ di^{-1}(h_0)(t)) \models p_0 \]

A.4 Proof of Lemma 2

Lemma 2
Let exp be an expression, exp be a state expression, w ∈ S ∪ X, rexp be a state rigid expression, n ∈ N, eexp be an event expression, eexp a state event expression, e ∈ E, and p a temporal formula. Then the following holds:

\[ a \quad (h, t) \models exp_0[exp/w] = ((h : w \sim exp), t) \models exp_0 \]
\[ b \quad (h, t) \models exp_0[rexp/n] = ((h : n \sim rexp), t) \models exp_0 \]
\[ c \quad (h, t) \models eexp_0[eexp/e] = ((h : e \sim eexp), t) \models eexp_0 \]
\[ d \quad (h, t) \models p[exp/w] \iff ((h : w \sim exp), t) \models p \]
\[ e \quad (h, t) \models p[rexp/n] \iff ((h : n \sim rexp), t) \models p \]
\[ f \quad (h, t) \models p[eexp/e] \iff ((h : e \sim eexp), t) \models p \]

Proof 18

\[ a \quad (h, t) \models exp_0[exp/w] = ((h : w \sim exp), t) \models exp_0 \]

Proof by induction on the structure of exp_0:
Proofs of Dense Model Theorems

- \( \exp_0 = \text{rex}p \):
  \[
  (h, t) \models \text{rex}p[\exp/w]
  \]
  \[
  = \quad \% \text{ Def. 22}
  (h, t) \models \exp
  \]
  \[
  = \quad \% \text{ Def. 26, } w \notin \text{ var}\exp
  \]
  \[
  ((h : w \rightsquigarrow \exp), t) \models \exp
  \]

- \( \exp_0 = \mathbf{v} \):
  \[
  \mathbf{v} \equiv w
  \]
  \[
  (h, t) \models \mathbf{v}[\exp/w]
  \]
  \[
  = \quad \% \text{ Def. 22}
  (h, t) \models \exp
  \]
  \[
  = \quad \% \text{ Def. 26}
  ((h : w \rightsquigarrow \exp), t) \models \mathbf{v}
  \]
  \[
  \mathbf{v} \not\equiv w
  \]
  \[
  (h, t) \models \mathbf{v}[\exp/w]
  \]
  \[
  = \quad \% \text{ Def. 22}
  (h, t) \models \mathbf{v}
  \]
  \[
  = \quad \% \text{ Def. 26}
  ((h : w \rightsquigarrow \exp), t) \models \mathbf{v}
  \]

- \( \exp_0 = \mathbf{v}' \):
  \[
  \mathbf{v} \equiv w
  \]
  \[
  (h, t) \models \mathbf{v}'[\exp/w]
  \]
  \[
  = \quad \% \text{ Def. 22}
  (h, t) \models \exp'
  \]
  \[
  = \quad \% \text{ Def. 26}
  ((h : w \rightsquigarrow \exp), t) \models \mathbf{v}'
  \]
  \[
  \mathbf{v} \not\equiv w
  \]
  \[
  (h, t) \models \mathbf{v}'[\exp/w]
  \]
  \[
  = \quad \% \text{ Def. 22}
  (h, t) \models \mathbf{v}'
  \]
  \[
  = \quad \% \text{ Def. 26}
  ((h : w \rightsquigarrow \exp), t) \models \mathbf{v}'
  \]

- \( \exp_0 = \neg\mathbf{v} \):
  \[
  \mathbf{v} \equiv w
  \]
  \[
  (h, t) \models \neg\mathbf{v}[\exp/w]
  \]
  \[
  = \quad \% \text{ Def. 22}
  (h, t) \models \neg\exp
  \]
  \[
  = \quad \% \text{ Def. 26}
  ((h : w \rightsquigarrow \exp), t) \models \neg\mathbf{v}
  \]
A.4 Proof of Lemma 2

\[ v \not\equiv w \]

\[ (h, t) \models v[exp/w] = \% \text{Def. 22} \]

\[ (h, t) \models v = \% \text{Def. 26} \]

\[ ((h : w \leadsto exp), t) \models v \]

- \( exp_0 = x: \)

\[ x \equiv w \]

\[ (h, t) \models x[exp/w] = \% \text{Def. 22} \]

\[ (h, t) \models x = \% \text{Def. 26} \]

\[ ((h : w \leadsto exp), t) \models x \]

\[ x \not\equiv w \]

\[ (h, t) \models x[exp/w] = \% \text{Def. 22} \]

\[ (h, t) \models x = \% \text{Def. 26} \]

\[ ((h : w \leadsto exp), t) \models x \]

- \( exp_0 = x': \)

\[ x \equiv w \]

\[ (h, t) \models x'[exp/w] = \% \text{Def. 22} \]

\[ (h, t) \models x' = \% \text{Def. 26} \]

\[ ((h : w \leadsto exp), t) \models x' \]

\[ x \not\equiv w \]

\[ (h, t) \models x'[exp/w] = \% \text{Def. 22} \]

\[ (h, t) \models x' = \% \text{Def. 26} \]

\[ ((h : w \leadsto exp), t) \models x' \]

- \( exp_0 = \bar{x}: \)

\[ x \equiv w \]

\[ (h, t) \models \bar{x}[exp/w] = \% \text{Def. 22} \]

\[ (h, t) \models \bar{x} = \% \text{Def. 26} \]

\[ ((h : w \leadsto exp), t) \models \bar{x} \]
Proofs of Dense Model Theorems

\[ x \not\in w \]

\[(h, t) \models x[exp/w] \]

\[ \% \text{ Def. 22} \]

\[(h, t) \models x \]

\[ \% \text{ Def. 26} \]

\[ ((h : w \rightsquigarrow exp), t) \models x \]

- \( exp_0 \equiv exp_1 + exp_2 \)

\[(h, t) \models (exp_1 + exp_2)[exp/w] \]

\[ \% \text{ Def. 22} \]

\[(h, t) \models exp_1[exp/w] + exp_2[exp/w] \]

\[ \% \text{ Def. 20} \]

\[(h, t) \models exp_1[exp/w] + (h, t) \models exp_2[exp/w] \]

\[ \% \text{ Induction} \]

\[ ((h : w \rightsquigarrow exp), t) \models exp_1 + ((h : w \rightsquigarrow exp), t) \models exp_2 \]

\[ \% \text{ Def. 20} \]

\[ ((h : w \rightsquigarrow exp), t) \models exp_1 + exp_2 \]

\[ \text{Proof by induction on structure of } exp_0 \]

- \( exp_0 = \mu: \)

\[(h, t) \models \mu[rexp/n] \]

\[ \% \text{ Def. 23} \]

\[(h, t) \models \mu \]

\[ \% \text{ Def. 26 and 20} \]

\[ ((h : n \rightsquigarrow rexp), t) \models \mu \]

- \( exp_0 = n_0: \)

\[ n_0 \equiv n \]

\[(h, t) \models n_0[rexp/n] \]

\[ \% \text{ Def. 23} \]

\[(h, t) \models rexp \]

\[ \% \text{ Def. 26} \]

\[ ((h : n \rightsquigarrow rexp), t) \models n_0 \]

\[ n_0 \not\equiv n \]

\[(h, t) \models n_0[rexp/n] \]

\[ \% \text{ Def. 23} \]

\[(h, t) \models n_0 \]

\[ \% \text{ Def. 26} \]

\[ ((h : n \rightsquigarrow rexp), t) \models n_0 \]
A.4 Proof of Lemma 2

- \( \text{exp}_0 = \neg n_0 \):
  \[ n_0 \equiv n \]
  \[ (h, t) \models n_0[\text{exp}/n] \]
  \[ = \quad \% \text{Def. 23} \]
  \[ (h, t) \models \text{exp'} \]
  \[ = \quad \% \text{Def. 26} \]
  \[ ((h : n \leadsto \text{exp}), t) \models n'_0 \]

- \( n_0 \not\equiv n \):
  \[ (h, t) \models n'_0[\text{exp}/n] \]
  \[ = \quad \% \text{Def. 23} \]
  \[ (h, t) \models n'_0 \]
  \[ = \quad \% \text{Def. 26} \]
  \[ ((h : n \leadsto \text{exp}), t) \models n'_0 \]

- \( \text{exp}_0 = \neg n_0 \):
  \[ n_0 \equiv n \]
  \[ (h, t) \models \neg n_0[\text{exp}/n] \]
  \[ = \quad \% \text{Def. 23} \]
  \[ (h, t) \models \neg \text{exp} \]
  \[ = \quad \% \text{Def. 26} \]
  \[ ((h : n \leadsto \text{exp}), t) \models \neg n_0 \]

- \( n_0 \not\equiv n \):
  \[ (h, t) \models \neg n_0[\text{exp}/n] \]
  \[ = \quad \% \text{Def. 23} \]
  \[ (h, t) \models \neg n_0 \]
  \[ = \quad \% \text{Def. 26} \]
  \[ ((h : n \leadsto \text{exp}), t) \models \neg n_0 \]

- \( \text{exp}_0 = w \):
  \[ (h, t) \models w[\text{exp}/n] \]
  \[ = \quad \% \text{Def. 23} \]
  \[ (h, t) \models w \]
  \[ = \quad \% \text{Def. 26} \]
  \[ ((h : n \leadsto \text{exp}), t) \models w \]

- \( \text{exp}_0 = w' \):
  \[ (h, t) \models w'[\text{exp}/n] \]
  \[ = \quad \% \text{Def. 23} \]
  \[ (h, t) \models w' \]
  \[ = \quad \% \text{Def. 26} \]
  \[ ((h : n \leadsto \text{exp}), t) \models w' \]
Proofs of Dense Model Theorems

- $\exp_0 = 'w$:

  $$(h, t) \models 'w[\text{rexp/}n]$$
  
  $$(h, t) \models 'w$$
  
  $$(h : n \leadsto \text{rexp}), t) \models 'w$$

- $\exp_0 \equiv \exp_1 + \exp_2$

  $$(h, t) \models (\exp_1 + \exp_2)[\text{rexp/}n]$$
  
  $$(h, t) \models \exp_1[\text{rexp/}n] + \exp_2[\text{rexp/}n]$$
  
  $$(h, t) \models \exp_1[\text{rexp/}n] + (h, t) \models \exp_2[\text{rexp/}n]$$
  
  $$(h : n \leadsto \text{rexp}), t) \models \exp_1 + ((h : n \leadsto \text{rexp}), t) \models \exp_2$$
  
  $$(h : n \leadsto \text{rexp}), t) \models \exp_1 + \exp_2$$

  $c \ (h, t) \models ev\exp_0[ev\exp/e] = ((h : e \leadsto ev\exp), t) \models ev\exp_0$

