Designs with angelic nondeterminism

Pedro Ribeiro

Department of Computer Science
University of York

Technical Report
Revision: 704f887 (2014-02-04)

Supervisor: Ana Cavalcanti

February 2013
Abstract

The Unifying Theories of Programming (UTP) of Hoare and He is a predicative framework of relations suitable for the rigorous study of different programming paradigms. It promotes the reuse of results through the linking of theories. Particular aspects of programs can also be studied in isolation.

In the UTP, the theory of designs provides not only a model for terminating programs (where pre and postcondition pairs can be specified), but also a basis for characterising state-rich concurrent and reactive programs. These are programs whose interactions with the environment cannot simply be described by relations between inputs and outputs. In this context, process calculi such as Communicating Sequential Processes (CSP) and Circus have been given semantics in the UTP through the theory of reactive designs.

Angelic nondeterminism is a useful specification construct that allows for a high degree of abstraction. It has traditionally been studied within the refinement calculus. Previous work has proposed a theory of angelic nondeterminism in the UTP through a predicative model of binary multirelations. Such models, however, can only model terminating programs. In this report we propose a new UTP theory of designs with angelic nondeterminism with the long-term aim of developing a model for process calculi.
## Contents

1 Introduction

1.1 Motivation ........................................ 5
1.2 Objectives ........................................ 7
1.3 Overview of theories .............................. 7
1.4 Outline ........................................... 9

2 UTP

2.1 Theories ........................................... 10
2.1.1 Alphabets ...................................... 11
2.1.2 Healthiness conditions ........................ 11
2.1.3 Refinement ...................................... 12
2.1.4 Operators ...................................... 13

2.2 Designs ............................................ 15
2.2.1 Alphabet ....................................... 15
2.2.2 Healthiness conditions ........................ 15
2.2.3 Operators ...................................... 19
2.2.4 Refinement ...................................... 20

2.3 Linking theories ................................... 20
2.3.1 Subset theories .................................. 20
2.3.2 Bijective links .................................. 21
2.3.3 Galois connections ............................. 21

2.4 Final considerations .............................. 23

3 Binary multirelations .............................. 24

3.1 Introduction ...................................... 24
3.2 Healthiness conditions ........................... 25
3.3 Refinement ordering .............................. 27
3.4 Operators ........................................ 28
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>Predicative model</td>
<td>221</td>
</tr>
<tr>
<td>C.1</td>
<td>d2bmb</td>
<td>221</td>
</tr>
<tr>
<td>C.2</td>
<td>bmb2d</td>
<td>228</td>
</tr>
<tr>
<td>C.3</td>
<td>Other lemmas</td>
<td>242</td>
</tr>
<tr>
<td>D</td>
<td>PBMH</td>
<td>245</td>
</tr>
<tr>
<td>D.1</td>
<td>Properties</td>
<td>245</td>
</tr>
<tr>
<td>D.2</td>
<td>Distribution properties</td>
<td>246</td>
</tr>
<tr>
<td>D.3</td>
<td>Closure properties</td>
<td>247</td>
</tr>
<tr>
<td>D.4</td>
<td>Lemmas</td>
<td>248</td>
</tr>
<tr>
<td>D.5</td>
<td>Set theory</td>
<td>251</td>
</tr>
<tr>
<td>E</td>
<td>Sequential composition (A)</td>
<td>254</td>
</tr>
<tr>
<td>E.1</td>
<td>Algebraic properties</td>
<td>254</td>
</tr>
<tr>
<td>E.2</td>
<td>Closure properties</td>
<td>255</td>
</tr>
<tr>
<td>E.3</td>
<td>Distributivity with respect to disjunction</td>
<td>258</td>
</tr>
<tr>
<td>E.4</td>
<td>Distributivity with respect to conjunction</td>
<td>259</td>
</tr>
<tr>
<td>E.5</td>
<td>Extreme points</td>
<td>260</td>
</tr>
<tr>
<td>E.6</td>
<td>Algebraic properties and sequential composition</td>
<td>260</td>
</tr>
<tr>
<td>E.7</td>
<td>Skip</td>
<td>261</td>
</tr>
<tr>
<td>E.8</td>
<td>Other lemmas</td>
<td>262</td>
</tr>
<tr>
<td>F</td>
<td>Sequential composition (; Dac)</td>
<td>269</td>
</tr>
<tr>
<td>F.1</td>
<td>Properties</td>
<td>269</td>
</tr>
<tr>
<td>G</td>
<td>Linking theories</td>
<td>271</td>
</tr>
<tr>
<td>G.1</td>
<td>acdash2ac and ac2acdash</td>
<td>271</td>
</tr>
<tr>
<td>G.2</td>
<td>PBMH</td>
<td>272</td>
</tr>
<tr>
<td>G.3</td>
<td>Lemmas</td>
<td>272</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

In this chapter we present the motivation and objectives of this work. The motivation is presented in Section 1.1, while the objectives are discussed in Section 1.2. In Section 1.3 we discuss the overall approach followed in this work and how our model relates to existing theories. Finally the structure of this report is outlined in Section 1.4

1.1 Motivation

The UTP of Hoare and He [1] provides a relational framework suitable for the study of different programming paradigms. Relations are characterised by their alphabet and a predicate that restricts the possible values of the variables in the alphabet. The alphabet consists of program variables and auxiliary variables that capture additional information, such as time. A collection of UTP theories exist for a variety of programming paradigms and techniques such as concurrency [1], logic programming [1], high-order programming [1], object-orientation [2], pointers [3], time [4, 6] and others. The UTP distinguishes itself in the ability to promote unification of results through the linking of theories, while allowing concepts to be studied in isolation.

The theory of designs is the definitive treatment of total correctness for sequential programs in the UTP. It considers an alphabet containing program variables as well as auxiliary variables that capture the start and termination of programs. Designs can be understood as encoding the traditional pre and postcondition pairs. In order to characterise reactive programs, the
relationship between initial and final states is not sufficient. Instead, intermediate information also needs to be recorded [7]. This is captured in the UTP through the theory of reactive programs that includes additional observational variables for this purpose.

The combination of the theory of designs and the theory of reactive processes characterises theories for process calculi such as CSP [8] and Circus [9, 10]. Every predicate of the theory of CSP can be specified as a reactive design [11]. These are designs whose preconditions depend on observations of the final or later values of the variables, and whose termination is not necessarily guaranteed. This corresponds to designs that do not necessarily satisfy the healthiness condition $H_3$ [1], a necessary condition to establish the link with the theory of CSP [1].

Angelic nondeterminism is a useful abstraction in the context of formal specifications. It has traditionally been studied in the refinement calculus [11–13] through the monotonic predicate transformers. There it is defined precisely as the dual of demonic nondeterminism. Its characterisation in a relational setting, such as that of the UTP, however, is more challenging and has required the use of multirelational models [14].

Multirelations are relations that relate initial states to sets of final states. In [15] Rewitzky presents the foundational work on multirelations that allows both forms of nondeterminism to be expressed in the same relational model. The set of final states can either be interpreted as encoding angelic or demonic choices. If the sets of final states encode angelic choices, then the relation between an initial state and sets of final states encodes demonic choices, or vice-versa. The model of up-closed binary multirelations is the most important as it has a lattice-theoretic structure [15].

In [14], Cavalcanti et al. propose a UTP theory based on multirelations that can encode angelic nondeterminism. Although the model in [14] does not make use of the observational variables of the original theory of designs, it captures termination. Its focus on sequential programs makes it not applicable to reactive programs.

Morris and Tyrrell [16–19], and Hesselink [20] have pursued the modelling of both types of nondeterminism at the expression or term level. Their focus is on functional languages. Tyrrell et al. [21] have attempted an axiomatization for an algebra resembling CSP where external choice is referred to as “angelic choice”, however this is different from standard CSP semantics [8].

In summary, despite the different attempts at modelling angelic nondeterminism, to the best of our knowledge, no suitable model has been de-
veloped for process calculi. The model that we propose in this work presents a first step towards addressing this problem.

1.2 Objectives

In light of our discussion, in this work we propose a new UTP theory of designs that is capable of expressing both demonic and angelic nondeterminism. In order to exploit existing theories, it is our aim to develop a theory that uses the auxiliary variables of the original theory of designs. Furthermore, as a prerequisite for modelling reactive programs, such a theory needs to encompass designs whose preconditions refer to the value of final states.

In addition, it is essential that we can validate the model we propose with respect to the existing theories. Following the spirit of the UTP, we explore the relationship with both the theory of binary multirelations [15] and that of [14] by establishing links with them.

In our account we define the basic operators of the new theory and prove expected properties based on results from the literature. Since we provide a new model where preconditions may refer to the final set of states, not all results are immediately obvious. This is the case, for instance, for the sequential composition operator.

1.3 Overview of theories

As an aid to the development of our theory, we develop an extended model of binary multirelations. This isomorphic model provides insights into the definition of certain aspects of the theory, such as the sequential composition operator, whose definition is not trivial.

Below we provide an overview of the relationship between the theory proposed and an extended binary multirelational model. Their respective relationship with each of the established models of [14] and [15] is also discussed. The overall relationship between the theories is illustrated in Figure 1.1. Each theory is named after its characterising healthiness condition and the respective isomorphisms are established by pairs of functions. The definition of these is established in Chapters 3 to 5 while a full account of the existing theories can be found in [14, 15].

In the UTP the theory of designs is characterised by the healthiness
conditions $H_1$ and $H_2$ [1]. The theory that we propose is, in addition, characterised by the healthiness condition $A$. It is based on that of [14], whose only healthiness condition is specified by the function $PBMH$.

Since the precondition of reactive designs may impose requirements on final states, this is also allowed in our theory. As a result, it becomes possible for designs to specify sets of final states available for angelic choice, even when termination is not guaranteed. This means that designs in our theory do not necessarily satisfy the healthiness condition $H_3$ of designs.

In order to motivate the development of the new model, we develop an isomorphic model that can describe $A$-healthy designs as an extended version of binary multirelations. The difference with respect to the original model of binary multirelations [15] is that we can distinguish sets of final states that terminate from those that may not terminate. This binary multirelational model is characterised by the healthiness conditions $BMH_0$-$BMH_2$. Its
subset that corresponds to the original binary multirelations is characterised, in addition, by the healthiness condition BMH3.

Finally, in this report we establish that both models that we propose are isomorphic through the pair of functions $d2bmb$ and $bmb2d$. In the following section we focus on subsets of interest of both models and their relationship with the existing theories.

Cavalcanti et al. [14] establish that their UTP model is isomorphic to the model of up-closed binary multirelations, whose healthiness condition we denote as BMH. This relationship is established by a pair of functions, $p2bm$ and $bm2p$ [14], respectively, which we include as part of Figure 1.1.

The functions $d2pbmh$ and $pbmh2d$ establish that the model of A-healthy designs that also satisfy the healthiness condition H3 is isomorphic to that of [14]. Finally, the pair of functions $bmb2bm$ and $bmb2bm$ establish that the subset of the extended binary multirelational model that also satisfies BMH3 is isomorphic to the original model of binary multirelations [15].

This concludes our overview on how the theory that we propose relates to both the extended binary multirelational model and the existing theories.

1.4 Outline

In Chapter 2 the UTP is introduced based on the full account in [1]. The general notions of UTP theories are presented, followed by the theory of designs. We also briefly explain how theories can be related in the UTP.

Chapter 3 introduces the original theory of binary multirelations [15]. This includes the healthiness conditions, the refinement ordering and the main operators of the theory.

In Chapter 4 we introduce an extended model of binary multirelations that can cater for sets of final states that may not terminate. The healthiness conditions are defined and the main operators presented. In addition, we study the subset that is isomorphic to the original theory of binary multirelations.

Chapter 5 describes the new UTP theory of designs with angelic nondeterminism. It introduces the healthiness conditions and defines the main operators. Likewise, we also study the subset that is isomorphic with the model of [14].

Finally, in Chapter 6 we present the conclusions of this work, including indications for future work.
In this chapter we introduce the underlying mathematical theory [1] used in the definition of the theories of interest: the UTP. We begin by characterising the components of UTP theories in Section 2.1. We then focus our attention on the theory of designs in Section 2.2. Finally, we explain how theories can be linked in Section 2.3. A full account on the UTP and the theory of designs can be found in [1, 22].

2.1 Theories

The UTP of Hoare and He [1] is a relational mathematically rigorous approach to characterising and reasoning about programs based on the principle of observation. The UTP promotes unification while allowing different aspects of programs to be considered in isolation. In [1] a collection of theories are presented that target multiple aspects of different programming paradigms, such as functionality, concurrency, logic programming and high-order programming. Recent publications have added to the strength of the UTP by proposing new theories capable of handling angelic nondeterminism [23], object-orientation [2], pointers [3], time [4–6] and others.

A UTP theory is characterised by three main components: an alphabet, a set of healthiness conditions and a set of operators. In Section 2.1.1 we introduce the notion of an alphabet. In Section 2.1.2 we discuss how the healthiness conditions characterise a theory. Finally, in Section 2.1.3 the core notion of refinement in the UTP is explained followed by the operators of theories in Section 2.1.4.
2.1.1 Alphabets

The alphabet of a UTP theory consists of a set of variables that can take values corresponding to observations made of a program behaviour. These can be either program variables, or alternatively, auxiliary variables that capture information like termination, execution time, and so on. Similar to the conventions of Z, in the UTP initial states are characterised by a set of undashed variables (for example, the set: \{ok, v\}), while final or subsequent states are characterised by a set of dashed variables (for example, the corresponding set: \{(ok', v')\}).

A UTP relation consists of an alphabet and a logical predicate over the variables in its alphabet that describes the relationship between initial and after states. For example, in the case of a program whose only purpose is incrementing the initial value of \(x\) we could describe it using the relation: \(x' = x + 1\). This relation concisely describes all pairs of values \((x, x')\) that satisfy the given predicate. Thus relations characterise the possible observations of a program.

The alphabet of a relation is split into two disjoint subsets: the set of undashed variables characterises the input values while the set of dashed variables characterises the after values. For a relation \(R\) these are specified by \(\text{in}\alpha(R)\) and \(\text{out}\alpha(R)\), for the input and output alphabets, respectively.

A relation is homogeneous if and only if the input and output alphabets are exactly the same, except for the fact that variables are undashed and dashed in either set, respectively. This is formally captured by the following definition, where \((\text{in}\alpha(R))'\) is the set of variables obtained by dashing every variable contained in the set \(\text{in}\alpha(R)\).

**Definition 1 (Homogeneous relation)** A relation \(R\) is homogeneous if and only if \((\text{in}\alpha(R))' = \text{out}\alpha(R)\).

When defining a theory it is also necessary to restrict the set of predicates that are valid in a given theory. This is addressed by defining healthiness conditions.

2.1.2 Healthiness conditions

In the UTP the set of predicates valid in a certain theory is defined by what are known as healthiness conditions. These are normally specified by
idempotent monotonic functions whose fixed points are the valid predicates of the theory. These properties ensure that correctness is preserved through refinement.

For instance, in the context of theories concerning time, it is often possible to make observations of a system in discrete-time units using a variable $t$. It is expected that any plausible theory describing such a system must guarantee that time is increasingly monotonic, thus this can be enforced by defining the healthiness condition $HC$.

**Example 1**

$$HC(P) \equiv P \land t \leq t'$$

This healthiness condition is defined in terms of conjunction, so it is called a conjunctive healthiness condition [3]. A general result on conjunctive healthiness conditions [3] enables us to establish that $HC$ is idempotent and monotonic with respect to the refinement ordering. An observation in this theory is valid if and only if it is a fixed point of $HC$.

### 2.1.3 Refinement

The theory of relations forms a complete lattice [1], where the ordering is given by (reverse) universal implication. The top of the lattice is $false$ and the bottom is $true$. This ordering corresponds to the notion of refinement. Its definition is presented below, where the square brackets stand for universal quantification over all the variables in the alphabet [1].

**Definition 2 (Refinement)**

$$P \sqsubseteq Q \equiv [Q \Rightarrow P]$$

Refinement can be understood as preserving the notion of correctness in the sense that, if a predicate $Q$ refines $P$, then all possible behaviours exhibited by $Q$ are permitted by $P$. This notion is paramount for the UTP framework and it is the same across the different theories. The relation $true$ imposes no restriction and permits the observation of any value for all variables in the alphabet, while $false$ permits none.
2.1.4 Operators

A UTP theory comprises a number of operators that characterise how the theory may be used algebraically to specify more complex behaviours. In the theory of relations there are a number of core operators that correspond to typical constructs found in programming languages, such as assignment (:=), conditional (\( A \triangleleft c \triangleright B \)), and sequential composition (\( ; \)). In what follows we present some of the most important operators of the theory of relations.

Sequential composition

In UTP theories whose relations are homogeneous, sequential composition is defined in a consistent way through the notion of substitution as shown in the following definition.

Definition 3 (Sequential composition)

\[
P ; Q \triangleq \exists v_0 \bullet P[v_0/v'] \land Q[v_0/v]
\]

The intuition here is that the sequential composition of two relations \( P \) and \( Q \) involves some intermediate, unobservable state, whose vector of variables is represented by \( v_0 \). This vector is substituted in place for the final values of \( P \), as represented by \( v' \), as well as substituted for the initial values of \( Q \), as represented by \( v \). It is finally hidden by the existential quantifier.

Skip

An important construct in the relational theory is the program \( \pi_R \), otherwise also known as Skip, whose definition is presented below.

Definition 4 (Skip)

\[
\pi_R \triangleq (v' = v)
\]

This is a program that always terminates successfully and upon termination guarantees that all variables maintain their initial values. The most interesting property of \( \pi_R \) is that it is the left-unit for sequential composition \([\pi]\).
Demonic choice

Due to the lattice-theoretic approach of the UTP, demonic choice (⊓) corresponds to the greatest lower bound of the refinement ordering. This means that its definition is simply disjunction.

Definition 5 (Demonic choice)

\[ P \sqcap Q \equiv P \lor Q \]

Unfortunately the least upper bound, which is conjunction, does not correspond to the notion of angelic choice. As mentioned previously, it is not possible to represent both choices directly within the relational model, unless a binary multirelational model is used [14].

Recursion

Recursion is defined in the UTP as the weakest fixed point. Since we have a complete lattice it is possible to find a complete lattice of fixed points due to a result by Tarski [1, 24]. In the following definition \( F \) is a monotonic function and \( \sqcap \) is the greatest lower bound.

Definition 6 (Recursion)

\[ \mu X \bullet F(X) \equiv \sqcap \{ X \mid [F(X) \subseteq X] \} \]

A non-terminating recursion, such as \( (\mu Y \bullet Y) \), is equated with the bottom of the lattice, \( \text{true} [1] \). Intuitively this means that it does not terminate, but if we sequentially compose this recursion with another program, then it becomes possible to recover from the non-terminating recursion as shown in the following example [22].

Example 2

\[
\begin{align*}
(\mu Y \bullet Y) \ ; \ x' &= 0 & \{\text{Definition of recursion}\} \\
= \sqcap \{ X \mid [(\mu Y \bullet Y)(X) \subseteq X] \} \ ; \ x' &= 0 & \{\text{Function application}\} \\
= \sqcap \{ X \mid [X \subseteq X] \} \ ; \ x' &= 0 & \{\text{Reflexivity of } \subseteq\} \\
= \sqcap \{ X \mid \text{true} \} \ ; \ x' &= 0 & \{\text{Property of } \sqcap\}
\end{align*}
\]
This issue motivated the definition of the theory of designs that we present in the following section.

2.2 Designs

As already mentioned, when considering theories of total correctness for reasoning about programs, the theory of relations is not appropriate due to the fact that it is possible to recover from non-terminating programs successfully [1, 22]. In other words, the bottom of the lattice, true, is not necessarily a left-zero of sequential composition as would be needed. As a result, Hoare and He [1] have introduced the theory of designs, which addresses this issue.

2.2.1 Alphabet

The theory of designs is defined by considering the addition of two auxiliary variables to the alphabet: ok and ok'.

\[ \text{ok, ok'} : \{ \text{true, false} \} \]

Their purpose is to track whether a program has been started, in which case ok is true, and whether a program has successfully terminated, in which case ok' is true.

In the following section we present the healthiness conditions that define the theory of designs. Finally we discuss how designs can be refined.

2.2.2 Healthiness conditions

Any valid predicate of this theory has to obey two basic principles: that no guarantees can be made by a program before it has started, and, that no program may require non-termination. These two principles are formally characterised by the healthiness conditions H1, and H2, respectively [1]. We include their definitions [1] below.
Definition 7 (H1)

\[ \text{H1}(P) \triangleq ok \Rightarrow P \]

The definition of H1 states that any guarantees made by P can only be established once it has started. Otherwise, any observation is permitted and it behaves like the bottom of the lattice, which is the same as the one for relations: \textit{true}.

Definition 8 (H2)

\[ \text{H2}(P) \triangleq \neg P[false/ok'] \Rightarrow (P[true/ok'] \land ok') \]

The definition of H2 states that if it is possible for a program P not to terminate, that is with \textit{ok'} being \textit{false}, then it must also be possible for it to terminate, that is with \textit{ok'} being \textit{true}. The definition presented here is equivalent to that originally presented by Hoare and He [1], but instead considers H2 in isolation. In Appendix A we prove that it is equivalent.

A predicate that is both H1 and H2 satisfies the following property of designs.

Law 2.2.1 (H1 $\circ$ H2)

\[ \text{H1} \circ \text{H2}(P) = (ok \land \neg P[false/ok']) \Rightarrow (P[true/ok'] \land ok') \]

Here the design is split into two parts: a precondition and a postcondition. It is defined using the notation of Hoare and He [1] as shown in the following definition.

Definition 9 (Design)

\[ (P \vdash Q) \triangleq (ok \land P) \Rightarrow (ok' \land Q) \]

In fact, a design is more commonly written using the following notation, where we use the shorthand notation \(P^a = P[a/ok']\), with \(t = true\) and \(f = false\), as introduced by Woodcock and Cavalcanti [22].

Law 2.2.2 (Design) A predicate P is a design if and only if it can be written in the following form

\[ \text{H1} \circ \text{H2}(P) = (\neg P^f \vdash P^t) \]
It is worth noting that the functions $H_1$ and $H_2$ (and indeed all of the healthiness conditions of designs) are idempotent and monotonic with respect to refinement [1]. Furthermore none of the proofs establishing these results rely on the property of homogeneity. Therefore it is possible to define a non-homogeneous theory of designs.

Hoare and He [1] identified another two healthiness conditions of interest which we discuss further below. The third healthiness condition $H_3$ requires $\Pi_D$, the Skip of designs, to be a right-unit for sequential composition [1].

Definition 10 ($\Pi_D$)

$$\Pi_D \triangleq (true \vdash v' = v)$$

Skip is a program that always terminates successfully and does not change the program variables.

Definition 11 (H3)

$$H_3(P) \triangleq P ; \Pi_D$$

From this definition it may not be immediately obvious how designs are further restricted by $H_3$. In fact, it requires the precondition not to have any dashed variables (as confirmed by Theorem 2.2.1). In order to understand the intuition behind it we consider an example of a design that is not $H_3$-healthy.

Example 3

$$(x' \neq 2 \vdash true) \quad \{\text{Definition of designs}\}$$

$$= (ok \land x' \neq 2) \Rightarrow ok' \quad \{\text{Propositional calculus}\}$$

$$= ok \Rightarrow (x' = 2 \lor ok')$$

In this case we have a program that upon having started can either terminate and any final values are permitted, or can assign the value 2 to the variable $x$ and termination is then not required. In the context of a theory of total correctness for sequential programs this is a behaviour that would not normally be expected. However it is worth noting that in the context of reactive processes non $H_3$-designs are important, since there are some requirements imposed on programs even when they diverge [7, 14].
The healthiness condition $H_3$ can also be interpreted as guaranteeing that if a program may not terminate, then it has arbitrary behaviour. Thus a predicate that is $H_3$-healthy is also necessarily $H_2$-healthy [14].

If we expand the definition of $H_3$ by applying the definition of sequential definition for designs we obtain the following result [1, 22].

**Theorem 2.2.1 (P-sequence-$\Pi_D$)**

\[ (\neg P^f \mid P^f) = (\neg P^f \mid P^f) ; \Pi_D \iff \neg P^f = \exists v' \bullet \neg P^f \]

This theorem shows that the value of any dashed variables in $\neg P^f$ must be irrelevant. Therefore any design that is $H_3$-healthy can only have a condition as its precondition, that is, a predicate that only mentions undashed variables, and thus can only impose restrictions on previous programs.

Finally the last healthiness condition of interest is $H_4$ that restricts designs to feasible programs. It is defined by the following algebraic equation [1] that requires that $true$ be a right-zero.

**Definition 12 (H4)**

\[ P ; true = true \]

The intuition here is that this prevents the top of the lattice, Miracle, itself a trivial refinement of any program, from being allowed. In order to understand the reason for this, consider the definition of Miracle.

**Definition 13 (Miracle)**

\[
\text{Miracle} \triangleq (true \vdash false) \quad \{\text{Property of designs}\}
= ok \Rightarrow false \quad \{\text{Propositional calculus}\}
= \neg ok
\]

Miracle represents a program that could never be started ($\neg ok$). Furthermore, if it could, and indeed its precondition makes no restriction, it would establish the impossible: $false$. Any conceivable implementable program must not behave in this way. However, Miracle is an important construct in refinement calculi [14] [22].