Proof by induction on structure of $ev\exp_0$:

- $ev\exp_0 = \lambda$:

  $$(h, t) \models \lambda[ev\exp/e]$$
  
  $$(h, t) \models \lambda$$
  
  $$(h : e \leadsto ev\exp), t) \models \lambda$$

- $ev\exp_0 = \text{a?}$:

  $$(h, t) \models \text{a?}[ev\exp/e]$$
  
  $$(h, t) \models \text{a?}$$
  
  $$(h : e \leadsto ev\exp), t) \models \text{a?}$$

- $ev\exp_0 = \text{a!}$:

  $$(h, t) \models \text{a!}[ev\exp/e]$$
  
  $$(h, t) \models \text{a!}$$
  
  $$(h : e \leadsto ev\exp), t) \models \text{a!}$$
A.4 Proof of Lemma 2

\[
\begin{align*}
\text{• } & \text{ev}_0 \equiv i: \\
& (h, t) \models i[\text{ev}_0] \\
& = % \text{Def. 24} \\
& (h, t) \models i \\
& = % \text{Def. 26 and } 20 \\
& ((h : e \mapsto \text{ev}_0), t) \models i \\
\text{• } & \text{ev}_0 \equiv e: \\
& (h, t) \models e[\text{ev}_0] \\
& = % \text{Def. 24} \\
& (h, t) \models e \\
& = % \text{Def. 26 and } 20 \\
& ((h : e \mapsto \text{ev}_0), t) \models e \\
\text{• } & \text{ev}_0 \equiv \epsilon_0: \\
& \epsilon_0 \equiv \epsilon \\
& (h, t) \models \epsilon_0[\text{ev}_0] \\
& = % \text{Def. 24} \\
& (h, t) \models \epsilon \\
& = % \text{Def. 26} \\
& ((h : \epsilon \mapsto \text{ev}_0), t) \models \epsilon_0 \\
& \epsilon_0 \not\equiv \epsilon \\
& (h, t) \models \epsilon_0[\text{ev}_0] \\
& = % \text{Def. 24} \\
& (h, t) \models \epsilon_0 \\
& = % \text{Def. 26} \\
& ((h : \epsilon \mapsto \text{ev}_0), t) \models \epsilon_0 \\
\text{• } & \text{ev}_0 \equiv \epsilon'_0: \\
& \epsilon_0 \equiv \epsilon \\
& (h, t) \models \epsilon_0[\text{ev}_0] \\
& = % \text{Def. 24} \\
& (h, t) \models \epsilon'_0 \\
& = % \text{Def. 26} \\
& ((h : \epsilon \mapsto \text{ev}_0), t) \models \epsilon'_0 \\
& \epsilon_0 \not\equiv \epsilon \\
& (h, t) \models \epsilon_0[\text{ev}_0] \\
& = % \text{Def. 24} \\
& (h, t) \models \epsilon'_0 \\
& = % \text{Def. 26} \\
& ((h : \epsilon \mapsto \text{ev}_0), t) \models \epsilon'_0
\end{align*}
\]


- $evexp_0 = \gamma_0:
  \gamma_0 \equiv \epsilon$

  \[
  (h, t) \models \gamma_0[evexp/\epsilon] \\
  = \% \text{ Def. 24} \\
  (h, t) \models evexp \\
  = \% \text{ Def. 26} \\
  ((h : \epsilon \sim evexp), t) \models \gamma_0
  
  \]

- $\gamma_0 \neq \epsilon$

  \[
  (h, t) \models \gamma_0[evexp/\epsilon] \\
  = \% \text{ Def. 24} \\
  (h, t) \models \gamma_0 \\
  = \% \text{ Def. 26} \\
  ((h : \epsilon \sim evexp), t) \models \gamma_0
  
  \]

\[d \quad (h, t) \models p[exp/w] \iff ((h : w \rightsquigarrow exp), t) \models p\]

**Proof by induction on structure of $p$:**

- $p = \text{true}$:

  \[
  (h, t) \models \text{true}[exp/w] \\
  = \% \text{ Def. 25} \\
  (h, t) \models \text{true} \\
  = \% \text{ Def. 26} \\
  ((h : w \rightsquigarrow exp), t) \models \text{true}
  
  \]

- $p = (exp_1 = exp_2)$:

  \[
  (h, t) \models (exp_1 = exp_2)[exp/w] \\
  = \% \text{ Def. 25} \\
  (h, t) \models exp_1[exp/w] = exp_2[exp/w] \\
  = \% \text{ Def. 20} \\
  (h, t) \models exp_1[exp/w] = (h, t) \models exp_2[exp/w] \\
  = \% \text{ Lemma 2a} \\
  ((h : w \rightsquigarrow exp), t) \models exp_1 = ((h : w \rightsquigarrow exp), t) \models exp_2 \\
  = \% \text{ Def. 20} \\
  ((h : w \rightsquigarrow exp), t) \models exp_1 = exp_2
  
  \]
A.4 Proof of Lemma 2

- \( p = (exp_1 < exp_2) \):

\[
(h,t) \models (exp_1 < exp_2)[exp/w]
= \ % \text{Def. 25}
(h,t) \models exp_1[exp/w] < exp_2[exp/w]
= \ % \text{Def. 20}
(h,t) \models exp_1[exp/w] < (h,t) \models exp_2[exp/w]
= \ % \text{Lemma ref.su.hea}
(((h : w \leadsto exp), t) \models exp_1 < ((h : w \leadsto exp), t) \models exp_2
= \ % \text{Def. 20}
(((h : w \leadsto exp), t) \models exp_1 < exp_2

- \( p = (evexp_1 = evexp_2) \):

\[
(h,t) \models (evexp_1 = evexp_2)[exp/w]
= \ % \text{Def. 25}
(h,t) \models evexp_1 = evexp_2
= \ % \text{Def. 20}
(h,t) \models evexp_1 = (h,t) \models evexp_2
= \ % \text{Def. 26}
(((h : w \leadsto exp), t) \models evexp_1 = ((h : w \leadsto exp), t) \models evexp_2
= \ % \text{Def. 20}
(((h : w \leadsto exp), t) \models evexp_1 = evexp_2

- \( p = \neg p_1 \):

\[
(h,t) \models (\neg p_1)[exp/w]
= \ % \text{Def. 25}
(h,t) \models \neg(p_1[exp/w])
= \ % \text{Def. 20}
\text{not } (h,t) \models p_1[exp/w]
= \ % \text{Induction}
\text{not } ((h : w \leadsto exp), t) \models p_1
= \ % \text{Def. 20}
(((h : w \leadsto exp), t) \models \neg p_1

- \( p = p_1 \lor p_2 \):

\[
(h,t) \models (p_1 \lor p_2)[exp/w]
= \ % \text{Def. 25}
(h,t) \models (p_1[exp/w] \lor p_2[exp/w])
= \ % \text{Def. 20}
(h,t) \models p_1[exp/w] \text{ or } (h,t) \models p_2[exp/w]
= \ % \text{Induction}
(((h : w \leadsto exp), t) \models p_1 \text{ or } ((h : w \leadsto exp), t) \models p_2
= \ % \text{Def. 20}
(((h : w \leadsto exp), t) \models p_1 \lor p_2
• $p = p_1 \tilde{U} p_2$:

$$(h,t) \models (p_1 \tilde{U} p_2)[exp/w]$$

= % Def. 25

$$(h,t) \models (p_1 [exp/w] \tilde{U} p_2 [exp/w])$$

= % Def. 20

$\exists t_0 > t : (h, t_0) \models p_2 [exp/w] \text{ and } \forall t_1 \in (t, t_0) : (h, t_1) \models p_1 [exp/w]$$

= % Induction

$\exists t_0 < t : ((h : w \sim exp), t_0) \models p_2$

and $\forall t_1 \in (t_0, t) : ((h : w \sim exp), t_1) \models p_1$

= % Def. 20

$((h : w \sim exp), t) \models p_1 \tilde{U} p_2$

• $p = p_1 \hat{S} p_2$:

$$(h,t) \models (p_1 \hat{S} p_2)[exp/w]$$

= % Def. 25

$$(h,t) \models (p_1 [exp/w] \hat{S} p_2 [exp/w])$$

= % Def. 20

$\exists t_0 < t : (h, t_0) \models p_2 [exp/w] \text{ and } \forall t_1 \in (t_0, t) : (h, t_1) \models p_1 [exp/w]$$

= % Induction

$\exists t_0 < t : ((h : w \sim exp), t_0) \models p_2$

and $\forall t_1 \in (t_0, t) : ((h : w \sim exp), t_1) \models p_1$

= % Def. 20

$((h : w \sim exp), t) \models p_1 \hat{S} p_2$

• $p = \exists x.p_1$ for $x \notin \text{var(exp)} \cup \{w\}$:

$$(h,t) \models (\exists x.p_1)[exp/w]$$

= % Def. 25

$$(h,t) \models \exists x.(p_1 [exp/w])$$

= % Def. 20

$\exists h_1 : h_1 \text{x-variant of } h \text{ and } (h_1, t) \models p_1 [exp/w]$$

= % Induction

$\exists h_1 : h_1 \text{x-variant of } h \text{ and } ((h_1 : w \sim exp), t) \models p_1$

= % $h_1 = (h_1 : w \sim exp), h_3 = (h : w \sim exp)$

$h_1 \text{x-variant of } h \text{ iff } h_2 \text{x-variant of } h_3$

$\exists h_2 : h_2 \text{x-variant of } h_3 \text{ and } (h_2, t) \models p_1$

= % Def. 20

$((h : w \sim exp), t) \models \exists x.p_1$

158
A.4 Proof of Lemma 2

- \( p = \exists c.p_1 \):

\[
(h, t) \models (\exists c.p_1)[exp/w]
\]

\[
= \% \text{Def. 25}
\]

\[
(h, t) \models \exists c.(p_1[exp/w])
\]

\[
= \% \text{Def. 20}
\]

\[
\exists h_1 : h_1 \text{-variant of } h \text{ and } (h_1, t) \models p_1[exp/w]
\]

\[
= \% \text{Induction}
\]

\[
\exists h_1 : h_1 \text{-variant of } h \text{ and } ((h_1 : w \leadsto exp), t) \models p_1
\]

\[
= \% \quad h_2 = (h_1 : w \leadsto exp), h_3 = (h : w \leadsto exp)
\]

\[
\quad h_1 \text{-variant of } h \iff h_2 \text{-variant of } h_3
\]

\[
\exists h_2 : h_2 \text{-variant of } h_3 \text{ and } (h_2, t) \models p_1
\]

\[
= \% \text{Def. 20}
\]

\[
((h : w \leadsto exp), t) \models \exists c.p_1
\]