For completeness we also provide the definition of the bottom of the lattice of designs, which is called Abort. There are in fact two possible ways of expressing it as a design.
Definition 14 (Abort)

\[
\text{Abort} \triangleq (\text{false} \vdash \text{true}) \quad \{\text{Property of designs}\}
\]

\[
= (\text{false} \land \text{ok}) \Rightarrow \text{ok}' \quad \{\text{Propositional calculus}\}
\]

\[
= (\text{false} \land \text{ok}) \Rightarrow (\text{false} \land \text{ok}') \quad \{\text{Property of designs}\}
\]

\[
= (\text{false} \vdash \text{false})
\]

Abort provides no guarantees at all: it may fail to terminate, and if it does terminate there are no guarantees on the final values. Indeed it is not required to guarantee anything at all since its precondition is false.

2.2.3 Operators

In the following theorems we introduce the meet and join of the lattice of designs as presented in [22]. Like in the lattice of relations, the greatest lower bound corresponds to demonic choice.

Theorem 2.2.2 (Greatest lower bound)

\[
\bigcap_i (P_i \vdash Q_i) = (\bigwedge_i P_i) \vdash (\bigvee_i Q_i)
\]

Theorem 2.2.3 (Least upper bound)

\[
\bigcup_i (P_i \vdash Q_i) = (\bigvee_i P_i) \vdash (\bigvee_i P_i \Rightarrow Q_i)
\]

Sequential composition

The definition of sequential composition for designs can be deduced from Definition 3. Here we present the result as proved in [1, 22].

Theorem 2.2.4 (Sequential composition of designs)

\[
(P_0 \vdash P_1) ; (Q_0 \vdash Q_1) = (\neg (\neg P_0 ; \text{true}) \land \neg (P_1 ; \neg Q_0) \vdash P_1 ; Q_1)
\]

This definition can be interpreted as establishing \(P_1\) followed by \(Q_1\) provided that \(P_0\) holds and \(P_1\) satisfies \(Q_0\). As pointed out in [22] if \(P_0\) is a condition then the definition can be further simplified.
2.2.4 Refinement

As in other UTP theories, the refinement ordering in the theory of designs is the same: universal (reverse) implication. This can be used to establish the following result [22].

Theorem 2.2.5 (Refinement)

\[(P_0 \vdash P_1) \sqsubseteq (Q_0 \vdash Q_1) = [P_0 \land Q_1 \Rightarrow P_1] \land [P_0 \Rightarrow Q_0]\]

Theorem 2.2.5 confirms the intuition about refinement as found in other calculi: preconditions can be weakened while postconditions can be strengthened.

This section concludes our overview of the theory of designs. In the following section we focus on how theories can be related and combined.

2.3 Linking theories

The UTP provides a very powerful framework that allows relationships to be established between different theories. This means that results in different theories can be re-used. We elaborate on some of principles behind the linking of theories in the following sections. A full account is available in [1].

Following the convention of Hoare and He [1] we assume the existence of a pair of functions \(L\) and \(R\) that map one theory into another: \(L\) maps the (potentially) more expressive theory into the (potentially) weaker theory and \(R\) vice-versa.

2.3.1 Subset theories

The simplest form of relationship that can be established is that between subset theories [1]. Consider the case where a theory \(T\) is a subset of \(S\), then it is possible to find a function \(R : T \mapsto S\) which is simply the identity [1]. Defining \(L : S \mapsto T\) for the reverse direction may be slightly more complicated as the subset theory is normally less expressive.

Hoare and He [1] pinpoint the most important properties of such a function \(L : S \mapsto T\): weakening or strengthening, idempotence and ideally monotonicity. As highlighted in [1] monotonicity is not always necessarily observed. We reproduce the respective definitions below.
Definition 15 (Weakening)
\[
\forall X \in S \ni L(X) \subseteq X
\]

Definition 16 (Strengthening)
\[
\forall X \in S \ni X \subseteq L(X)
\]

We follow Hoare and He's convention and refer to a function that is both weakening and idempotent as a *link* and, if it is also monotonic we refer to it as a *retract*.

### 2.3.2 Bijective links

When two theories have equal expressive power, the pair of linking functions between them can be proved to form a bijection. In other words, each function undoes exactly the other and thus as expected the following identities hold.

**Definition 17 (Bijection)** A function L is a bijection if and only if \( R = L^{-1} \), where the inverse function of L, \( L^{-1} \) exists, and the following identities hold for all P.

\[
L \circ R(P) = P \land R \circ L(P) = P
\]

A bijection constitutes the strongest form of relationship between theories. It can apply even when the alphabets are different or when theories are presented in different styles [1]. Indeed this is often what is sought: proving that two theories have exactly the same expressive power, yet their shape may suit different contexts better.

### 2.3.3 Galois connections

Often, though, and as seen previously in subset theories, a theory is more expressive than its counterpart. Therefore the linking function is not a bijection as there has to be some weakening or strengthening in either direction. A pair of functions describing this relationship constitutes what is known as a Galois connection. Here we reproduce the definition of [1] and provide a pictorial illustration in Figure 2.1.
Figure 2.1: Galois connection between two lattices, $S$ and $T$.

**Definition 18 (Galois connection)**  Let $S$ and $T$ be lattices, and let $L : S \mapsto T$ and $R : T \mapsto S$, the pair $(L, R)$ is a Galois connection if and only if for all $X \in S$ and $Y \in T$:

$$R(Y) \sqsubseteq X \iff Y \sqsubseteq L(X)$$

As pointed out earlier, a bijection presents a stronger relationship than a Galois connection. However, it is not the case that every bijection is a Galois connection [1]. Hoare and He [1] give the example of negation whose inverse is precisely itself, however negation is not monotonic.
2.4 Final considerations

The UTP framework provides a way of rigorously formalising programs in a relational setting. A UTP theory consists of an alphabet, a set of of healthiness conditions and a set of operators, whose syntax forms the signature of the theory [1]. The most general theory in the UTP is that of relations. Unfortunately it is not sufficient on its own to appropriately define theories of total correctness for programs.

The theory of designs provides a compromise, where by extending the alphabet with additional observational variables, termination can be characterised appropriately. The set of valid predicates is defined by a set of healthiness conditions: \(H_1\) and \(H_2\) characterise designs, and equally determine a unique syntactic form. The other healthiness conditions, while optional, are also important from the point of view of refinement of sequential programs. However in the context of theories such as those for reactive processes it is essential that we can consider designs that are not necessarily \(H_3\)-healthy.

Finally we have briefly considered how UTP theories can be related. This is achieved by linking functions that can map predicates from one theory into another. When considering theories that have equal expressive power the linking function is a bijection. However, often theories have different expressive power, therefore there must be some weakening or strengthening. In this case the pair of linking functions forms a Galois connection. In addition, it is also possible to establish relationships with sub theories. The importance of these linking functions is that results can be borrowed from other theories and then re-used in different contexts. This forms part of the toolkit that is in the essence of the UTP unification.
Chapter 3

Binary multirelations

In this chapter the theory of binary multirelations [15] is presented. In Section 3.1 the theory is introduced and formally defined. The single healthiness condition of the theory is explored in Section 3.2 along with its characterisation as a fixed point. Section 3.3 describes the refinement ordering and its extreme points. Finally, the operators are presented in Section 3.4.

3.1 Introduction

A binary multirelation, an element of a type named $BM$ here, is a relation between an initial program state and a set of final states, where a $State$ is the type of records with a component for each program variable.

Definition 19

$$BM ::= State \leftrightarrow \mathbb{P} State$$

For instance, the program that assigns the number 1 to the only program variable $x$ when started from any initial state is defined as follows.

Example 4

$$(x := 1)_{BM} \triangleq \{ s : State, ss : \mathbb{P} State \mid (x \mapsto 1) \in ss \}$$

Following [14], the notation $(x \mapsto 1)$ denotes a record whose only component is $x$ and its respective value is 1.
The binary multirelational model is richer than the relational model in that it relates each initial state to a set of final states. This set can be interpreted as either encoding angelic or demonic choices, depending on which model is chosen \[14, 15\]. In our discussion we choose to present a model where the set of final states encodes angelic choices. This deliberate choice is justified in \[14, 25\] as maintaining the refinement order of the isomorphic \[\text{UTP}\] model introduced in \[14\]. Since it is our goal to study an extended version of binary multirelations and its relationship with an equivalent \[\text{UTP}\] model, it is desirable also in our context that the refinement order is maintained.

Demonic choices are encoded by the different ways in which the set of final states can be chosen. For example, the program that angelically assigns the value 1 or 2 to the only program variable \(x\) is specified by the following relation, where \(\sqcup_{BM}\) is the angelic choice operator for binary multirelations.

Example 5

\[(x := 1)_{BM} \sqcup_{BM} (x := 2)_{BM} = \{s : \text{State}, ss : \mathbb{P}\text{State} \mid (x \mapsto 1) \in ss \land (x \mapsto 2) \in ss\}\]

This definition allows any superset of the set \(\{(x \mapsto 1), (x \mapsto 2)\}\) to be chosen. The choice of values 1 and 2 for the program variable \(x\) are available in every set of final states \(ss\), and so are available in every demonic choice.

3.2 Healthiness conditions

In general, not all relations of type \(BM\) are valid. The subset of interest is that of upward-closed binary multirelations \[15, 20\]. The following healthiness condition \[14\] characterises it.

Definition 20 (BMH)

\[\text{BMH} \equiv \forall s : \text{State}; ss_0, ss_1 : \mathbb{P}\text{State} \cdot ((s, ss_0) \in B \land ss_0 \subseteq ss_1) \Rightarrow (s, ss_1) \in B\]

If a particular initial state \(s\) is related to a set of final states \(ss_0\), then it is also related to any superset of \(ss_0\). This means that if it is possible to terminate
in some final state that is in \( ss_0 \), then the addition of any other final states to that same set does not change the final states available for angelic choice, which correspond to those in the distributed intersection of all sets of final states available for demonic choice.

The set of binary multirelations of interest can alternatively be characterised by the fixed points of the following function.

**Definition 21** (bmh\textsubscript{upclosed})

\[
\text{bmh}_{\text{upclosed}}(B) \\
\triangleq \\
\{ s : \text{State}, ss : \mathbb{P}\text{State} \mid \exists ss_0 : \mathbb{P}\text{State} \bullet (s, ss_0) \in B \land ss_0 \subseteq ss \}
\]

This equivalence is established by the following Law

**Law 3.2.1** (bmh\textsubscript{upclosed}-BMH)

\[
\text{BMH} \iff \text{bmh}_{\text{upclosed}}(B) = B
\]

*Proof.*

\[
\text{BMH} \iff \forall s : \text{State}; ss_0, ss_1 : \mathbb{P}\text{State} \bullet ((s, ss_0) \in B \land ss_0 \subseteq ss_1) \Rightarrow (s, ss_1) \in B
\]

\[
\iff \forall s : \text{State}; ss_1 : \mathbb{P}\text{State} \bullet (\exists ss_0 : \mathbb{P}\text{State} \bullet (s, ss_0) \in B \land ss_0 \subseteq ss) \Rightarrow (s, ss_1) \in B
\]

\[
\iff \forall s : \text{State}, ss : \mathbb{P}\text{State} \mid \exists ss_0 : \mathbb{P}\text{State} \bullet (s, ss_0) \in B \land ss_0 \subseteq ss \subseteq B
\]

\[
\iff \forall s : \text{State}, ss : \mathbb{P}\text{State} \mid \exists ss_0 : \mathbb{P}\text{State} \bullet (s, ss_0) \in B \land ss_0 \subseteq ss \}
\]

\[
\iff \{ s : \text{State}, ss : \mathbb{P}\text{State} \mid \exists ss_0 : \mathbb{P}\text{State} \bullet (s, ss_0) \in B \land ss_0 \subseteq ss \} \cup B = B
\]

\[
\iff \text{bmh}_{\text{upclosed}}(B)
\]
The set of fixed points can be used interchangeably with the healthiness condition as it characterises exactly the upward-closed binary multirelations.

### 3.3 Refinement ordering

The refinement order for healthy binary multirelations $B_0$ and $B_1$, as presented in [14] is reproduced below.

**Definition 22** ($\sqsubseteq_{BM}$)

$$B_0 \sqsubseteq_{BM} B_1 \triangleq B_0 \supseteq B_1$$

It is defined as subset inclusion, similarly to the refinement order for set-based relations [14]. This partial order over $BM$ forms a lattice. It allows an increase in the degree of angelic nondeterminism and a decrease in demonic nondeterminism. This aspect is discussed further in Section 3.4.

In what follows we define the extreme points of the lattice as given by the subset ordering. These correspond respectively to the notions of a miraculous program, as defined by $\top_{BM}$, and abort, as defined by $\bot_{BM}$.

**Definition 23** (Miracle)

$$\top_{BM} \triangleq \emptyset$$

**Definition 24** (Abort)

$$\bot_{BM} \triangleq State \times \mathbb{P} State$$

The top of the lattice $\top_{BM}$ is defined as the empty set while the bottom $\bot_{BM}$ is defined as the universal relation. The consequence is that a miraculous program cannot be executed, while abort exhibits arbitrary behaviour for every possible initial state. This allows us to establish the following law.

**Law 3.3.1** (Refinement)

$$\bot_{BM} \sqsubseteq_{BM} B \sqsubseteq_{BM} \top_{BM}$$

**Proof.** Follows from the subset ordering. □
Having presented the refinement ordering and its extreme points, in the following section we introduce the operators of the theory, including interesting properties regarding refinement.

### 3.4 Operators

In this section we present the main operators of the theory of binary multirelations \[15\] and discuss their properties.

#### 3.4.1 Assignment

The assignment operator is defined as follows.

**Definition 25**

\[
(x := e)_{BM} \overset{\triangleq}{=} \{ s : \text{State}, ss : \mathbb{P} \text{State} \mid s \oplus (x \mapsto e) \in ss \}
\]

It relates every initial state \( s \) to every possible set of final states \( ss \), such that \( ss \) includes a state where \( s \) is overridden with a record where \( x \) has the value of the expression \( e \).

#### 3.4.2 Angelic choice

The angelic choice operator is defined as intersection.

**Definition 26** (\( \sqcup_{BM} \))

\[
B_0 \sqcup_{BM} B_1 \overset{\triangleq}{=} B_0 \cap B_1
\]

This operator corresponds to the least upper bound of the lattice. It captures the intuition that the final states available to the angel must be in the intersection of all choices available for demonic choice. Consequently, the operator observes the following law with respect to refinement.

**Law 3.4.1**

\[
B_0 \sqsubseteq_{BM} (B_0 \sqcup_{BM} B_1)
\]
Proof.

\[ B_0 \sqsubseteq_{BM} (B_0 \sqcup_{BM} B_1) \quad \{ \text{Definition of } \sqsubseteq_{BM} \text{ and } \sqcup_{BM} \} \]
\[ = B_0 \supseteq (B_0 \cap B_1) \quad \{ \text{Property of sets} \} \]
\[ = \text{true} \]

As expected, this allows the degree of angelic nondeterminism to be increased. We observe that the proofs shown follow from the original model of [15]. Here we simply prove them as they provide auxiliary results for our discussion.

3.4.3 Demonic choice

The demonic choice operator is precisely defined as the dual of the angelic choice operator by considering set union.

Definition 27 \((\sqcap_{BM})\)

\[ B_0 \sqcap_{BM} B_1 \cong B_0 \sqcup B_1 \]

The sets of final states available for demonic choice correspond to those in either \(B_0\) or \(B_1\). It corresponds to the greatest lower bound of the lattice. Therefore it observes the following law with respect to the refinement order.

Law 3.4.2

\[ (B_0 \sqcap_{BM} B_1) \sqsubseteq_{BM} B_0 \]

Proof.

\[ (B_0 \sqcap_{BM} B_1) \sqsubseteq_{BM} B_0 \quad \{ \text{Definition of } \sqsubseteq_{BM} \text{ and } \sqcup_{BM} \} \]
\[ = (B_0 \cup B_1) \supseteq B_0 \quad \{ \text{Property of sets} \} \]
\[ = \text{true} \]

For an example, we consider the demonic choice over two assignments.
Example 6

\[
(x := 1)_{BM} \cap_{BM} (x := 2)_{BM}
= \{ s : State, ss : \mathbb{P} State \mid s \oplus (x \mapsto 1) \in ss \lor s \oplus (x \mapsto 2) \in ss \}\]

In this case, all initial states \( s \) are related to every set of final states \( ss \) that contains either a component where \( x \) is mapped to 1 or 2, or both. This means that it is impossible for the angel to enforce a particular choice, as the intersection of all sets of final states for a particular initial state, is empty.

The angelic and demonic choice operators distribute over one another.

Law 3.4.3

\[
B_0 \cap_{BM} (B_1 \sqcup_{BM} B_2) = (B_0 \cap_{BM} B_1) \sqcup_{BM} (B_0 \cap_{BM} B_2)
\]

Proof. Follows from the definition of \( \cap_{BM}, \sqcup_{BM} \) and property of sets. \( \square \)

This property follows from the distributive properties of set union and set intersection. This property is equally applicable in the theory of predicate transformers and the UTP model of [14].

3.4.4 Sequential composition

Sequential composition for binary multirelations [14, 15] is defined below.

Definition 28 ( ;_{BM})

\[
B_0 ;_{BM} B_1 \overset{\triangleq}{=} \left\{ s : State, ss_1 : \mathbb{P} State \mid \exists ss_0 : \mathbb{P} State \bullet (s, ss_0) \in B_0 \land ss_0 \subseteq \{ s : State \mid (s, ss_1) \in B_1 \} \right\}
\]

It is defined by considering every initial state \( s \) in \( B_0 \) and set of final states \( ss_1 \), such that there is some intermediate set of states \( ss_0 \) that is related from \( s \) in \( B_0 \), and \( ss_0 \) is a subset of the set of initial states in \( B_1 \) that achieve \( ss_1 \). As noted in [14] for healthy binary multirelations this definition can be simplified as shown in the following law.
Law 3.4.4 (\( ;_{BM\text{-healthy}}\)) Provided \(B_0\) is BMH-healthy.

\[
B_0 ;_{BM} B_1 = \{ s : State, ss : \mathbb{P} \text{State} \mid (s, \{ s : State \mid (s, ss) \in B_1 \}) \in B_0 \}
\]

Proof.

\[
B_0 ;_{BM} B_1 = \{ s : State, ss_1 : \mathbb{P} \text{State} \mid \exists ss_0 : \mathbb{P} \text{State} . (s, ss_0) \in B_0 \land ss_0 \subseteq \{ s : State \mid (s, ss_1) \in B_1 \} \}
\]

\(\Box\)

This definition is used as the basis for the definition of sequential composition in the isomorphic UTP model of [14]. This is also the basis for our interpretation of the definition of sequential composition in the extended binary multirelational model that we present in Chapter 4.

3.5 Final considerations

In this chapter we have introduced the theory of binary multirelations. This model allows the specification of programs that have both angelic and demonic nondeterminism in a relational setting. It is known to be isomorphic to the predicate transformers model [14] [15].

In addition, the model is also isomorphic to the UTP model of [14], a theory of designs with angelic nondeterminism. However, these models can only consider final states that are necessarily terminating. This corresponds to designs with angelic nondeterminism that observe H3.

The binary multirelational theory presented, along with the isomorphic predicative UTP model of [14] provide the basis for developing an extended multirelational theory in the following Chapter 4.
Chapter 4

Binary multirelational model

In this chapter we introduce an extended binary multirelational model that can model sets of final states that are not necessarily terminating. This is achieved by extending the original model of [15], presented in the previous chapter, using an extra symbol that denotes the possibility for non-termination.

The following Section 4.1 introduces the model and formally defines the binary multirelations of interest. In Section 4.2 the healthiness conditions are defined. Their characterisation as fixed points is presented in Section 4.3. In Section 4.4 the refinement order is defined. The operators of the theory are explored in Section 4.5. Finally, Section 4.6 formalizes the relationship between this model and that of [15].

4.1 Introduction

Similar to the original model of binary multirelations, a relation in this model associates to each initial program state a set of final states. The notion of final state, however, is different, as formalised by the following type $BM_{\bot}$.

Definition 29

\[
\begin{align*}
\text{State}_{\bot} & = (\text{State} \cup \{\bot\}) \\
BM_{\bot} & = \text{State} \leftrightarrow \mathcal{P} \text{State}_{\bot}
\end{align*}
\]

Each initial state is related to a set of final states of type $\text{State}_{\bot}$, a final state that may include the symbol $\bot$. This symbol indicates that for a particular
set of final states, the program may or may not terminate. If a set of final states does not contain \( \bot \) then the program must terminate.

For example, consider the program that assigns the value 1 to the variable \( x \) but may or may not terminate. This is specified by the following relation, where \( :=_{BM\bot} \) is the assignment operator that does not require termination.

**Example 7**

\[
(x :=_{BM\bot} 1) = \{ s : State, ss : \mathbb{P} State_\bot \mid s \oplus (x \mapsto 1) \in ss \}
\]

Every initial state \( s \) is associated with a set of final states \( ss \) where the state obtained from \( s \) by overriding the value of the component \( x \) with 1 is included. Since \( ss \) is of type \( State_\bot \), all sets of final states in \( ss \) include those with and without \( \bot \).

It is also possible to specify a program that must terminate for certain sets of final states but not necessarily for others as shown in the following example, where \( \cap_{BM\bot} \) is the demonic choice operator of the theory.

**Example 8**

\[
(x :=_{BM} 1) \cap_{BM\bot} (x :=_{BM\bot} 2) = \\
\left\{ s : State, ss : \mathbb{P} State_\bot \mid (s \oplus (x \mapsto 1) \in ss \land \bot \notin ss) \lor (s \oplus (x \mapsto 2) \in ss) \right\}
\]

Since \( BM \) is in fact a subset of \( BM\bot \), it is possible to use some of the existing operators, such as the terminating assignment operator \( :=_{BM} \). In this case, there is a demonic choice between the terminating assignment of 1 to \( x \), and the assignment of 2 to \( x \) that does not require termination.

Similar to the original theory of binary multirelations, the set of final states encodes the choices available to the angel. The demonic choices are encoded by the different ways in which the set of final states can be chosen.

### 4.2 Healthiness conditions

In this section the healthiness conditions of the theory are introduced as predicates. Their characterisation as fixed points is developed in Section [4.3](#).
4.2.1 BMH0

The first healthiness condition of interest is BMH0. It enforces the upward closure of the original theory of binary multirelations [15] for sets of final states that are necessarily terminating, but in addition enforces a similar property for sets of final states that are not required to terminate.

Definition 30 (BMH0)

\[ \forall s : State, ss_0, ss_1 : P State \bullet ((s, ss_0) \in B \land ss_0 \subseteq ss_1 \land (\bot \in ss_0 \iff \bot \in ss_1)) \Rightarrow (s, ss_1) \in B \]

It states that for every initial state \( s \), and for every set of final states \( ss_0 \) in a relation \( B \), any superset \( ss_1 \) of that final set of states is also associated with \( s \) such that \( \bot \) is in \( ss_0 \) if and only if it is in \( ss_1 \). That is, BMH0 requires the upward closure for sets of final states that terminate, and for those that may or may not terminate, but separately.

The definition of BMH0 can actually be split into two conjunctions as shown in the following Law 4.2.1. BMH is the healthiness condition of the original theory and is defined in the previous Chapter 3.

Law 4.2.1

BMH0

\[ \iff \left( \left( \forall s : State, ss_0, ss_1 : P State \bullet ((s, ss_0) \in B \land ss_0 \subseteq ss_1 \land (\bot \in ss_0 \iff \bot \in ss_1)) \Rightarrow (s, ss_1) \in B \right) \land \text{BMH} \right) \]

Proof.

BMH0 \{Definition of BMH0\}

\[ \iff \left( \forall s : State, ss_0, ss_1 : P State \bullet ((s, ss_0) \in B \land ss_0 \subseteq ss_1 \land (\bot \in ss_0 \iff \bot \in ss_1)) \Rightarrow (s, ss_1) \in B \right) \]

\{Propositional calculus\}

\[ \iff \left( \forall s : State, ss_0, ss_1 : P State \bullet ((s, ss_0) \in B \land ss_0 \subseteq ss_1 \land ((\bot \in ss_0 \land \bot \in ss_1) \lor (\bot \notin ss_1 \land \bot \notin ss_0)) \Rightarrow (s, ss_1) \in B \right) \]
This result confirms that for sets of final states that terminate this healthiness condition enforces BMH exactly as in the original theory of binary multirelations \cite{15}. This ensures that if it is possible to terminate in some final state, then termination is also guaranteed in any superset.

4.2.2 BMH1

The second healthiness condition BMH1 requires that if it possible to choose a set of final states where termination is not guaranteed, then it must also be possible to choose an equivalent set of states where termination is guaranteed. This healthiness condition is similar in nature to H2 in the theory of designs.
Definition 31 (BMH1)

\[ \forall s : State; \ ss : \mathbb{P} \ State_{\bot} \bullet (s, ss \cup \{\bot\}) \in B \Rightarrow (s, ss) \in B \]

If it is possible to reach a set of final states \((ss \cup \{\bot\})\) from some initial state \(s\), where termination is not required, then the set of final states \(ss\), possibly without \(\bot\), so that termination is required is also associated with \(s\).

This healthiness condition excludes relations that only offer sets of final states that may not terminate. Consider the following example.

Example 9

\[ \{ s : State, ss : \mathbb{P} \ State_{\bot} \mid (x \mapsto 1) \in ss \land \bot \in ss \} \]

This relation describes an assignment to the only program variable \(x\) where termination is not guaranteed. However, it discards the inclusive situation where termination may indeed occur. The inclusion of an equivalent final set of states that requires termination does not change the choices available to the angel as it is still impossible to guarantee termination.