- \( p = \exists n.p_1 \):

\[
(h, t) \models (\exists n.p_1)[exp/w]
\]

\[
= \% \text{Def. 25}
\]

\[
(h, t) \models \exists n.(p_1[exp/w])
\]

\[
= \% \text{Def. 20}
\]

\[
\exists h_1 : h_1 \text{-variant of } h \text{ and } (h_1, t) \models p_1[exp/w]
\]

\[
= \% \text{Induction}
\]

\[
\exists h_1 : h_1 \text{-variant of } h \text{ and } ((h_1 : w \leadsto exp), t) \models p_1
\]

\[
= \% \quad h_2 = (h_1 : w \leadsto exp), h_3 = (h : w \leadsto exp)
\]

\[
\quad h_1 \text{-variant of } h \iff h_2 \text{-variant of } h_3
\]

\[
\exists h_2 : h_2 \text{-variant of } h_3 \text{ and } (h_2, t) \models p_1
\]

\[
= \% \text{Def. 20}
\]

\[
((h : w \leadsto exp), t) \models \exists n.p_1
\]

\[
e (h, t) \models p[\text{exp}/n] \iff ((h : n \leadsto \text{exp}), t) \models p
\]

Proof by induction on structure of \( p \):

- \( p = \text{true} \):

\[
(h, t) \models \text{true}[\text{exp}/n]
\]

\[
= \% \text{Def. 25}
\]

\[
(h, t) \models \text{true}
\]

\[
= \% \text{Def. 26}
\]

\[
((h : n \leadsto \text{exp}), t) \models \text{true}
\]
• \( p = (\text{exp}_1 = \text{exp}_2) \):

\[
(h, t) \models (\text{exp}_1 = \text{exp}_2)[\text{exp}/n]
\]

= % Def. 25

\[
(h, t) \models \text{exp}_1[\text{exp}/n] = \text{exp}_2[\text{exp}/n]
\]

= % Def. 20

\[
(h, t) \models \text{exp}_1[\text{exp}/n] = (h, t) \models \text{exp}_2[\text{exp}/n]
\]

= % Lemma 2b

\[
((h : n \rightarrow \text{exp}), t) \models \text{exp}_1 = ((h : n \rightarrow \text{exp}), t) \models \text{exp}_2
\]

= % Def. 20

\[
((h : n \rightarrow \text{exp}), t) \models \text{exp}_1 = \text{exp}_2
\]

• \( p = (\text{exp}_1 < \text{exp}_2) \):

\[
(h, t) \models (\text{exp}_1 < \text{exp}_2)[\text{exp}/n]
\]

= % Def. 25

\[
(h, t) \models \text{exp}_1[\text{exp}/n] < \text{exp}_2[\text{exp}/n]
\]

= % Def. 20

\[
(h, t) \models \text{exp}_1[\text{exp}/n] < (h, t) \models \text{exp}_2[\text{exp}/n]
\]

= % Lemma 2b

\[
((h : n \rightarrow \text{exp}), t) \models \text{exp}_1 < ((h : n \rightarrow \text{exp}), t) \models \text{exp}_2
\]

= % Def. 20

\[
((h : n \rightarrow \text{exp}), t) \models \text{exp}_1 < \text{exp}_2
\]

• \( p = (\text{evexp}_1 = \text{evexp}_2) \):

\[
(h, t) \models (\text{evexp}_1 = \text{evexp}_2)[\text{exp}/n]
\]

= % Def. 25

\[
(h, t) \models \text{evexp}_1 = \text{evexp}_2
\]

= % Def. 20

\[
(h, t) \models \text{evexp}_1 = (h, t) \models \text{evexp}_2
\]

= % Def. 26

\[
((h : n \rightarrow \text{exp}), t) \models \text{evexp}_1 = ((h : n \rightarrow \text{exp}), t) \models \text{evexp}_2
\]

= % Def. 20

\[
((h : n \rightarrow \text{exp}), t) \models \text{evexp}_1 = \text{evexp}_2
\]

• \( p = \neg p_1 \):

\[
(h, t) \models (\neg p_1)[\text{exp}/n]
\]

= % Def. 25

\[
(h, t) \models \neg (p_1[\text{exp}/n])
\]

= % Def. 20

not \( (h, t) \models p_1[\text{exp}/n] \)

= % Induction

not \( ((h : n \rightarrow \text{exp}), t) \models p_1 \)

= % Def. 20

\[
((h : n \rightarrow \text{exp}), t) \models \neg p_1
\]
A.4 Proof of Lemma 2

- \( p = p_1 \lor p_2 : \)

\[
(h, t) \models (p_1 \lor p_2)[\text{rexp/n}]
= \quad \text{Def. 25}
(h, t) \models (p_1[\text{rexp/n}] \lor p_2[\text{rexp/n}])
= \quad \text{Def. 20}
(h, t) \models p_1[\text{rexp/n}] \lor (h, t) \models p_2[\text{rexp/n}]
= \quad \text{Induction}
((h : n \leadsto \text{rexp}), t) \models p_1 \lor ((h : n \leadsto \text{rexp}), t) \models p_2
= \quad \text{Def. 20}
((h : n \leadsto \text{rexp}), t) \models p_1 \lor p_2
\]

- \( p = p_1 \hat{\text{U}} p_2 : \)

\[
(h, t) \models (p_1 \hat{\text{U}} p_2)[\text{rexp/n}]
= \quad \text{Def. 25}
(h, t) \models (p_1[\text{rexp/n}] \hat{\text{U}} p_2[\text{rexp/n}])
= \quad \text{Def. 20}
\exists t_0 \succ t : (h, t_0) \models p_1[\text{rexp/n}] \quad \text{and} \quad \forall t_1 \in (t, t_0) : (h, t_1) \models p_1[\text{rexp/n}]
= \quad \text{Induction}
\exists t_0 \succ t : ((h : n \leadsto \text{rexp}), t_0) \models p_2
\quad \text{and} \quad \forall t_1 \in (t, t_0) : ((h : n \leadsto \text{rexp}), t_1) \models p_1
= \quad \text{Def. 20}
((h : n \leadsto \text{rexp}), t) \models p_1 \hat{\text{U}} p_2
\]

- \( p = p_1 \hat{\text{S}} p_2 : \)

\[
(h, t) \models (p_1 \hat{\text{S}} p_2)[\text{rexp/n}]
= \quad \text{Def. 25}
(h, t) \models (p_1[\text{rexp/n}] \hat{\text{S}} p_2[\text{rexp/n}])
= \quad \text{Def. 20}
\exists t_0 \prec t : (h, t_0) \models p_1[\text{rexp/n}] \quad \text{and} \quad \forall t_1 \in (t_0, t) : (h, t_1) \models p_1[\text{rexp/n}]
= \quad \text{Induction}
\exists t_0 \prec t : ((h : n \leadsto \text{rexp}), t_0) \models p_2
\quad \text{and} \quad \forall t_1 \in (t_0, t) : ((h : n \leadsto \text{rexp}), t_1) \models p_1
= \quad \text{Def. 20}
((h : n \leadsto \text{rexp}), t) \models p_1 \hat{\text{S}} p_2
\]
\[ p = \exists x. p_1 : \]

\[
(h, t) \models (\exists x. p_1)[\text{exp}/n]
\]

\[
= \quad \% \, \text{Def. 25}
\]

\[
(h, t) \models \exists x. p_1[\text{exp}/n]
\]

\[
= \quad \% \, \text{Def. 20}
\]

\[
\exists h_1 : h_1 \text{ x-variant of } h \text{ and } (h_1, t) \models p_1[\text{exp}/n]
\]

\[
= \quad \% \, \text{Induction}
\]

\[
\exists h_1 : h_1 \text{ x-variant of } h \text{ and } ((h_1 : n \sim \text{exp}), t) \models p_1
\]

\[
= \quad \% \, h_2 = (h_1 : n \sim \text{exp}), h_3 = (h : n \sim \text{exp})
\]

\[
h_1 \text{ x-variant of } h \iff h_2 \text{ x-variant of } h_3
\]

\[
\exists h_2 : h_2 \text{ x-variant of } h_3 \text{ and } (h_2, t) \models p_1
\]

\[
= \quad \% \, \text{Def. 20}
\]

\[
((h : n \sim \text{exp}), t) \models \exists x. p_1
\]

\[ p = \exists e. p_1 : \]

\[
(h, t) \models (\exists e. p_1)[\text{exp}/n]
\]

\[
= \quad \% \, \text{Def. 25}
\]

\[
(h, t) \models \exists e. p_1[\text{exp}/n]
\]

\[
= \quad \% \, \text{Def. 20}
\]

\[
\exists h_1 : h_1 \text{ e-variant of } h \text{ and } (h_1, t) \models p_1[\text{exp}/n]
\]

\[
= \quad \% \, \text{Induction}
\]

\[
\exists h_1 : h_1 \text{ e-variant of } h \text{ and } ((h_1 : n \sim \text{exp}), t) \models p_1
\]

\[
= \quad \% \, h_2 = (h_1 : n \sim \text{exp}), h_3 = (h : n \sim \text{exp})
\]

\[
h_1 \text{ e-variant of } h \iff h_2 \text{ e-variant of } h_3
\]

\[
\exists h_2 : h_2 \text{ e-variant of } h_3 \text{ and } (h_2, t) \models p_1
\]

\[
= \quad \% \, \text{Def. 20}
\]

\[
((h : n \sim \text{exp}), t) \models \exists e. p_1
\]

\[ p = \exists n_0. p_1 : \text{ for } n_0 \notin \text{var}\text{exp} \cup \{n\} \]

\[
(h, t) \models (\exists n_0. p_1)[\text{exp}/n]
\]

\[
= \quad \% \, \text{Def. 25}
\]

\[
(h, t) \models \exists n_0. p_1[\text{exp}/n]
\]

\[
= \quad \% \, \text{Def. 20}
\]

\[
\exists h_1 : h_1 \text{ n}_0\text{-variant of } h \text{ and } (h_1, t) \models p_1[\text{exp}/n]
\]

\[
= \quad \% \, \text{Induction}
\]

\[
\exists h_1 : h_1 \text{ n}_0\text{-variant of } h \text{ and } ((h_1 : n \sim \text{exp}), t) \models p_1
\]

\[
= \quad \% \, h_2 = (h_1 : n \sim \text{exp}), h_3 = (h : n \sim \text{exp})
\]

\[
h_1 \text{ n}_0\text{-variant of } h \iff h_2 \text{ n}_0\text{-variant of } h_3
\]

\[
\exists h_2 : h_2 \text{ n}_0\text{-variant of } h_3 \text{ and } (h_2, t) \models p_1
\]

\[
= \quad \% \, \text{Def. 20}
\]

\[
((h : n \sim \text{exp}), t) \models \exists n_0. p_1
\]
A.4 Proof of Lemma 2

\[ f \cdot h, t \vdash p[\text{evexp}/\epsilon] \iff ((h : \epsilon \rightarrow \text{evexp}), t) \vdash p \]

Proof by induction on structure of \( p \):

- \( p = \text{true} \):

  \[
  (h, t) \vdash \text{true}[\text{evexp}/\epsilon]
  = \quad \text{Def.} \ 25
  (h, t) \vdash \text{true}
  = \quad \text{Def.} \ 26
  ((h : \epsilon \rightarrow \text{evexp}), t) \vdash \text{true}
  
  \]

- \( p = (\text{exp}_1 = \text{exp}_2) \):

  \[
  (h, t) \vdash (\text{exp}_1 = \text{exp}_2)[\text{evexp}/\epsilon]
  = \quad \text{Def.} \ 25
  (h, t) \vdash \text{exp}_1 = \text{exp}_2
  = \quad \text{Def.} \ 20
  (h, t) \vdash \text{exp}_1 = (h, t) \vdash \text{exp}_2
  = \quad \text{Def.} \ 26
  ((h : \epsilon \rightarrow \text{evexp}), t) \vdash \text{exp}_1 = ((h : \epsilon \rightarrow \text{evexp}), t) \vdash \text{exp}_2
  = \quad \text{Def.} \ 20
  ((h : \epsilon \rightarrow \text{evexp}), t) \vdash \text{exp}_1 = \text{exp}_2
  
  \]

- \( p = (\text{exp}_1 < \text{exp}_2) \):

  \[
  (h, t) \vdash (\text{exp}_1 < \text{exp}_2)[\text{evexp}/\epsilon]
  = \quad \text{Def.} \ 25
  (h, t) \vdash \text{exp}_1 < \text{exp}_2
  = \quad \text{Def.} \ 20
  (h, t) \vdash \text{exp}_1 < (h, t) \vdash \text{exp}_2
  = \quad \text{Def.} \ 26
  ((h : \epsilon \rightarrow \text{evexp}), t) \vdash \text{exp}_1 < ((h : \epsilon \rightarrow \text{evexp}), t) \vdash \text{exp}_2
  = \quad \text{Def.} \ 20
  ((h : \epsilon \rightarrow \text{evexp}), t) \vdash \text{exp}_1 < \text{exp}_2
  
  \]

- \( p = (\text{evexp}_1 = \text{evexp}_2) \):

  \[
  (h, t) \vdash (\text{evexp}_1 = \text{evexp}_2)[\text{evexp}/\epsilon]
  = \quad \text{Def.} \ 25
  (h, t) \vdash \text{evexp}_1[\text{evexp}_1/\epsilon] = \text{evexp}_2[\text{evexp}_1/\epsilon]
  = \quad \text{Def.} \ 20
  (h, t) \vdash \text{evexp}_1[\text{evexp}_1/\epsilon] = (h, t) \vdash \text{evexp}_2[\text{evexp}_1/\epsilon]
  = \quad \text{Lemma} \ 2c
  ((h : \epsilon \rightarrow \text{evexp}), t) \vdash \text{evexp}_1 = ((h : \epsilon \rightarrow \text{evexp}), t) \vdash \text{evexp}_2
  = \quad \text{Def.} \ 20
  ((h : \epsilon \rightarrow \text{evexp}), t) \vdash \text{evexp}_1 = \text{evexp}_2
  
  \]
• $p = \neg p_1$:

$$(h, t) \models (\neg p_1)[\text{evexp}/e]$$

$= \quad \% \text{ Def. 25}$$

$$(h, t) \models \neg (p_1[\text{evexp}/e])$$

$= \quad \% \text{ Def. 20}$$

$not \ (h, t) \models p_1[\text{evexp}/e]$

$= \quad \% \text{ Induction}$

$not \ ((h : e \leadsto \text{evexp}), t) \models p_1$

$= \quad \% \text{ Def. 20}$$

$((h : e \leadsto \text{evexp}), t) \models \neg p_1$


• $p = p_1 \lor p_2$:

$$(h, t) \models (p_1 \lor p_2)[\text{evexp}/e]$$

$= \quad \% \text{ Def. 25}$$

$$(h, t) \models (p_1[\text{evexp}/e] \lor p_2[\text{evexp}/e])$$

$= \quad \% \text{ Def. 20}$$

$$(h, t) \models p_1[\text{evexp}/e] \lor (h, t) \models p_2[\text{evexp}/e]$$

$= \quad \% \text{ Induction}$

$((h : e \leadsto \text{evexp}), t) \models p_1 \lor ((h : e \leadsto \text{evexp}), t) \models p_2$

$= \quad \% \text{ Def. 20}$$

$((h : e \leadsto \text{evexp}), t) \models p_1 \lor p_2$


• $p = p_1 \hat{\cup} p_2$:

$$(h, t) \models (p_1 \hat{\cup} p_2)[\text{evexp}/e]$$

$= \quad \% \text{ Def. 25}$$

$$(h, t) \models (p_1[\text{evexp}/e] \hat{\cup} p_2[\text{evexp}/e])$$

$= \quad \% \text{ Def. 20}$$

$$\exists t_0 > t : (h, t_0) \models p_2[\text{evexp}/e] \text{ and } \forall t_1 \in (t, t_0) : (h, t_1) \models p_1[\text{evexp}/e]$$

$= \quad \% \text{ Induction}$

$$\exists t_0 > t : ((h : e \leadsto \text{evexp}), t_0) \models p_2$$

and $\forall t_1 \in (t, t_0) : ((h : e \leadsto \text{evexp}), t_1) \models p_1$

$= \quad \% \text{ Def. 20}$$

$((h : e \leadsto \text{evexp}), t) \models p_1 \hat{\cup} p_2$
A.4 Proof of Lemma 2

- \( p = p_1 \hat{S} p_2 \):

\[
(h, t) \models (p_1 \hat{S} p_2)[\text{evexp}/c]
\]
\[
= \quad \% \text{ Def. 25}
\]
\[
(h, t) \models (p_1[\text{evexp}/c] \hat{S} p_2[\text{evexp}/c])
\]
\[
= \quad \% \text{ Def. 20}
\]
\[
\exists t_0 < t : (h, t_0) \models p_2[\text{evexp}/c] \text{ and } \forall t_1 \in (t_0, t) : (h, t_1) \models p_1[\text{evexp}/c]
\]
\[
= \quad \% \text{ Induction}
\]
\[
\exists t_0 < t : ((h : c \sim \text{evexp}), t_0) \models p_2
\]
\[
\text{and } \forall t_1 \in (t_0, t) : ((h : c \sim \text{evexp}), t_1) \models p_1
\]
\[
= \quad \% \text{ Def. 20}
\]
\[
((h : c \sim \text{evexp}), t) \models p_1 \hat{S} p_2
\]

- \( p = \exists x. p_1 \):

\[
(h, t) \models (\exists x. p_1)[\text{evexp}/c]
\]
\[
= \quad \% \text{ Def. 25}
\]
\[
(h, t) \models \exists x.(p_1[\text{evexp}/c])
\]
\[
= \quad \% \text{ Def. 20}
\]
\[
\exists h_1 : h_1 \text{-variant of } h \text{ and } (h_1, t) \models p_1[\text{evexp}/c]
\]
\[
= \quad \% \text{ Induction}
\]
\[
\exists h_1 : h_1 \text{-variant of } h \text{ and } ((h_1 : c \sim \text{evexp}), t) \models p_1
\]
\[
= \quad \% \text{ h}_3 = (h_1 : c \sim \text{evexp}), h_3 = (h : c \sim \text{evexp})
\]
\[
\text{h}_1 \text{-variant of } h \text{ iff } h_2 \text{-variant of } h_3
\]
\[
\exists h_2 : h_2 \text{-variant of } h_3 \text{ and } (h_2, t) \models p_1
\]
\[
= \quad \% \text{ Def. 20}
\]
\[
((h : c \sim \text{evexp}), t) \models \exists x. p_1
\]

- \( p = \exists c_0. p_1 : \text{for } c_0 \notin \text{evar(evexp)} \cup \{c\} \):

\[
(h, t) \models (\exists c_0. p_1)[\text{evexp}/c]
\]
\[
= \quad \% \text{ Def. 25}
\]
\[
(h, t) \models \exists c_0.(p_1[\text{evexp}/c])
\]
\[
= \quad \% \text{ Def. 20}
\]
\[
\exists h_1 : h_1 \text{-c}_0\text{-variant of } h \text{ and } (h_1, t) \models p_1[\text{evexp}/c]
\]
\[
= \quad \% \text{ Induction}
\]
\[
\exists h_1 : h_1 \text{-c}_0\text{-variant of } h \text{ and } ((h_1 : c \sim \text{evexp}), t) \models p_1
\]
\[
= \quad \% \text{ h}_3 = (h_1 : c \sim \text{evexp}), h_3 = (h : c \sim \text{evexp})
\]
\[
\text{h}_1 \text{-c}_0\text{-variant of } h \text{ iff } h_2 \text{-c}_0\text{-variant of } h_3
\]
\[
\exists h_2 : h_2 \text{-c}_0\text{-variant of } h_3 \text{ and } (h_2, t) \models p_1
\]
\[
= \quad \% \text{ Def. 20}
\]
\[
((h : c \sim \text{evexp}), t) \models \exists c_0. p_1
\]