The definition of BMH1 can be stated in a slightly different way by strengthening the antecedent as shown in the following Lemma 4.2.1.

Lemma 4.2.1

BMH1

\[ \Leftrightarrow \forall s : State, ss : \mathbb{P} \ State_{\bot} \bullet (s, ss \cup \{\bot\}) \in B \land \bot \notin ss \Rightarrow (s, ss) \in B \]

Proof.

BMH1

\[ \Leftrightarrow \forall s : State, ss : \mathbb{P} \ State_{\bot} \bullet (s, ss \cup \{\bot\}) \in B \Rightarrow (s, ss) \in B \]

\{Definition of BMH1\}

\[ \Leftrightarrow \forall s : State, ss : \mathbb{P} \ State_{\bot} \bullet \left( (s, ss \cup \{\bot\}) \in B \land (\bot \in ss \lor \bot \notin ss) \Rightarrow (s, ss) \in B \right) \]

\{Predicate calculus\}

\[ \Leftrightarrow \forall s : State, ss : \mathbb{P} \ State_{\bot} \bullet \left( (s, ss \cup \{\bot\}) \in B \land \left( (s, ss \cup \{\bot\}) \in B \land (\bot \notin ss) \Rightarrow (s, ss) \in B \right) \right) \]

\{Predicate calculus\}

\[ \Leftrightarrow \forall s : State, ss : \mathbb{P} \ State_{\bot} \bullet \left( (s, ss \cup \{\bot\}) \in B \land \bot \notin ss \Rightarrow (s, ss) \in B \right) \]

\{Property of sets (Lemma B.3.5)\}
\[
\forall s : \text{State}, ss : \mathbb{P} \text{State}_\bot \bullet \left( \left( (s, ss) \in B \land \bot \in ss \right) \Rightarrow (s, ss) \in B \right)
\]

\[
\forall s : \text{State}, ss : \mathbb{P} \text{State}_\bot \bullet \left( \left( (s, ss \cup \{\bot\}) \in B \land \bot \notin ss \right) \Rightarrow (s, ss) \in B \right)
\]

This property could alternatively be restated by restricting the type of \( ss \) to \( \mathbb{P} \text{State} \). This concludes our discussion regarding BMH1.

### 4.2.3 BMH2

The third healthiness condition captures a redundancy in the model, namely that a set of final states defined by either the empty set or the set \( \{\bot\} \) characterises abortion.

**Definition 32 (BMH2)**

\[
\forall s : \text{State} \bullet (s, \emptyset) \in B \Leftrightarrow (s, \{\bot\}) \in B
\]

Therefore we require that for all initial states \( s \), it is related to the empty set of final states if, and only if, it is also related to the set of final states \( \{\bot\} \).

If we consider BMH1 in isolation, it covers the reverse implication of BMH2 because if \( (s, \{\bot\}) \) is in the relation, so is \( (s, \emptyset) \). However, the implication of BMH2 is stronger than BMH1 by requiring \( (s, \{\bot\}) \) to be in the relation if \( (s, \emptyset) \) is in the relation.

The reason for letting this redundancy persist in the model is to keep it as similar as possible to the original model of binary multirelations. This is of particular interest as it helps with linking these models.

### 4.2.4 BMH3

The fourth healthiness condition characterises a subset of the model, of type \( BM_\bot \), that corresponds to the original theory of binary multirelations.

**Definition 33 (BMH3)**

\[
\forall s : \text{State} \bullet \left( ((s, \emptyset) \notin B) \Rightarrow (\forall ss : \mathbb{P} \text{State}_\bot \bullet (s, ss) \in B \Rightarrow \bot \notin ss) \right)
\]
If an initial state $s$ is not related to the empty set, then it must also be the case that for all sets of final states $ss$ related to $s$, $\bot$ is not included in the set of final states $ss$.

This healthiness condition excludes relations that do not guarantee termination for particular initial states, yet establish some set of final states. Example 7 is an instance of such a relation. This is also the case for the original theory of binary multirelations. If it is possible for a program not to terminate when started from some initial state, then execution from that state must lead to arbitrary behaviour. This is the same intuition behind $H3$ in the theory of designs.

It is precisely the restriction imposed by $BMH3$ that we avoid with the binary multirelational model proposed. However, in order to study its relationship with the existing models the subset of $BMH3$-healthy relations is of interest.

### 4.3 Healthiness conditions as fixed points

In this section we specify functions whose fixed points characterise the new model of binary multirelations. We also specify functions that characterise the subset corresponding to the original model of [15]. This characterisation allows, for example, to prove that the healthiness conditions are idempotent.

In Sections [4.3.1] to [4.3.4], each healthiness condition is characterised by a corresponding function. The systematic exploration of the properties of the functional composition of each function is deferred to Appendices [B.1] to [B.2]. Finally, in Sections [4.3.5] and [4.3.6] the two functions that characterise the model as a whole, and its subset of interest, are presented. Furthermore, we prove that the fixed points correspond exactly to the relations satisfied by the predicative healthiness conditions defined earlier.

In general, for each healthiness condition of interest, we use the notation $bmh_x$ to denote the function whose fixed points correspond exactly to the relations characterised by the healthiness condition $BMHx$.

$$bmh_x(B) = B \iff BMHx$$

Furthermore, the notation $bmh_{x,y}$ denotes the functional composition of the respective functions:

$$bmh_{x,y}(B) = bmh_x \circ bmh_y(B)$$

This concludes the discussion of the notation used in the following sections.
4.3.1 bmh₀

The first function of interest is bmh₀ whose fixed points are the BMH₀-healthy binary multirelations.

**Definition 34 (bmh₀)**

\[
\text{bmh₀}(B) = \{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
| \exists ss_0 \cdot (s, ss_0) \in B \land ss_0 \subseteq ss \land (\bot \in ss_0 \iff \bot \in ss) \}
\]

This definition is justified by the following Lemma 4.3.1.

**Lemma 4.3.1 (BMH₀-iff-bmh₀)**

\[\text{BMH₀} \iff \text{bmh₀}(B) = B\]

**Proof.**

\[
\text{BMH₀} \quad \{\text{Definition of BMH₀}\}
\]

\[
\iff \left( \forall s_0 : \text{State}, ss_0, ss_1 : \mathbb{P} \text{State}_\bot \cdot \\
\left( ((s_0, ss_0) \in B \land ss_0 \subseteq ss_1 \land (\bot \in ss_0 \iff \bot \in ss_1)) \Rightarrow (s_0, ss_1) \in B \right) \right)
\quad \{\text{Predicate calculus: quantifier scope}\}
\]

\[
\iff \left( \forall s_0 : \text{State}, ss_1 : \mathbb{P} \text{State}_\bot \cdot \\
\left( \exists ss_0 : \mathbb{P} \text{State}_\bot \cdot (s_0, ss_0) \in B \land ss_0 \subseteq ss_1 \right) \Rightarrow (s_0, ss_1) \in B \right)
\quad \{\text{Property of sets: subset inclusion}\}
\]

\[
\iff \left\{ s_0 : \text{State}, ss_1 : \mathbb{P} \text{State}_\bot \cdot \\
\left( \exists ss_0 : \mathbb{P} \text{State}_\bot \cdot (s_0, ss_0) \in B \land ss_0 \subseteq ss_1 \right) \subseteq B \\
\land (\bot \in ss_0 \iff \bot \in ss_1) \right\}
\quad \{\text{Property of sets}\}
\]

\[
\iff \left\{ s_0 : \text{State}, ss_1 : \mathbb{P} \text{State}_\bot \\
\left( \exists ss_0 : \mathbb{P} \text{State}_\bot \cdot (s_0, ss_0) \in B \land ss_0 \subseteq ss_1 \right) \cup B \right\} = B
\quad \{\text{Property of sets}\}
\]

\[
\iff \left\{ s_0 : \text{State}, ss_1 : \mathbb{P} \text{State}_\bot \\
\left( \exists ss_0 : \mathbb{P} \text{State}_\bot \cdot (s_0, ss_0) \in B \land ss_0 \subseteq ss_1 \right) \\
\lor (\bot \in ss_0 \iff \bot \in ss_1) \lor (s_0, ss_1) \in B \right\}
\quad \{\text{Instantiation of existential quantifier for ss₀ = ss₁}\}
\]

Revision: 704f887 (2014-02-04 11:14:10 +0000) 39
When healthiness conditions are expressed as fixed points of a function it is essential that they are idempotent [1]. This is established for each of the functions $bmh$ in Appendix B.1. In the case of $bmh_0$ this is established by the Lemma B.1.1.

4.3.2 $bmh_1$

In this section the function $bmh_1$ that characterises BMH1-healthy relations is presented.

**Definition 35 ($bmh_1$)**

$$bmh_1(B) = \{ s : State, ss : \mathbb{P} State \mid (s, ss \cup \{ \bot \}) \in B \lor (s, ss) \in B \}$$

The function returns all pairs $(s, ss)$ in $B$, such that if a set of final states includes $\bot$ then there is also a set of final states without $\bot$. Its relationship with BMH1 is justified by the following Lemma 4.3.2.

**Lemma 4.3.2 (BMH1-iff-bmh$_1$)**

$$BMH1 \iff bmh_1(B) = B$$

**Proof.**

**BMH1** \{Definition of BMH1\}

$$\iff \forall s : State; ss : \mathbb{P} State \bullet (s, ss \cup \{ \bot \}) \in B \Rightarrow (s, ss) \in B$$ \{Property of sets and definition of subset inclusion\}

$$\iff \{ s : State; ss : \mathbb{P} State \mid (s, ss \cup \{ \bot \}) \in B \} \subseteq B$$ \{Property of sets\}

$$\iff (\{ s : State; ss : \mathbb{P} State \mid (s, ss \cup \{ \bot \}) \in B \} \cup B) = B$$ \{Property of sets\}
\[ \Leftrightarrow \left( \{ s : \text{State} ; \ ss : \mathbb{P} \text{State}_\bot \mid (s, ss \cup \{ \bot \}) \in B \lor (s, ss) \in B \} = B \right) \quad \text{(Definition of } \text{bmh}_1) \]

\[ \Leftrightarrow \text{bmh}_1(B) = B \]

Lemma [B.1.2] establishes that \text{bmh}_1 is idempotent. This concludes our discussion regarding the definition of \text{bmh}_1.

### 4.3.3 \text{bmh}_2

The healthiness condition \text{BMH}2 is characterised by the function \text{bmh}_2.

**Definition 36**

\[
\text{bmh}_2(B) \equiv \left\{ s : \text{State}, \ ss : \mathbb{P} \text{State}_\bot \mid (s, ss) \in B \land ((s, \{ \bot \}) \in B \Leftrightarrow (s, \emptyset) \in B) \right\}
\]

The definition considers every pair \((s, ss)\) in \(B\) and requires that \((s, \{ \bot \})\) is in \(B\) if and only if \((s, \emptyset)\) is also in \(B\). If the equivalence is not satisfied then \text{bmh}_2 yields the empty set. This definition is justified by the following Lemma 4.3.3.

**Lemma 4.3.3 (BMH2-iff-bmh\textsubscript{2})**

\[ \text{BMH}2 \Leftrightarrow \text{bmh}_2(B) = B \]

**Proof.**

\[
\begin{align*}
\text{BMH}2 & \Leftrightarrow \forall s : \text{State} \bullet \ (s, \emptyset) \in B \Leftrightarrow (s, \{ \bot \}) \in B \quad \{\text{Definition of BMH}2\} \\
& \Leftrightarrow \forall s : \text{State} \bullet \ (s, \emptyset) \in B \Rightarrow (s, \{ \bot \}) \in B \quad \{\text{Predicate calculus}\} \\
& \Leftrightarrow \forall s : \text{State} \bullet \ ((s, \emptyset) \in B \Rightarrow (s, \{ \bot \}) \in B) \quad \{\text{Predicate calculus}\} \\
& \Leftrightarrow \forall s : \text{State} \bullet \ ((\exists ss_0 : \mathbb{P} \text{State}_\bot \bullet (s, \emptyset) \in B \land (s, ss_0) \in B) \Rightarrow (s, \{ \bot \}) \in B) \quad \{\text{Predicate calculus}\} \\
& \Leftrightarrow \forall s : \text{State} \bullet \ ((\exists ss_0 : \mathbb{P} \text{State}_\bot \bullet (s, \{ \bot \}) \in B \land (s, ss_0) \in B) \Rightarrow (s, \emptyset) \in B) \quad \{\text{Predicate calculus}\}
\end{align*}
\]
\[\forall s : \text{State}, ss_0 : \mathbb{P} \text{State}_\bot \bullet \left( (s, ss_0) \in B \Rightarrow ((s, \{\bot\}) \in B \lor (s, \emptyset) \notin B) \right) \]
\[\frac{}{(s, ss_0) \in B \Rightarrow ((s, \emptyset) \in B \lor (s, \{\bot\}) \notin B)}\]
\{Predicate calculus\}

\[\forall s : \text{State}, ss_0 : \mathbb{P} \text{State}_\bot \bullet (s, ss_0) \in B \Rightarrow \left( (s, \{\bot\}) \in B \lor (s, \emptyset) \notin B \right)\]
\{Predicate calculus\}

\[\forall s : \text{State}, ss_0 : \mathbb{P} \text{State}_\bot \bullet (s, ss_0) \in B \Rightarrow ((s, \{\bot\}) \in B \iff (s, \emptyset) \in B)\]
\{Property of sets\}

\[\exists B \subseteq \{s : \text{State}, ss : \mathbb{P} \text{State}_\bot \mid (s, \{\bot\}) \in B \iff (s, \emptyset) \in B\} \]
\{Property of sets\}

\[\exists B = (B \cap \{s : \text{State}, ss : \mathbb{P} \text{State}_\bot \mid (s, \{\bot\}) \in B \iff (s, \emptyset) \in B\})\]
\{Property of sets\}

\[\exists B = \{s : \text{State}, ss : \mathbb{P} \text{State}_\bot \mid (s, ss) \in B \land ((s, \{\bot\}) \in B \iff (s, \emptyset) \in B)\}\]
\{Definition of \(\text{bmh}_2\)\}

\[\exists B = \text{bmh}_2(B)\]

\[\square\]

Similarly, Lemma \[\text{B.1.3}\] establishes that \(\text{bmh}_2\) is an idempotent function.