165
\[ \begin{align*}
p &= \exists n_0.p_1:
\end{align*}\]

\[\begin{align*}
(h,t) &\models (\exists n.p_1)[\text{evexp}/c] \\
&= \% \text{Def. 25} \\
(h,t) &\models \exists n.(p_1[\text{evexp}/c]) \\
&= \% \text{Def. 20} \\
\exists h_1: &\text{ h is n-variant of } h \text{ and } (h_1,t) \models p_1[\text{evexp}/c] \\
&= \% \text{Induction} \\
\exists h_1: &\text{ h is n-variant of } h \text{ and } ((h_1: e \leadsto \text{evexp}), t) \models p_1 \\
&= \% h_2 = (h_1: e \leadsto \text{evexp}), h_3 = (h: e \leadsto \text{evexp}) \\
&\quad \text{h is n-variant of } h \text{ if } h_2 \text{ is n-variant of } h_3 \\
\exists h_2: &\text{ h is n-variant of } h_3 \text{ and } (h_2,t) \models p_1 \\
&= \% \text{Def. 20} \\
((h: e \leadsto \text{evexp}), t) &\models \exists n.p_1
\end{align*}\]

### A.5 Proof of Lemma 3

**Lemma 3**

*Given a machine in DTL \((B, I \land \square T)\) then there exists a semantic machine \(M = (B, I, T)\) such that \(\text{Comp}(M) = \text{Hist}(I \land \square T)\).*

**Proof 19**

Let \(I \triangleq \{\sigma \in \Sigma \mid \exists h: \sigma = \theta_h(0) \land h \models I\}\) and \(T \triangleq \{(\delta, \sigma_0, \sigma_1) \in \Delta \times \Sigma^2 \mid \exists h: \exists t: \text{Step}_h(t) = (\delta, \sigma_0, \sigma_1) \land \text{Step}_h(t) \notin \text{STU} \land h \models \square T\}\) then \(\text{Hist}(I \land \square T) = \text{Comp}(M)\).

**Proof:**

\[\begin{align*}
\text{Hist}(I \land \square T) \\
&\quad = \% \text{Def. 21} \\
&\quad \{h \in \mathcal{H} \mid h \models I \land \square T\} \\
&\quad = \% \text{Def. 20} \\
&\quad \{h \in \mathcal{H} \mid h \models I \land \forall t: (h,t) \models \square T\} \\
&\quad = \% \text{Def. 28, def. of } I \text{ and } T \\
&\quad \{h \in \mathcal{H} \mid \theta_h(0) \in I \land \forall t: \text{Step}_h(t) \in \text{STU} \lor \text{Step}_h(t) \in T\} \\
&\quad = \% \text{Def. 16} \\
&\quad \text{Comp}(M)
\end{align*}\]

### A.6 Proof of Lemma 4

**Lemma 4**

*Given DTL machine specification of a system \((B, I \land \square T \land L)\) then there exists a semantic machine specification \(S = (B, \text{Comp}(M) \land L)\) such that \(\text{Comp}(M) \land L = \text{Hist}(I \land \square T \land L)\).*

**Proof 20**

Let \(I \triangleq \{\sigma \in \Sigma \mid \exists h: \sigma = \theta_h(0) \land h \models I\}\) and \(T \triangleq \{(\delta, \sigma_0, \sigma_1) \in \Delta \times \Sigma^2 \mid \exists h: \exists t: \text{Step}_h(t) = (\delta, \sigma_0, \sigma_1) \land \text{Step}_h(t) \notin \text{STU} \land h \models \square T\}\)
A.7 Proof of Lemma 5

and let \( L \triangleq Hist(L) \) then \( Hist(1 \land \Box T \land L) = \text{Comp}(M) \cap L \). Because of machine closedness \( cl(\text{Comp}(M) \cap Hist(L)) = \text{Comp}(M) \). Proof:

\[
\begin{align*}
\text{Hist}(1 \land \Box T \land L) &= \% \text{ Def. 21} \\
\{ h \in \mathcal{H} \mid h \models 1 \land \Box T \} \cap \text{Hist}(L) &= \% \text{ Def. 20} \\
\{ h \in \mathcal{H} \mid h \models 1 \land \forall t : (h, t) \models T \} \cap \text{Hist}(L) &= \% \text{ Def. 28, def. of } I, T \text{ and } L \\
\{ h \in \mathcal{H} \mid \theta_h(0) \in I \land \forall t : \text{Step}_h(t) \in \text{STU} \lor \text{Step}_h(t) \in T \} \cap L &= \% \text{ Def. 16} \\
\text{Comp}(M) \cap L
\end{align*}
\]

A.7 Proof of Lemma 5

Lemma 5 (Properties of \( \mathcal{O} \) and \( \otimes \))

Given systems \( (B_1, H_0), (B_1, H_1), (B_2, H_2) \) and \( (B_2, H_3) \) then

\[
(a) \quad H_0 \subseteq H_1 \implies H_0 \otimes H_2 \subseteq H_1 \otimes H_2 \\
(b) \quad \mathcal{O}_{X_{12}}(H_1 \otimes H_2) = \mathcal{O}_{X_1}(H_1) \otimes \mathcal{O}_{X_2}(H_2) \\
(c) \quad H_0 \subseteq H_1 \implies \mathcal{O}_{X_1}(H_0) \subseteq \mathcal{O}_{X_1}(H_1) \\
(d) \quad (H_0 \cap H_1) \otimes (H_2 \cap H_3) \subseteq (H_0 \otimes H_2) \cap (H_1 \otimes H_3)
\]

Proof 21

(a) \( H_0 \subseteq H_1 \implies H_0 \otimes H_2 \subseteq H_1 \otimes H_2 \)

\[
\begin{align*}
h \in H_0 \otimes H_2 &= \% \text{ Def. 32} \\
\exists h_1 \in H_0, h_2 \in H_2. \otimes (h, h_1, h_2) \\
\implies h \in H_0 \subseteq H_1 \\
\exists h_1 \in H_1, h_2 \in H_2. \otimes (h, h_1, h_2) \\
&= \% \text{ Def. 32} \\
h \in H_1 \otimes H_2
\end{align*}
\]

(b) \( \mathcal{O}_{X_{12}}(H_1 \otimes H_2) = \mathcal{O}_{X_1}(H_1) \otimes \mathcal{O}_{X_2}(H_2) \)

167
Proofs of Dense Model Theorems

\[ h \in \mathcal{O}_{X_2}(H_1 \otimes H_2) \]
\[ = \quad \% \text{Def. 30} \]
\[ \exists h_3 \in H_1 \otimes H_2 : h X_1 \cup X_2 \text{-variant of } h_3 \]
\[ = \quad \% \text{Def. 32} \]
\[ \exists h_1, h_2, h_3 : h_1 \in \mathcal{O} \wedge h_2 \in \mathcal{O} \wedge \circ(h, h_1, h_2) \]
\[ \wedge h X_1 \cup X_2 \text{-variant of } h_3 \]
\[ = \quad \% \psi = \psi_3, \psi_4 = \psi_1, \psi_5 = \psi_2 \]
\[ \theta_1 \frac{[s]}{[s_1 \setminus X_1 \cup X_2]} = \theta_3 \frac{[s]}{[s_1 \setminus X_1 \cup X_2]}, \]
\[ \theta_1 \frac{[s]}{[s_1 \setminus X_1]} = \theta_1 \frac{[s]}{[s_1 \setminus X_1]}, \]
\[ \theta_2 \frac{[s]}{[s_2 \setminus X_2]} = \theta_2 \frac{[s]}{[s_2 \setminus X_2]}, \]
\[ \theta_3 = \theta_1, \theta = \theta_4 = \theta_5 \]
\[ \exists h_1, h_2, h_4, h_5 : h_1 \in H_1 \wedge h_2 \in H_2 \wedge \circ(h, h_4, h_5) \]
\[ h_4 X_1 \text{-variant of } h_1 \wedge h_5 X_2 \text{-variant of } h_2 \]
\[ = \quad \% \text{Def. 30} \]
\[ \exists h_4 \in \mathcal{O}_{X_4}(H_1), h_5 \in \mathcal{O}_{X_5}(H_2) : \circ(h, h_4, h_5) \]
\[ = \quad \% \text{Def. 32} \]
\[ h \in \mathcal{O}_{X_2}(H_1) \otimes \mathcal{O}_{X_2}(H_2) \]

(c) \[ H_0 \subseteq H_1 \text{ implies } \mathcal{O}_{X_1}(H_0) \subseteq \mathcal{O}_{X_1}(H_1) \]
\[ = \quad \% \text{Def. 30} \]
\[ \exists h_1 : h_1 \in H_0 \wedge h X_1 \text{-variant of } h_1 \]
\[ \rightarrow \quad \% \mathcal{O}_{X_1}(H_0) \subseteq H_1 \]
\[ \exists h_1 : h_1 \in H_1 \wedge h X_1 \text{-variant of } h_1 \]
\[ = \quad \% \text{Def. 30} \]
\[ h \in \mathcal{O}_{X_1}(H_1) \]

(d) \[ (H_0 \cap H_1) \otimes (H_2 \cap H_3) \subseteq (H_0 \otimes H_2) \cap (H_1 \otimes H_3) \]
\[ = \quad \% \text{Def. 32} \]
\[ \exists h_1, h_2 : h_1 \in (H_0 \cap H_1) \wedge h_2 \in (H_2 \cap H_3) \wedge \circ(h, h_1, h_2) \]
\[ \rightarrow \quad \% \text{Calculus} \]
\[ \exists h_1, h_2 : h_1 \in H_0 \wedge h_2 \in H_2 \wedge \circ(h, h_1, h_2) \]
\[ \wedge \exists h_1, h_2 : h_1 \in H_1 \wedge h_2 \in H_3 \wedge \circ(h, h_1, h_2) \]
\[ = \quad \% \text{Def. 32} \]
\[ h \in H_0 \otimes H_2 \wedge h \in H_1 \otimes H_3 \]

A.8 Proof of Theorem 3

**Theorem 3 (Compositional refinement)**

Given concrete systems \( S_i = (B_i, H_i) \) \((i = 1, 2)\) and abstract systems \( S_j = (B_j, H_j) \) \((j = 3, 4)\) such that \( \mathcal{O}(B_1) = \mathcal{O}(B_3) \) and \( \mathcal{O}(B_2) = \mathcal{O}(B_4) \) then \( S_1 \text{ ref } S_3 \) and \( S_2 \text{ ref } S_4 \) implies \( S_1 \parallel S_2 \text{ ref } S_3 \parallel S_4 \).
A.9 Proof of Lemma 6

Lemma 6
Given DTL machine specification $\mathcal{S} = (B, I \land \square T \land L)$ then $\mathcal{O}_X(Hist(I \land \square T \land L)) = Hist((\exists X . (I \land \square T \land L)))$

Proof 23

$$Hist((\exists X . (I \land \square T \land L)))$$

$\equiv \%$ Def. 21

$\{ h \mid h \models (\exists X . (I \land \square T \land L))\}$

$\equiv \%$ Def. 20

$\{ h \mid \exists h_1 : h \land h_1 \models I \land \square T \land L\}$

$\equiv \%$ Def. 21

$\{ h \mid \exists h_1 : h_1 \text{ X-variant of } h \land h_1 \in Hist(I \land \square T \land L)\}$

$\equiv \%$ Def. 30

$\mathcal{O}_X(Hist(I \land \square T \land L))$

A.10 Proof of Theorem 4

Theorem 4 (Refinement of machine specifications)
Given concrete machine specification $\mathcal{S}_c \triangleq (B_c, I_c \land \square T_c \land L_c)$ where $B_c \triangleq (B_c^P, (V_c, X_c))$ and abstract machine specification $\mathcal{S}_a \triangleq (B_a, I_a \land \square T_a \land L_a)$ where $B_a \triangleq (B_a^P, (V_a, X_a))$. Then $\mathcal{S}_c$ refines $\mathcal{S}_a$ denoted $\mathcal{S}_c \text{ ref } \mathcal{S}_a$ if

$$\mathcal{O}(B_c) = \mathcal{O}(B_a) \text{ and } (\exists X_c . (I_c \land \square T_c \land L_c)) \rightarrow (\exists X_a . (I_a \land \square T_a \land L_a))$$

169
Proof 24

\begin{align*}
\mathcal{S}_c & \text{ ref } \mathcal{S}_a \\
\mathcal{O}(B_c) &= \mathcal{O}(B_a) \\
\mathcal{O}_{X_c}(Hist(I_c \land \Box T_c \land L_c)) &\subseteq \mathcal{O}_{X_a}(Hist(I_a \land \Box T_a \land L_a)) \\
\mathcal{O}(B_c) &= \mathcal{O}(B_a) \\
Hist((\exists X_c . \, (I_c \land \Box T_c \land L_c))) &\subseteq Hist((\exists X_a . \, (I_a \land \Box T_a \land L_a))) \\
\mathcal{O}(B_c) &= \mathcal{O}(B_a) \\
(\exists X_c . \, (I_c \land \Box T_c \land L_c)) &\rightarrow (\exists X_a . \, (I_a \land \Box T_a \land L_a))
\end{align*}

A.11 Proof of Theorem 5

Theorem 5 (Semantic merge is almost conjunction)

Given machine system specifications \( (B_i, I_i \land \Box T_i \land L_i) \) where \( B_i \overset{\Delta}{=} (\text{In}_{i}, \text{Out}_{i}, (V_i, X_i)) \), for \( i = 1, 2 \) and composed machine system specification as in definition 35, i.e., \( (B, H) \) where \( H \overset{\Delta}{=} \exists e_1, e_2, B_1 \circ B_2 \, (e, e_1, e_2) \land (I_1 \land \Box T_1 \land L_1)[e_1/e] \land (I_2 \land \Box T_2 \land L_2)[e_2/e] \) and

\begin{align*}
Hist(I_1 \land \Box T_1 \land L_1) \otimes Hist(I_2 \land \Box T_2 \land L_2) &= Hist(H)
\end{align*}

Proof 25

\begin{align*}
h \in Hist(H) \\
\Rightarrow & \text{ Def. 20} \\
\exists h_1 : h_1 \{e_1, e_2\}-variant of h \land \\
h_1 \models B_1 \circ B_2 \, (e, e_1, e_2) \land (I_1 \land \Box T_1 \land L_1)[e_1/e] \land (I_2 \land \Box T_2 \land L_2)[e_2/e] \\
\Rightarrow & \text{ Def. 20 and 25} \\
\exists h_1 : h_1 \{e_1, e_2\}-variant of h \land \\
h_1 \models B_1 \circ B_2 \, (e, e_1, e_2) \land \\
(h : e \rightarrow e_1) \models I_1 \land \Box T_1 \land L_1 \land \\
(h : e \rightarrow e_2) \models I_2 \land \Box T_2 \land L_2 \\
\Rightarrow & \text{ Def. 21} \\
h \in Hist(I_1 \land \Box T_1 \land L_1) \otimes Hist(I_2 \land \Box T_2 \land L_2)
\end{align*}
A.12 Proof of Theorem 6

Theorem 6
Given machine specification $S \triangleq (B, H)$ and given set of shared variables $V_1 \subseteq V$ then

$$Enc_{V_1}(Hist(H)) = Hist(H \land (\epsilon = e \Rightarrow V'_1 = V_1))$$

Proof 26

\[
\begin{align*}
  h &\in Enc_{V_1}(Hist(H)) \\
  &= \% \text{ Def. 33} \\
  h &\in Hist(H) \land \forall t : \psi(t)(\epsilon) = e \rightarrow \theta(t)|_{V_1} = \lim_{t \rightarrow t_1} \theta(t_1)|_{V_1} \\
  &= \% \text{ Semantics of } (\epsilon = e \Rightarrow V'_1 = V_1) \\
  h &\in Hist(H) \land h \in Hist(\epsilon = e \Rightarrow V'_1 = V_1) \\
  &= \% \text{ Calculus} \\
  h &\in Hist(H \land (\epsilon = e \Rightarrow V'_1 = V_1))
\end{align*}
\]

A.13 Proof of Lemma 7

Lemma 7
Given concrete system $S_c \triangleq (B_c, H_c)$ and abstract system $S_a \triangleq (B_a, H_a)$ s.t. $\mathcal{O}(B_c) = \mathcal{O}(B_a)$. If there exists a refinement mapping from $S_c$ to $S_a$, then $S_c$ ref $S_a$.

Proof 27

\[
\begin{align*}
S_c \text{ ref } S_a \\
  &= \% \text{ Def. 31} \\
  \mathcal{O}(B_c) = \mathcal{O}(B_a) \text{ and} \\
  \mathcal{O}_X(H_c) \subseteq \mathcal{O}_X(H_a) \\
  &\leftarrow \% \mathcal{O}(B_c) = \mathcal{O}(B_a) \\
  \mathcal{O}_X(H_c) \subseteq \mathcal{O}_X(H_a) \\
  &\leftarrow \% \text{ Def. 37}, \ i.e. \mathcal{O}_X(f(H_c)) \subseteq \mathcal{O}_X(H_a), \mathcal{O}_X(H_c) = \mathcal{O}_X(f(H_c)) \\
  &\text{true}
\end{align*}
\]

A.14 Proof of Lemma 8

Lemma 8
Given concrete machine specification $S_c \triangleq (B_c, \text{Comp}(M_c) \cap L_c)$ and given abstract machine specification $S_a \triangleq (B_a, \text{Comp}(M_a) \cap L_a)$ s.t. $\mathcal{O}(B_c) = \mathcal{O}(B_a)$. If there exists a refinement mapping from $S_c$ to $S_a$ then $S_c$ ref $S_a$.

Proof 28

We first prove the following result:
Proofs of Dense Model Theorems

For all \( h_c \in \text{Comp}(M_c) \), there exists a \( h_a \in \text{Comp}(M_a) \) s.t. for all \( t \in \mathbb{R}_{\geq 0}, (\psi_c(t), \theta_c(t)) = (\psi_a(t), \theta(t)_a) \) and \( f(\theta_a(t)) = \theta_a(t) \).

\[
\begin{align*}
    h_c &\in \text{Comp}(M_c) \\
    = &\quad \text{Def. 16} \\
    \theta_c(0) &\in I_c \land \\
    \forall t : (\psi_c(t), \theta_c(t), \lim_{t \to t_i} \theta_c(t_i)) \in T_c \lor (\psi_c(t)(e) \in \{\lambda, i, e\} \land \theta_c(t) = \lim_{t \to t_i} \theta_c(t_i)) \\
    \rightarrow &\quad \text{Def. 38} \\
    f(\theta_a(0)) &\in I_a \\
    \land \forall t : (\psi_a(t), f(\theta_a(t)), \lim_{t \to t_i} f(\theta_a(t_i))) \in T_a \lor \\
    (\psi_a(t)(e) \in \{\lambda, i, e\} \land f(\theta_a(t)) = \lim_{t \to t_i} f(\theta_a(t_i))) \\
    = &\quad \text{Def. 16} \\
    h_a &\in \text{Comp}(M_a) \land \forall t : (\psi_a(t), \theta_a(t)(e) = (\psi_a(t), \theta(t)_a) \land f(\theta_a(t)) = \theta_a(t)
\end{align*}
\]

The proof of Lemma 8 is then as follows:

\[
\begin{align*}
    \mathcal{S}_c &\text{ ref } \mathcal{S}_a \\
    = &\quad \text{Def. 31} \\
    \mathcal{O}(B_c) &\equiv \mathcal{O}(B_a) \land \\
    \mathcal{O}_{X_c}(\text{Comp}(M_c) \cap L_c) &\subseteq \mathcal{O}_{X_a}(\text{Comp}(M_a) \cap L_a) \\
    = &\quad \text{Def. 31} \\
    \mathcal{O}(B_c) &\equiv \mathcal{O}(B_a) \\
    \mathcal{O}_{X_c}(\text{Comp}(M_c) \cap L_c) &\subseteq \mathcal{O}_{X_a}(\text{Comp}(M_a) \cap L_a) \\
    \leftarrow &\quad \text{Def. 37, } f(\text{Comp}(M_c) \cap L_c) \subseteq L_a \\
    \text{Property of } f, f(\text{Comp}(M_c) \cap L_c) &\subseteq f(\text{Comp}(M_c)), \\
    \text{by above result, } f(\text{Comp}(M_c)) &\subseteq \text{Comp}(M_a), \\
    \text{Resulting in, } f(\text{Comp}(M_c) \cap L_c) &\subseteq \text{Comp}(M_a) \cap L_a \\
    \text{From Lemma 5(c), } \mathcal{O}_{X_c}(f(\text{Comp}(M_c) \cap L_c)) &\subseteq \mathcal{O}_{X_a}(\text{Comp}(M_a) \cap L_a) \\
    \text{From Def. 37, } \mathcal{O}_{X_c}(\text{Comp}(M_c) \cap L_c) &\subseteq \mathcal{O}_{X_a}(f(\text{Comp}(M_c) \cap L_c))
\end{align*}
\]

true

A.15 Proof of Theorem 7

Theorem 7 (Compositional relative refinement)