### 4.3.4 \(\text{bmh}_3\)

This section introduces the definition of \(\text{bmh}_3\) whose fixed points are BMH3-healthy relations.

**Definition 37**

\[
\text{bmh}_3(B) \equiv \{s : \text{State}, ss : \mathbb{P} \text{State}_\bot \mid ((s, \emptyset) \in B \lor \bot \notin ss) \land (s, ss) \in B\}
\]

The definition considers every pair \((s, ss)\) in \(B\) and requires that either \(ss\) is a set of final states with guaranteed termination or \((s, \emptyset)\) is in \(B\), and thus the initial state \(s\) leads to arbitrary behaviour. This is justified by the following Law \[\text{[B.3.1]}\]
Law 4.3.1 (BMH3-bmh₃)

\[ \text{BMH3} \Leftrightarrow \text{bmh₃}(B) = B \]

Proof.

\[ \text{BMH3} \quad \Rightarrow \forall s : \text{State} \bullet ((s, \emptyset) \notin B) \Rightarrow (\forall ss : \wp \text{State}_\bot \bullet (s, ss) \in B \Rightarrow \bot \notin ss) \]

\{Definition of BMH3\}

\[ \Rightarrow \forall s : \text{State}, ss : \wp \text{State}_\bot \bullet ((s, \emptyset) \notin B) \Rightarrow ((s, ss) \in B \Rightarrow \bot \notin ss) \]

\{Predicate calculus\}

\[ \Rightarrow \forall s : \text{State}, ss : \wp \text{State}_\bot \bullet ((s, ss) \in B \land \bot \in ss) \Rightarrow (s, \emptyset) \in B \]

\{Predicate calculus\}

\[ \Rightarrow B \subseteq \{ s : \text{State}, ss : \wp \text{State}_\bot \mid ((s, \emptyset) \in B \lor \bot \notin ss) \} \]

\{Property of sets\}

\[ \Rightarrow B = (B \cap \{ s : \text{State}, ss : \wp \text{State}_\bot \mid ((s, \emptyset) \in B \lor \bot \notin ss) \}) \]

\{Property of sets\}

\[ \Rightarrow B = \{ s : \text{State}, ss : \wp \text{State}_\bot \mid ((s, \emptyset) \in B \lor \bot \notin ss) \land (s, ss) \in B \} \]

\{Definition of bmh₃\}

\[ \Rightarrow B = \text{bmh₃}(B) \]

Finally, Lemma [B.1.4] establishes that bmh₃ is an idempotent function.

This section concludes our discussion regarding the definition of the bmhₓ functions. Their functional composition is studied in detail in Appendix [B.1].

In the following sections we focus our attention only on the functional compositions that characterise the theory and its subset of interest.

4.3.5 BMH0-BMH2 as a fixed point (bmh₀,₁,₂)

The relations in the theory are characterised by the conjunction of the healthiness conditions BMH0-BMH2. These relations can also be characterised as fixed points of the function bmh₀,₁,₂ as defined below.
Definition 38

\[
\text{bmh}_{0,1,2}(B) \ni \begin{cases} 
  s : \text{State}, \ ss : \mathbb{P} \text{State} \\
  \exists ss_0 \bullet ((s, ss_0) \in B \lor (s, ss_0 \cup \{\bot\}) \in B) \\
  \land ((s, \{\bot\}) \in B \iff (s, \emptyset) \in B) \\
  \land ss_0 \subseteq ss \land (\bot \in ss_0 \iff \bot \in ss)
\end{cases}
\]

This definition is justified by the functional composition of the respective \textbf{bmh} functions as shown in the following Lemma 4.3.4.

Lemma 4.3.4

\[
\text{bmh}_0 \circ \text{bmh}_1 \circ \text{bmh}_2(B) = \begin{cases} 
  s : \text{State}, \ ss : \mathbb{P} \text{State} \\
  \exists ss_0 \bullet ((s, ss_0) \in \text{bmh}_2(B) \lor (s, ss_0 \cup \{\bot\}) \in \text{bmh}_2(B)) \\
  \land ss_0 \subseteq ss \land (\bot \in ss_0 \iff \bot \in ss)
\end{cases}
\]

Proof.

\[
\text{bmh}_0 \circ \text{bmh}_1 \circ \text{bmh}_2(B) \quad \{\text{Definition of \text{bmh}_0 \circ \text{bmh}_1}\}
\]

\[
= \begin{cases} 
  s : \text{State}, \ ss : \mathbb{P} \text{State} \\
  \exists ss_0 \bullet ((s, ss_0) \in \text{bmh}_2(B) \lor (s, ss_0 \cup \{\bot\}) \in \text{bmh}_2(B)) \\
  \land ss_0 \subseteq ss \land (\bot \in ss_0 \iff \bot \in ss)
\end{cases} \quad \{\text{Definition of \text{bmh}_2}\}
\]

\[
= \begin{cases} 
  s : \text{State}, \ ss : \mathbb{P} \text{State} \\
  \exists ss_0 : \text{State} \bullet \\
  (s, ss_0) \in \begin{cases} 
    s : \text{State}, \ ss : \mathbb{P} \text{State} \\
    (s, ss) \in B \land ((s, \{\bot\}) \in B \iff (s, \emptyset) \in B)
  \end{cases} \\
  \lor \\
  (s, ss_0 \cup \{\bot\}) \in \begin{cases} 
    s : \text{State}, \ ss : \mathbb{P} \text{State} \\
    (s, ss) \in B \land ((s, \{\bot\}) \in B \iff (s, \emptyset) \in B)
  \end{cases} \\
  \land ss_0 \subseteq ss \land (\bot \in ss_0 \iff \bot \in ss)
\end{cases} \quad \{\text{Property of sets}\}
\]
In the following Lemma 4.3.5 we prove that \( bmh_{0,1,2} \) is an idempotent function.

**Lemma 4.3.5 (bmh\(_{0,1,2}\)-idempotent)**

\[
\text{bmh}_{0,1,2} \circ \text{bmh}_{0,1,2}(B) = \text{bmh}_{0,1,2}(B)
\]

**Proof.**

\[
\text{bmh}_{0,1,2} \circ \text{bmh}_{0,1,2}(B)
= \begin{cases}
s : \text{State}, ss : \mathbb{P} \text{ State}_\bot & \\
\exists ss_0 \bullet ((s, ss_0) \in \text{bmh}_{0,1,2}(B) \lor (s, ss_0 \cup \{ \bot \}} \in \text{bmh}_{0,1,2}(B)) \\
\land ((s, \{ \bot \}) \in \text{bmh}_{0,1,2}(B) \leftrightarrow (s, \emptyset) \in \text{bmh}_{0,1,2}(B)) \\
\land ss_0 \subseteq ss \land (\bot \in ss_0 \leftrightarrow \bot \in ss) & \\
\end{cases}
\]

\[\text{Law B.2.6, Law B.2.5 and predicate calculus}\]

\[
\text{bmh}_{0,1,2} \circ \text{bmh}_{0,1,2}(B)
= \begin{cases}
s : \text{State}, ss : \mathbb{P} \text{ State}_\bot & \\
\exists ss_0 \bullet ((s, ss_0) \in \text{bmh}_{0,1,2}(B) \lor (s, ss_0 \cup \{ \bot \}) \in \text{bmh}_{0,1,2}(B)) \\
\land ((s, \{ \bot \}) \in \text{bmh}_{0,1,2}(B) \leftrightarrow (s, \emptyset) \in \text{bmh}_{0,1,2}(B)) \\
\land ss_0 \subseteq ss \land (\bot \in ss_0 \leftrightarrow \bot \in ss) & \\
\end{cases}
\]

\[\text{Predicate calculus}\]

\[
\text{bmh}_{0,1,2} \circ \text{bmh}_{0,1,2}(B)
= \begin{cases}
s : \text{State}, ss : \mathbb{P} \text{ State}_\bot & \\
\exists ss_0 \bullet (s, ss_0) \in \text{bmh}_{0,1,2}(B) \land ss_0 \subseteq ss \\
\land (\bot \in ss_0 \leftrightarrow \bot \in ss) \\
\lor \\
\exists ss_0 \bullet (s, ss_0 \cup \{ \bot \}) \in \text{bmh}_{0,1,2}(B) \land ss_0 \subseteq ss \\
\land (\bot \in ss_0 \leftrightarrow \bot \in ss) & \\
\end{cases}
\]

\[\text{Law B.2.4}\]

Revision: 704f887 (2014-02-04 11:14:10 +0000) 45
\[
\begin{align*}
\begin{cases}
  s : \text{State}, \quad ss : \mathbb{P} \text{State}_\bot \\
  ((s, \{\bot\}) \in B \iff (s, \emptyset) \in B) \\
  \wedge \\
  \exists ss_1 \cdot \left( ((s, ss_1) \in B \lor (s, ss_1 \cup \{\bot\}) \in B) \wedge ss_1 \subseteq ss \wedge (\bot \in ss_1 \iff \bot \in ss) \right) \\
  \lor \\
  ((s, \{\bot\}) \in B \iff (s, \emptyset) \in B) \\
  \wedge \\
  \exists ss_1 \cdot \left( ((s, ss_1) \in B \lor (s, ss_1 \cup \{\bot\}) \in B) \wedge ss_1 \subseteq ss \wedge (\bot \in ss_1 \iff \bot \in ss) \right)
\end{cases}
= \text{bmh}_{0,1,2}(B)
\end{align*}
\]

The particular order of the functional composition is justified by Theorem 4.3.1.

**Theorem 4.3.1**

\[
\text{BMH}0 \land \text{BMH}1 \land \text{BMH}2 \iff \text{bmh}_{0,1,2}(B) = B
\]

*Proof.* Follows from Lemmas 4.3.6 to 4.3.8 and Lemma 4.3.9. □

This theorem, together with the respective lemmas enumerated in the following paragraphs, establishes that \( \text{bmh}_{0,1,2} \) is a suitable function for characterising \( \text{BMH}0\)-\( \text{BMH}2 \)-healthy relations. Appendix B.1 provides some reasoning as to why other orders of application are not desirable. For example, not all functions are necessarily commutative.

**From \( \text{bmh}_{0,1,2} \) to \( \text{BMH}0\)-\( \text{BMH}2 \)**

In the following laws we prove that the fixed points of \( \text{bmh}_{0,1,2} \) satisfy each of the predicative healthiness conditions \( \text{BMH}0, \text{BMH}1 \) and \( \text{BMH}2 \).
Lemma 4.3.6
\((\text{bmh}_{0,1,2}(B) = B) \Rightarrow \text{BMH0}\)

Proof.

\[
\text{BMH0} \quad \{\text{Definition of BMH0}\}
\]

\[
= \left( \forall s_0 : \text{State}, ss_0, ss_1 : \mathbb{P} \text{State}_{\perp} \bullet \right.
\]

\[
( (s_0, ss_0) \in B \land ss_0 \subseteq ss_1 \land (\perp \in ss_0 \Leftrightarrow \perp \in ss_1) ) \Rightarrow (s_0, ss_1) \in B
\]

\{Predicate calculus: quantifier scope\}

\[
= \left( \forall s_0 : \text{State}, ss_1 : \mathbb{P} \text{State}_{\perp} \bullet \right.
\]

\[
( \exists ss_0 : \mathbb{P} \text{State}_{\perp} \bullet (s_0, ss_0) \in B \land ss_0 \subseteq ss_1 \land (\perp \in ss_0 \Leftrightarrow \perp \in ss_1) )
\]

\{Assumption: \(\text{bmh}_{0,1,2}(B) = B\)\}

\[
= \left( \forall s_0 : \text{State}, ss_1 : \mathbb{P} \text{State}_{\perp} \bullet \right.
\]

\[
\exists ss_0 : \mathbb{P} \text{State}_{\perp} \bullet (s_0, ss_0) \in \text{bmh}_{0,1,2}(B) \land ss_0 \subseteq ss_1
\]

\{Law \[B.2.4]\}\}

\[
= \left( \forall s_0 : \text{State}, ss_1 : \mathbb{P} \text{State}_{\perp} \bullet \right.
\]

\[
\exists ss_0 : \mathbb{P} \text{State}_{\perp} \bullet \left( (s_0, ss_0) \in \text{bmh}_{0,1,2}(B) \land ss_0 \subseteq ss_1 \land (\perp \in ss_0 \Leftrightarrow \perp \in ss_1) \right)
\]

\{Law \[B.2.3]\}\}

\[
= \left( \forall s_0 : \text{State}, ss_1 : \mathbb{P} \text{State}_{\perp} \bullet \right.
\]

\[
( s_0, ss_1 ) \in \text{bmh}_{0,1,2}(B) \Rightarrow (s_0, ss_1) \in \text{bmh}_{0,1,2}(B)
\]

\{Predicate calculus\}

\[
= \text{true}
\]

\]

Lemma 4.3.7
\((\text{bmh}_{0,1,2}(B) = B) \Rightarrow \text{BMH1}\)

Revision: 704f887 (2014-02-04 11:14:10 +0000) 47
Proof.

BMH1 \quad \{\text{Lemma 4.2.1}\}
= \forall s : \text{State}, ss : \powerset s \cdot ((s, ss \cup \{\bot\}) \in B \land \bot \notin ss) \Rightarrow (s, ss) \in B
\quad \{\text{Assumption: } \text{bmh}_{0,1,2}(B) = B\}

= \left(\forall s : \text{State}, ss : \powerset s \cdot ((s, ss \cup \{\bot\}) \in \text{bmh}_{0,1,2}(B) \land \bot \notin ss) \Rightarrow (s, ss) \in \text{bmh}_{0,1,2}(B)\right)
\quad \{\text{Law B.2.3}\}

\begin{split}
= \left(\forall s : \text{State}, ss : \powerset s \cdot \\
\quad \left(\left((s, \{\bot\}) \in B \Leftrightarrow (s, \emptyset) \in B\right) \land \bot \notin ss \right) \\
\quad \land \\
\quad \exists ss_0 : \powerset s \cdot \\
\quad \left((ss_0) \in B \land (ss_0 \cup \{\bot\}) \in B \land ss_0 \subseteq (ss \cup \{\bot\}) \land (\bot \in ss_0 \Leftrightarrow \bot \in (ss \cup \{\bot\}))\right) \\
\quad \Rightarrow \\
\quad (s, ss) \in \text{bmh}_{0,1,2}(B)\right) \\
\quad \{\text{Property of sets}\}
\end{split}

\begin{split}
= \left(\forall s : \text{State}, ss : \powerset s \cdot \\
\quad \left(\left((s, \{\bot\}) \in B \Leftrightarrow (s, \emptyset) \in B\right) \land \bot \notin ss \right) \\
\quad \land \\
\quad \exists ss_0 : \powerset s \cdot \\
\quad \left((ss_0) \in B \land (ss_0 \cup \{\bot\}) \in B \land ss_0 \subseteq (ss \cup \{\bot\}) \land \bot \in ss_0 \Rightarrow \bot \in (ss \cup \{\bot\})\right) \\
\quad \Rightarrow \\
\quad (s, ss) \in \text{bmh}_{0,1,2}(B)\right) \\
\quad \{\text{Predicate calculus and property of sets}\}
\end{split}

\begin{split}
= \left(\forall s : \text{State}, ss : \powerset s \cdot \\
\quad \left(\left((s, \{\bot\}) \in B \Leftrightarrow (s, \emptyset) \in B\right) \land \bot \notin ss \right) \\
\quad \land \\
\quad \exists ss_0 : \powerset s \cdot \\
\quad \left((ss_0) \in B \land (ss_0 \cup \{\bot\}) \in B \land (ss_0 \setminus \{\bot\}) \subseteq ss \land \bot \in ss_0 \right) \\
\quad \Rightarrow \\
\quad (s, ss) \in \text{bmh}_{0,1,2}(B)\right) \\
\quad \{\text{Introduce fresh variable}\}
\end{split}
\[
\forall s : \text{State}, \; ss : \mathcal{P} \text{State}_\bot \bullet \\
\left( ((s, \{ \bot \}) \in B \Leftrightarrow (s, \emptyset) \in B) \land \bot \notin ss \right) \\
\land \\
\exists ss_0, \; t : \mathcal{P} \text{State}_\bot \bullet \left( ((s, ss_0) \in B \lor (s, ss_0 \cup \{ \bot \}) \in B) \right) \\
\land t \subseteq ss \land \bot \in ss_0 \\
\land t = (ss_0 \setminus \{ \bot \}) \\
\Rightarrow \\
(s, ss) \in \text{bmh}_{0,1,2}(B)
\]

\{Lemma [B.3.2] \}

\[
\forall s : \text{State}, \; ss : \mathcal{P} \text{State}_\bot \bullet \\
\left( ((s, \{ \bot \}) \in B \Leftrightarrow (s, \emptyset) \in B) \land \bot \notin ss \right) \\
\land \\
\exists ss_0, \; t : \mathcal{P} \text{State}_\bot \bullet \left( ((s, ss_0) \in B \lor (s, ss_0 \cup \{ \bot \}) \in B) \right) \\
\land t \subseteq ss \land \bot \in ss_0 \\
\land t \cup \{ \bot \} = ss_0 \\
\Rightarrow \\
(s, ss) \in \text{bmh}_{0,1,2}(B)
\]

\{One-point rule\}

\[
\forall s : \text{State}, \; ss : \mathcal{P} \text{State}_\bot \bullet \\
\left( ((s, \{ \bot \}) \in B \Leftrightarrow (s, \emptyset) \in B) \land \bot \notin ss \right) \\
\land \\
\exists t : \mathcal{P} \text{State}_\bot \bullet \left( ((s, t \cup \{ \bot \}) \in B \lor (s, t \cup \{ \bot \} \cup \{ \bot \}) \in B) \right) \\
\land t \subseteq ss \land \bot \in ss_0 \\
\land t \cup \{ \bot \} = ss_0 \\
\Rightarrow \\
(s, ss) \in \text{bmh}_{0,1,2}(B)
\]

\{Property of sets and predicate calculus\}

\[
\forall s : \text{State}, \; ss : \mathcal{P} \text{State}_\bot \bullet \\
\left( ((s, \{ \bot \}) \in B \Leftrightarrow (s, \emptyset) \in B) \right) \\
\land \\
\exists t : \mathcal{P} \text{State}_\bot \bullet (s, t \cup \{ \bot \}) \in B \land t \subseteq ss \land \bot \notin t \land \bot \notin ss \\
\Rightarrow \\
(s, ss) \in \text{bmh}_{0,1,2}(B)
\]

\{Law [B.2.3] \}
\[
\forall s \colon \text{State}, \ ss : \mathbb{P} \text{State}_\bot \cdot \\
\left( (s, \{ \bot \}) \in B \iff (s, \emptyset) \in B \right) \\
\land \\
\exists t : \mathbb{P} \text{State}_\bot \cdot (s, t \cup \{ \bot \}) \in B \land t \subseteq ss \land \bot \not\in t \land \bot \not\in ss \\
\Rightarrow \\
\left( (s, \{ \bot \}) \in B \iff (s, \emptyset) \in B \right) \\
\land \\
\exists ss_0 : \text{State}_\bot \cdot \left( (s, ss_0) \in B \lor (s, ss_0 \cup \{ \bot \}) \in B \land ss_0 \subseteq ss_1 \land (\bot \in ss_0 \iff \bot \in ss_1) \right)
\]
Lemma 4.3.8

\((\text{bmh}_{0,1,2}(B) = B) \Rightarrow \text{BMH2}\)

Proof.

\[
\begin{align*}
\text{BMH2} & \quad \{\text{Definition of BMH2}\} \\
= \forall s : \text{State} \cdot (s, \emptyset) \in B \iff (s, \{\bot\}) \in B & \quad \{\text{Assumption: } \text{bmh}_{0,1,2}(B) = B\} \\
= \forall s : \text{State} \cdot (s, \emptyset) \in \text{bmh}_{0,1,2}(B) \iff (s, \{\bot\}) \in \text{bmh}_{0,1,2}(B) & \quad \{\text{Law B.2.5 and Law B.2.6}\} \\
= \forall s : \text{State} \cdot ((s, \emptyset) \in B \land (s, \{\bot\}) \in B) \iff ((s, \emptyset) \in B \land (s, \{\bot\}) \in B) & \quad \{\text{Predicate calculus}\} \\
= \text{true} \\
\end{align*}
\]

These laws confirm that a fixed point of \(\text{bmh}_{0,1,2}\) satisfies each of the predicative healthiness conditions \(\text{BMH0-\text{BMH2}}\). In the following laws we prove the reverse implication of Theorem 4.3.1.

From \(\text{BMH0-\text{BMH2}}\) to \(\text{bmh}_{0,1,2}\)

A binary multirelation that is \(\text{BMH0, BMH1 and BMH2-healthy}\) is a fixed point of \(\text{bmh}_{0,1,2}\).

Lemma 4.3.9 Provided \(B\) is \(\text{BMH0 – BMH2-healthy}\).

\[\text{bmh}_{0,1,2}(B) = B\]

Proof.

\[
\begin{align*}
\text{bmh}_{0,1,2}(B) = B & \quad \{\text{Definition of bmh}_{0,1,2}\} \\
\Leftrightarrow \left\{ \\
\begin{array}{l}
s : \text{State}, ss : \mathbb{P}\text{State}_{\bot} \\
\exists ss_0 \cdot ((s, ss_0) \in B \lor (s, s_{s_0} \cup \{\bot\}) \in B) \\
\land ((s, \{\bot\}) \in B \iff (s, \emptyset) \in B) \\
\land ss_0 \subseteq ss \land (\bot \in ss_0 \iff \bot \in ss)
\end{array}
\right\} = B & \quad \{\text{Assumption: } B\text{ is BMH2-healthy}\}
\end{align*}
\]
These proofs conclude our discussion of the healthiness conditions of the new theory of binary multirelations. These relations can be characterised either by the predicates BMH0-BMH2 or as fixed points of \( \text{bmh}_{0,1,2} \). In the following section we focus our attention on the subset of the theory that is in addition BMH3-healthy.

### 4.3.6 BMH0-BMH3 as a fixed point (bmh\(_{0,1,3,2}\))

The relations that are BMH0, BMH1, BMH2 and BMH3-healthy can be characterised as fixed points of the following function.
Definition 39

\[ \text{bmh}_{0,1,3,2}(B) = \begin{cases} \text{s : State, ss : } \mathbb{P} \text{ State}_\bot \\ ( (s, \emptyset) \in B \land (s, \{\bot\}) \in B) \\ \lor \\ (s, \{\bot\}) \notin B \land (s, \emptyset) \notin B \\ \land \\ \exists ss_0 \bullet ( (s, ss_0) \in B \land ss_0 \subseteq ss \land \bot \notin ss_0 \land \bot \notin ss ) \end{cases} \]

This definition is justified by the following Lemma

Lemma 4.3.10

\[ \text{bmh}_0 \circ \text{bmh}_1 \circ \text{bmh}_3 \circ \text{bmh}_2(B) = \begin{cases} \text{s : State, ss : } \mathbb{P} \text{ State}_\bot \\ ( (s, \emptyset) \in B \land (s, \{\bot\}) \in B) \\ \lor \\ (s, \{\bot\}) \notin B \land (s, \emptyset) \notin B \\ \land \\ \exists ss_0 \bullet ( (s, ss_0) \in B \land ss_0 \subseteq ss \land \bot \notin ss_0 \land \bot \notin ss ) \end{cases} \]

Proof.