Given concrete systems \( \mathcal{S}_i = (B_i, H_i) \) \((i = 1, 2)\) and given set \( W_c \) constraining \( B_{12} \) (the basis of \( \mathcal{S}_1 \parallel \mathcal{S}_2 \)). And given abstract systems \( \mathcal{S}_j = (B_j, H_j) \) \((j = 3, 4)\) and given set \( W_a \) constraining \( B_{34} \) (the basis of \( \mathcal{S}_3 \parallel \mathcal{S}_4 \)). Then the following holds:

\[
\begin{align*}
    H_1 \otimes H_2 \cap W_{c1} \otimes W_{c2} &\subseteq (H_1 \cap W_{c1}) \otimes (H_2 \cap W_{c2}) \\
    W_c &\subseteq W_{c1} \otimes W_{c2} \\
    W_{a3} \otimes W_{a4} &\subseteq W_a \\
    \mathcal{S}_1 \restriction W_{c1} \mathcal{S}_3 &\quad W_{c1} \text{ constraining } B_i \quad (i=1,2) \\
    \mathcal{S}_2 \restriction W_{c2} \mathcal{S}_4 &\quad W_{a3} \text{ constraining } B_j \quad (j=3,4)
\end{align*}
\]

\[
\begin{align*}
    \mathcal{S}_1 \parallel \mathcal{S}_2 \restriction W_c \mathcal{S}_3 \parallel \mathcal{S}_4
\end{align*}
\]

172
A.16 Proof of Lemma 9

**Proof 29**

Assume agreement on the bases. Then according to Def. 39 and 40 we must infer from the assumptions that $O_{\lambda_1} (H_1 \otimes H_2 \cap W_e) \subseteq O_{\lambda_2} (H_3 \otimes H_4 \cap W_a)$.

\[
O_{\lambda_1} (H_1 \otimes H_2 \cap W_e) \\
\subseteq \% \ H_1 \otimes H_2 \cap W_e \subseteq (H_1 \cap W_{e1}) \otimes (H_2 \cap W_{e2}) \\
W_e \subseteq W_{e1} \otimes W_{e2} \\
\text{Lemma 5(c)} \\
O_{\lambda_1} (H_1 \cap W_{e1}) \otimes (H_2 \cap W_{e2})) \\
\subseteq \% \ S_1_{w_1} \ref \, W_{e1} \, S_3 \\
S_2_{w_2} \ref \, W_{e2} \, S_4 \\
\text{Lemma 5(a), (b)} \\
O_{\lambda_2} (H_3 \cap W_{e3}) \otimes (H_4 \cap W_{e4})) \\
\subseteq \% \ \text{Lemma 5(c), (d)} \\
O_{\lambda_2} (H_3 \otimes H_4 \cap W_{e3} \otimes W_{e4}) \\
\subseteq \% \ W_{e3} \otimes W_{e4} \subseteq W_a \\
\text{Lemma 5(c)} \\
O_{\lambda_2} (H_3 \otimes H_4 \cap W_a)
\]

A.16 Proof of Lemma 9

**Lemma 9**

Given systems $S_i = (B_i, H_i)$ and sets $W_i$ constraining $B_i$ ($i = 1, 2$) with no restrictions on the event variables. Then the following holds:

\[
(H_1 \cap W_1) \otimes (H_2 \cap W_2) = H_1 \otimes H_2 \cap W_1 \otimes W_2
\]

**Proof 30**

From Lemma 5(d) we infer $(H_1 \cap W_1) \otimes (H_2 \cap W_2) \subseteq H_1 \otimes H_2 \cap W_1 \otimes W_2$ so we must prove $H_1 \otimes H_2 \cap W_1 \otimes W_2 \subseteq (H_1 \cap W_1) \otimes (H_2 \cap W_2)$.

\[
\begin{align*}
h \in H_1 \otimes H_2 \cap W_1 \otimes W_2 &= \% \ \text{Def.32} \\
\exists h_1, h_2 \colon h_1, h_2 \in H_1 \land h_2 \in H_2 \land \bigotimes(h, h_1, h_2) \\
\land \exists h_3, h_4 \colon h_3, h_4 \in W_1 \land h_4 \in W_2 \land \bigotimes(h, h_3, h_4) \\
\rightarrow \% \ \text{W; puts no restriction on e variables,} \ \theta = \theta_3 = \theta_4 \\
\exists h_1, h_2 \colon h_1 \in H_1 \land h_2 \in H_2 \land \bigotimes(h, h_1, h_2) \\
\land h \in W_1 \land h \in W_2 \\
\land h \in W_1 \land h \in W_2 \\
&= \% \ \text{Calc.} \\
\exists h_1, h_2 \colon h_1, h_2 \in H_1 \land h_2 \in H_2 \land h \in W_1 \land h \in W_2 \land \bigotimes(h, h_1, h_2) \\
\rightarrow \% \ \text{W; puts no restriction on e variables,} \ \theta = \theta_1 = \theta_2, \\
\text{W; constrains} B_i \\
\exists h_1, h_2 \colon h_1 \in H_1 \land W_1 \land h_2 \in H_2 \cap W_2 \land \bigotimes(h, h_1, h_2) \\
&= \% \ \text{Def.32} \\
h \in (H_1 \cap W_1) \otimes (H_2 \cap W_2)
\]
A.17 Proof of Lemma 10

Lemma 10
Given concrete systems $S_i = (B_i, H_i)$ ($i = 1, 2$) and given set $W_c$ constraining $B_{12}$. And given abstract systems $S_j = (B_j, H_j)$ ($j = 3, 4$) and given set $W_a$ constraining $B_{34}$ without restricting the $\epsilon$ variables. Then the following holds:

$$H_1 \otimes H_2 \cap W_{c1} \otimes W_{c2} \subseteq (H_1 \cap W_{c1}) \otimes (H_2 \cap W_{c2})$$
$$W_c \subseteq W_{c1} \otimes W_{c2}$$
$$S_{1 \text{ ref } W_a S_2}$$
$$W_{c1}\text{ constraining } B_i \ (i=1,2)$$

Proof 31
Assume agreement on the bases. Then according to Def. 39 and 40 we must infer from the assumptions that $O_{X_{12}}(H_1 \otimes H_2 \cap W_c) \subseteq O_{X_{34}}(H_3 \otimes H_4 \cap W_a)$. We will first prove the following:

$$O_{X_{34}}((H_3 \cap W_a) \otimes (H_4 \cap W_a)) \subseteq O_{X_{34}}(H_3 \otimes H_4 \cap W_a)$$

$$h \in O_{X_{34}}((H_3 \cap W_a) \otimes (H_4 \cap W_a))$$
$$= \text{ % Def. 30}$$
$$\exists h_1 : h_1 \in (H_3 \cap W_a) \otimes (H_4 \cap W_a) \land h \text{ X}_{34}\text{-variant of } h_1$$
$$= \text{ % Def. 32}$$
$$\exists h_1 : (\exists h_3, h_4 : h_3 \in (H_3 \cap W_a) \land h_4 \in (H_4 \cap W_a) \land \bigcirc(h_1, h_3, h_4))$$
$$\land h \text{ X}_{34}\text{-variant of } h_1$$
$$\rightarrow \text{ % } \theta_1 = \theta_3 = \theta_4, W_a \text{ doesn't restrict the } \epsilon \text{ variables}$$
$$\exists h_1 : (\exists h_3, h_4 : h_3 \in H_3 \land h_4 \in H_4 \land h_1 \in W_a \land \bigcirc(h_1, h_3, h_4))$$
$$\land h \text{ X}_{34}\text{-variant of } h_1$$
$$= \text{ % Calc.}$$
$$\exists h_1 : (\exists h_3, h_4 : h_3 \in H_3 \land h_4 \in H_4 \land \bigcirc(h_1, h_3, h_4))$$
$$\land h_1 \in W_a \land h \text{ X}_{34}\text{-variant of } h_1$$
$$= \text{ % Def. 32}$$
$$\exists h_1 : h_1 \in H_3 \otimes H_4 \cap W_a \land h \text{ X}_{34}\text{-variant of } h_1$$
$$= \text{ % Def. 30}$$
$$h \in O_{X_{34}}(H_3 \otimes H_4 \cap W_a)$$
A.18 Proof of Lemma 11

The proof of the Lemma is then as follows:

\[
\begin{align*}
\mathcal{O}_{X_{12}}(H_1 \otimes H_2 \cap W_c) \\
\subseteq & \quad H_1 \otimes H_2 \cap W_{c1} \otimes W_{c2} \subseteq (H_1 \cap W_{c1}) \otimes (H_2 \cap W_{c2}) \\
& \quad W_c \subseteq W_{c1} \otimes W_{c2} \\
& \quad \text{Lemma 5(c)} \\
\mathcal{O}_{X_{12}}((H_1 \cap W_{c1}) \otimes (H_2 \cap W_{c2})) \\
\subseteq & \quad S_1 \text{ref } W_c \ S_3 \\
& \quad S_2 \text{ref } W_c \ S_4 \\
& \quad \text{Lemma 5(a), (b)} \\
\mathcal{O}_{X_{34}}((H_3 \cap W_a) \otimes (H_4 \cap W_a)) \\
\subseteq & \quad \text{Above result} \\
\mathcal{O}_{X_{34}}(H_3 \otimes H_4 \cap W_a)
\end{align*}
\]

A.18 Proof of Lemma 11

Lemma 11

Given sets \( W_i \ (i = 1, 2) \) not restricting the \( c \) variables then

\[
W_1 \otimes W_2 = W_1 \cap W_2.
\]

Proof 32

\[
\begin{align*}
h \in W_1 \otimes W_2 \\
= & \quad \text{Def. 32} \\
\exists h_1, h_2 : h_1 \in W_1 \land h_2 \in W_2 \land \otimes (h, h_1, h_2) \\
= & \quad \% \ W_i \ don't \ restrict \ c \ variables, \ i.e. \\
\otimes (h, h_1, h_2) \leftrightarrow h = h_1 = h_2 \\
\exists h_1, h_2 : h_1 \in W_1 \land h_2 \in W_2 \land h = h_1 = h_2 \\
= & \quad \% \ \text{Calc.} \\
h \in W_1 \cap W_2
\end{align*}
\]

A.19 Proof of Theorem 8

Theorem 8 (Relative refinement of DTL machine specifications)

Given concrete machine specification \( S_c \overset{\triangle}{=} (B_c, I_c \land \square T_c \land L_c) \) and DTL formula \( W_c \) over \( B_c \) and abstract machine specification \( S_a \overset{\triangle}{=} (B_a, I_a \land \square T_a \land L_a) \) and DTL formula \( W_a \) over \( B_a \). Let \( G_c \overset{\triangle}{=} I_c \land \square T_c \land L_c \land W_c \) and \( G_a \overset{\triangle}{=} I_a \land \square T_a \land L_a \land W_a \). Then \( S_c \ \text{Hist}(W_c) \text{ref Hist}(W_a) S_a \) iff

\[
\mathcal{O}(B_c) = \mathcal{O}(B_a) \quad \text{and} \\
\models (\exists X_c . (G_c)) \rightarrow (\exists X_a . (G_a))
\]
Proof 33

\[ S_c \cdot Hist(W_c) \text{ref} Hist(W_a) \cdot S_a \]
\[ = \quad \% \text{Def. 39} \]
\[ \mathcal{O}(B_c) = \mathcal{O}(B_a) \]
\[ \mathcal{O}_{X_c}(Hist(I_c \land \Box T_c \land L_c) \cap Hist(W_c)) \]
\[ \subseteq \mathcal{O}_{X_a}(Hist(I_a \land \Box T_a \land L_a) \cap Hist(W_a)) \]
\[ = \quad \% \text{Def. 21 and 20} \]
\[ \mathcal{O}(B_c) = \mathcal{O}(B_a) \]
\[ \mathcal{O}_{X_c}(Hist(I_c \land \Box T_c \land L_c \land W_c)) \subseteq \mathcal{O}_{X_a}(Hist(I_a \land \Box T_a \land L_a \land W_a)) \]

A.20 Proof of Theorem 9

Theorem 9 (Relative composition corresponds to semantic merge)

Given machine system specifications \( (B_i, I_i \land \Box T_i \land L_i) \) where \( B_i \triangleq ((I_i, \text{Out}_i), (V_i, X_i)) \), and given DTL formulae \( W_i \) over \( B_i \) for \( i = 1, 2 \) and let \( \overline{W} \triangleq (Hist(W_1), Hist(W_2)) \) and given the relative composed system as in Def. 41, i.e., \( (B, H) \) where \( B \triangleq ((I_1 \text{Out} _2 \cup \text{Out}_1, \text{Out}_1 \cup \text{Out}_2 \cup \text{In}_1), (V_1 \cup V_2, X_1 \cup X_2)) \) and \( H \triangleq \exists \epsilon_1, \epsilon_2, (B_1 \ominus B_2 (\epsilon_1, \epsilon_2) \land (I_1 \land \Box T_1 \land L_1 \land W_1)[\epsilon_1/\epsilon] \land (I_2 \land \Box T_2 \land L_2 \land W_2)[\epsilon_2/\epsilon]) \) then

\[ Hist(I_1 \land \Box T_1 \land L_1) \overline{W} Hist(I_2 \land \Box T_2 \land L_2) = Hist(H) \]

Proof 34

\[ Hist(I_1 \land \Box T_1 \land L_1) \overline{Hist(I_2 \land \Box T_2 \land L_2)} \]
\[ = \quad \% \text{Def. 40} \]
\[ (Hist(I_1 \land \Box T_1 \land L_1) \cap Hist(W_1)) \otimes (Hist(I_2 \land \Box T_2 \land L_2) \cap Hist(W_2)) \]
\[ = \quad \% \text{Def. 21 and 20} \]
\[ (Hist(I_1 \land \Box T_1 \land L_1 \land W_1)) \otimes (Hist(I_2 \land \Box T_2 \land L_2 \land W_2)) \]
\[ = \quad \% \text{Theorem 5, def. of H} \]
\[ Hist(H) \]

A.21 Proof of Lemma 12

Lemma 12

Given machines \( M \triangleq (B, I, T) \) and \( M_1 \triangleq (B, I_1, T_1) \). Define machine \( M_2 \) as \( (B, I_2, T_2) \) where \( I_2 \) and \( T_2 \) are as follows:

- \( I_2 \triangleq I \cap I_1 \), and
- \( T_2 \triangleq T \cap T_1 \).

Then \( \text{Comp}(M_2) = \text{Comp}(M) \cap \text{Comp}(M_1) \).
A.22 Proof of Lemma 13

Proof 35

\[ h \in \text{Comp}(M) \cap \text{Comp}(M_1) \]
\[ = \quad \% \text{Def.} \ 16 \]
\[ \theta(0) \in I \land \theta(0) \in I_1 \]
\[ (\forall t : \text{Step}_h \in T \lor \text{Step}_h \in \text{STU}) \land (\forall t : \text{Step}_h \in T_1 \lor \text{Step}_h \in \text{STU}) \]
\[ = \quad \% \text{Calculus} \]
\[ \theta(0) \in I \land I_1 \]
\[ \forall t : \text{Step}_h \in T \land T_1 \lor \text{Step}_h \in \text{STU} \]
\[ = \quad \% \text{Def.} \ 16 \]
\[ h \in \text{Comp}(M_2) \]

A.22 Proof of Lemma 13

Lemma 13
Given concrete machine specification \( S_c \triangleq (B_c, \text{Comp}(M_c) \cap L_c) \) and set \( W_c = \text{Comp}(M_1) \cap L_1 \), and given abstract machine specification \( S_a \triangleq (B_a, \text{Comp}(M_a) \cap L_a) \) and set \( W_a = \text{Comp}(M_a) \cap L_a \) s.t. \( \Delta(B_c) = \Delta(B_a) \). If there exists a relative refinement mapping from \( S_c \) to \( S_a \) then \( S_c \downarrow W_c \ref \downarrow W_a S_a \).

Proof 36
We first prove the following result:
For all \( h_c \in \text{Comp}(M_c) \cap \text{Comp}(M_1) \), there exists a \( h_a \in \text{Comp}(M_a) \cap \text{Comp}(M_1) \) s.t. for all \( t \in \mathbb{R}^0 \), \( \langle \psi_c(t), \theta_c(t) \rangle = \langle \psi_a(t), \theta(t)_a \rangle \) and \( f(\theta_c(t)) = \theta_a(t) \).

\[ h_c \in \text{Comp}(M_c) \cap \text{Comp}(M_1) \]
\[ = \quad \% \text{Def.} \ 16 \]
\[ \theta_c(0) \in I_c \land I_1 \land \]
\[ \forall t : \langle \psi_c(t), \theta_c(t), \lim_{i \rightarrow t} \theta_c(t) \rangle \in T_c \cap T_1 \lor \psi_c(t) \in \{ \lambda, i, e \} \land \theta_c(t) = \lim_{i \rightarrow t} \theta_c(t) \]
\[ \rightarrow \quad \% \text{Def.} \ 42 \]
\[ f(\theta_c(0)) \in I_a \land I_1 \land \]
\[ \forall t : \langle \psi_a(t), f(\theta_a(t)), \lim_{i \rightarrow t} f(\theta_a(t)) \rangle \in T_a \cap T_1 \lor \psi_a(t) \in \{ \lambda, i, e \} \land \theta_a(t) = \lim_{i \rightarrow t} f(\theta_a(t)) \]
\[ = \quad \% \text{Def.} \ 16 \]
\[ h_a \in \text{Comp}(M_a) \cap \text{Comp}(M_1) \]
\[ \land \forall t : \langle \psi_a(t), \theta_a(t) \rangle = \langle \psi_a(t), \theta(t)_a \rangle \land f(\theta_a(t)) = \theta_a(t) \]
The proof of Lemma 13 is then as follows:

\[
\mathcal{S}_w \text{ ref } W. c \subseteq \mathcal{S}_a \\
\implies \% \text{ Def. } 39, \text{ Def. } w. c \text{ and } w. a \\
\mathcal{D}(B_c) = \mathcal{D}(B_a) \text{ and} \\
\mathcal{O}_{\mathcal{X}_c} (\text{Comp}(M_c) \cap \text{Comp}(M_{c1}) \cap L_c \cap L_{c1}) \\
\subseteq \mathcal{O}_{\mathcal{X}_a} (\text{Comp}(M_a) \cap \text{Comp}(M_{a1}) \cap L_a \cap L_{a1}) \\
\implies \% \mathcal{D}(B_c) = \mathcal{D}(B_a) \\
\mathcal{O}_{\mathcal{X}_c} (\text{Comp}(M_c) \cap \text{Comp}(M_{c1}) \cap L_c \cap L_{c1}) \\
\subseteq \mathcal{O}_{\mathcal{X}_a} (\text{Comp}(M_a) \cap \text{Comp}(M_{a1}) \cap L_a \cap L_{a1}) \\
\leftarrow \% \text{ From Def. } 42, \\
f(\text{Comp}(M_c) \cap \text{Comp}(M_{c1}) \cap L_c \cap L_{c1}) \subseteq L_a \cap L_{a1} \\
\text{Property of } f, \\
f(\text{Comp}(M_c) \cap \text{Comp}(M_{c1}) \cap L_c \cap L_{c1}) \subseteq f(\text{Comp}(M_c) \cap \text{Comp}(M_{c1})), \\
\text{by above result,} \\
f(\text{Comp}(M_c) \cap \text{Comp}(M_{c1})) \subseteq \text{Comp}(M_a) \cap \text{Comp}(M_{a1}), \\
\text{Resulting in,} \\
f(\text{Comp}(M_c) \cap \text{Comp}(M_{c1}) \cap L_c \cap L_{c1}) \\
\subseteq \text{Comp}(M_a) \cap \text{Comp}(M_{a1}) \cap L_a \cap L_{a1} \\
\text{From Lemma } 5(c), \\
\mathcal{O}_{\mathcal{X}_c} (f(\text{Comp}(M_c) \cap \text{Comp}(M_{c1}) \cap L_c \cap L_{c1})) \\
\subseteq \mathcal{O}_{\mathcal{X}_a} (\text{Comp}(M_a) \cap \text{Comp}(M_{a1}) \cap L_a \cap L_{a1}), \\
\text{From Def. } 42, \\
\mathcal{O}_{\mathcal{X}_c} (\text{Comp}(M_c) \cap \text{Comp}(M_{c1}) \cap L_c \cap L_{c1}) \\
= \mathcal{O}_{\mathcal{X}_a} (f(\text{Comp}(M_c) \cap \text{Comp}(M_{c1}) \cap L_c \cap L_{c1})) \\
\text{true}
\]