\[ \text{bmh}_0 \circ \text{bmh}_1 \circ \text{bmh}_3 \circ \text{bmh}_2(B) \quad \{ \text{Law B.2.8} \} \]

\[ = \begin{cases} \text{s : State, ss : } \mathbb{P} \text{ State}_\bot \\ \exists ss_0 \bullet ( (s, ss_0) \in \text{bmh}_2(B) \lor (s, ss_0 \cup \{\bot\}) \in \text{bmh}_2(B)) \\ \land \\ (s, \emptyset) \in \text{bmh}_2(B) \land ss_0 \subseteq ss \land (\bot \in ss_0 \Leftrightarrow \bot \in ss) \\ \lor \\ \exists ss_0 \bullet ( (s, ss_0) \in \text{bmh}_2(B) \land ss_0 \subseteq ss \land \bot \notin ss_0 \land \bot \notin ss ) \end{cases} \quad \{ \text{Definition of bmh}_2 \} \]
Predicate calculus: absorption law

\[
\begin{align*}
  s & : \text{State, } ss : \mathbb{P} \text{ State}_1 \\
  \exists ss_0 \bullet & \left( ((s, ss_0) \in B \lor (s, ss_0 \cup \{\bot\}) \in B) \right) \\
  & \land (s, \emptyset) \in B \land (s, \{\bot\}) \in B \\
  & \land ss_0 \subseteq ss \land (\bot \in ss_0 \iff \bot \in ss) \\
  \lor \\
  \exists ss_0 \bullet & \left( (s, ss_0) \in B \land ((s, \{\bot\}) \in B \iff (s, \emptyset) \in B) \right) \\
  & \land ss_0 \subseteq ss \land \bot \notin ss_0 \land \bot \notin ss \\
\end{align*}
\]

\{Predicate calculus\}

\[
\begin{align*}
  s & : \text{State, } ss : \mathbb{P} \text{ State}_1 \\
  \land (s, \emptyset) \in B \land (s, \{\bot\}) \in B \\
  \exists ss_0 \bullet & \left( ((s, ss_0) \in B \lor (s, ss_0 \cup \{\bot\}) \in B) \right) \\
  & \land ss_0 \subseteq ss \land (\bot \in ss_0 \iff \bot \in ss) \\
  \lor \\
  \exists ss_0 \bullet & \left( (s, ss_0) \in B \land ((s, \{\bot\}) \in B \iff (s, \emptyset) \in B) \right) \\
  & \land ss_0 \subseteq ss \land \bot \notin ss_0 \land \bot \notin ss \\
\end{align*}
\]

\{Law [3.2.9]\}

\[
\begin{align*}
  s & : \text{State, } ss : \mathbb{P} \text{ State}_1 \\
  \land (s, \emptyset) \in B \land (s, \{\bot\}) \in B \\
  \lor (s, \{\bot\}) \in B \\
  \exists ss_0 \bullet & \left( ((s, ss_0) \in B \lor (s, ss_0 \cup \{\bot\}) \in B) \right) \\
  & \land ss_0 \subseteq ss \land (\bot \in ss_0 \iff \bot \in ss) \\
  \lor \\
  \exists ss_0 \bullet & \left( (s, ss_0) \in B \land ((s, \{\bot\}) \in B \iff (s, \emptyset) \in B) \right) \\
  & \land ss_0 \subseteq ss \land \bot \notin ss_0 \land \bot \notin ss \\
\end{align*}
\]

\{Predicate calculus: absorption law\}
In Lemma B.1.13 we prove that $\text{bmh}_{0,1,3,2}$ is idempotent. This also follows directly from idempotency of the respective functions $\text{bmh}_0-\text{bmh}_3$.

The following Theorem 4.3.2, together with the respective lemmas that we discuss in the following sections, establishes that the fixed points of $\text{bmh}_{0,1,3,2}$ correspond to the conjunction of the predicative healthiness conditions $\text{BMH}0-\text{BMH}3$.

**Theorem 4.3.2**

$$\text{BMH}0 \land \text{BMH}1 \land \text{BMH}2 \land \text{BMH}3 \Leftrightarrow \text{bmh}_{0,1,3,2}(B) = B$$

*Proof.* The implication follows from Lemma 4.3.11. While the reverse implication follows from the fact that $\text{bmh}_{0,1,3,2}$ is a fixed point of $\text{bmh}_{0,1,2}$.
(Lemma B.1.14) and Lemmas 4.3.6 to 4.3.8 and Law 4.3.2.

In the following sections we prove the auxiliary results pertaining to Theorem 4.3.2. First, we consider the lemmas needed to prove the implication. This is followed by lemmas supporting the proof of the reverse implication.

**From bmh$_{0,1,3,2}$ to BMH0-BMH3**

Since the model of BMH0-BMH3 is a subset of the more general model of BMH0-BMH2, every fixed point of bmh$_{0,1,3,2}$ is also a fixed point of bmh$_{0,1,2}$. This result is established in Lemma B.1.14. Together with those results established in Section 4.3.5, this allows us to ascertain that any fixed point of bmh$_{0,1,3,2}$ also satisfies BMH0-BMH2.

Finally, the following Law 4.3.2 establishes that every fixed point of bmh$_{0,1,3,2}$ satisfies the predicative healthiness condition BMH3.

**Law 4.3.2**

\[(\text{bmh}_{0,1,3,2}(B) = B) \Rightarrow \text{BMH3}\]

**Proof.**

\[
\begin{align*}
\text{BMH3} & \quad \{\text{Definition of BMH3}\} \\
= \forall s_0 : \text{State} \bullet \left( \left( (s_0, \emptyset) \notin B \right) \Rightarrow \left( \forall ss_0 : \mathbb{P} \text{State}_\bot \bullet (s_0, ss_0) \in B \Rightarrow \bot \notin ss_0 \right) \right) \\
& \quad \{\text{Predicate calculus}\} \\
= \forall s_0 : \text{State} \bullet \left( \exists ss_0 : \mathbb{P} \text{State}_\bot \bullet (s_0, ss_0) \in B \wedge \bot \in ss_0 \right) \\
& \quad \{\text{Assumption: bmh}_{0,1,3,2}(B) = B\} \\
= \forall s_0 : \text{State} \bullet \left( (s_0, \emptyset) \in \text{bmh}_{0,1,3,2}(B) \right) \\
& \quad \{\text{Law B.2.10 and Law B.2.13}\}
\end{align*}
\]
Having established the proof for the implication of Theorem 4.3.2 in the following section we focus on the reverse implication.

From BMH0-BMH3 to bmh\textsubscript{0,1,3,2}

Finally, the Lemma 4.3.11 establishes the proof with respect to the reverse implication of Theorem 4.3.2.

Lemma 4.3.11

BMH0 \land BMH1 \land BMH2 \land BMH3 \Rightarrow bmh\textsubscript{0,1,3,2}(B) = B

Proof.

bmh\textsubscript{0,1,3,2}(B) \quad \{\text{Definition of bmh}\textsubscript{0,1,3,2}\}
\[
\begin{align*}
\text{Predicate calculus} & = \left\{ \begin{array}{l}
s : \text{State}, \ ss : \mathbb{P} \text{State}_\bot \\
((s, \emptyset) \in B \land (s, \{\bot\}) \in B) \\
\lor \\
\quad (s, \{\bot\}) \notin B \land (s, \emptyset) \notin B \\
\land \\
\quad \exists ss_0 : \mathbb{P} \text{State}_\bot \bullet \left( (s, ss_0) \in B \land ss_0 \subseteq ss \land \bot \notin ss_0 \land \bot \notin ss \right)
\end{array} \right\} \\
\{\text{Predicate calculus}\}
\end{align*}
\]

\[
\begin{align*}
\text{Predicate calculus} & = \left\{ \begin{array}{l}
s : \text{State}, \ ss : \mathbb{P} \text{State}_\bot \\
((s, \emptyset) \in B \land (s, \{\bot\}) \in B) \lor ((s, \{\bot\}) \notin B \land (s, \emptyset) \notin B) \\
\lor \\
\quad (s, \emptyset) \in B \land (s, \{\bot\}) \in B \\
\land \\
\quad \exists ss_0 : \mathbb{P} \text{State}_\bot \bullet (s, ss_0) \in B \land ss_0 \subseteq ss \land \bot \notin ss_0 \land \bot \notin ss \\
\end{array} \right\} \\
\{\text{Assumption: } B \text{ is BMH2-healthy}\}
\end{align*}
\]

\[
\begin{align*}
\text{Predicate calculus} & = \left\{ \begin{array}{l}
s : \text{State}, \ ss : \mathbb{P} \text{State}_\bot \\
((s, \emptyset) \in B \leftrightarrow (s, \{\bot\}) \in B) \\
\lor \\
\quad (s, \emptyset) \in B \land (s, \{\bot\}) \in B \\
\land \\
\quad \exists ss_0 : \mathbb{P} \text{State}_\bot \bullet (s, ss_0) \in B \land ss_0 \subseteq ss \land \bot \notin ss_0 \land \bot \notin ss \\
\end{array} \right\} \\
\{\text{Assumption: } B \text{ is BMH0-healthy}\}
\end{align*}
\]

\[
\begin{align*}
\text{Predicate calculus: instatiation of existential quantifier for } ss_0 = ss \\
\left\{ \begin{array}{l}
s : \text{State}, \ ss : \mathbb{P} \text{State}_\bot \\
((s, \emptyset) \in B \land (s, \{\bot\}) \in B) \\
\lor \\
\quad (s, \emptyset) \in B \land (s, \{\bot\}) \in B \\
\land \\
\quad (s, ss) \in B \land \bot \notin ss \\
\end{array} \right\}
\end{align*}
\]
These results establish that there are suitable functions whose fixed points characterise the theories of interest. The more general theory, that can encode sets of final states where termination is not guaranteed is characterised by \( \text{bmh}_{0,1,2} \). The function \( \text{bmh}_{0,1,3,2} \) characterises the subset that corresponds to the original theory of binary multirelations. The relationship with the original theory of binary multirelations is explored in Section 4.6.

### 4.4 Refinement ordering

The refinement order for the new binary multirelation model is defined exactly as in the original theory of binary multirelations \[15\].

**Definition 40 (Refinement)**

\[
B_1 \sqsubseteq_{BM} B_0 \equiv B_1 \supseteq B_0
\]

It is defined as reverse subset inclusion, such that a program \( B_0 \) refines \( B_1 \) if and only if \( B_0 \) is a subset of \( B_1 \).
The extreme points of the theory follow from the subset ordering. As expected of a theory of designs, they are the everywhere miraculous program and abort. Their definition is presented below.

**Definition 41 (Miracle)**

\[ \top_{BM_\bot} \cong \emptyset \]

As in the original theory, miracle is denoted by the absence of any relationship between any input state and any set of final states, that is, the program cannot possibly be executed.

**Definition 42 (Abort)**

\[ \bot_{BM_\bot} \cong \text{State} \times \mathcal{P}\text{State}_\bot \]

On the other hand, abort is characterised by the universal relation similarly to the original theory [15], such that every initial state is related to every possible set of final states.

### 4.5 Operators

In this section the operators of the theory are defined. In Sections 4.5.1 to 4.5.3 the main operators are defined, namely, assignment, angelic choice and demonic choice. In Section 4.5.4 the definition of sequential composition in the new model is presented.

In Chapter 5 we establish that the operators defined here are in correspondence with those of the new theory of designs with angelic nondeterminism. There we prove that the operators are closed. Together with the respective isomorphism establish between the theories, these results are sufficient to establish closure of the operators under BMH0-BMH2. The proof of closure using only the assumptions of this model is left as future work.

#### 4.5.1 Assignment

In the new model there is in fact the possibility to define two distinct assignment operators. The first one behaves exactly as in the original theory of binary multirelations \( (x :=_{BM} e) \). It specifies the assignment of the value
of expression \( e \) to the program variable \( x \); is guaranteed to terminate. This operator does not need to be redefined, since \( BM \subseteq BM_\perp \).

The new operator that we define below, however, behaves rather differently, in that the sets of final states may or may not be terminating.

**Definition 43**

\[
(x := BM_\perp e) \triangleq \{ s : State, ss : \mathbb{P} State_\perp | s \oplus (x \mapsto e) \in ss \}
\]

This assignment guarantees that for every initial state \( s \), there is some set of final states available for angelic choice where \( x \) has the value of expression \( e \). However, termination is not guaranteed. While the angel can choose the final value of \( x \) it cannot possibly guarantee termination in this case.

### 4.5.2 Angelic choice

The definition of angelic choice is the same as in the original theory of binary multirelations.

**Definition 44**

\[
B_0 \sqcup_{BM_\perp} B_1 \triangleq B_0 \cap B_1
\]

It is defined by set intersection, such that for every set of final states available for demonic choice in \( B_0 \) and \( B_1 \) when started from a particular initial state, only those that can be chosen both in \( B_0 \) and \( B_1 \) are available.

In the following paragraphs we explore some of the properties observed by the angelic choice operator.

**Properties**

An interesting property of angelic choice that is observed in this model is illustrated by the following Law 4.5.1. It considers the angelic choice between two assignments of the same value, yet only one is guaranteed to terminate.

**Law 4.5.1**

\[
(x := BM_\perp e) \sqcup_{BM_\perp} (x := BM e) = (x := BM e)
\]
Proof.

\((x :=_{BM} e) \sqcup_{BM} (x :=_{BM} e)\) \hspace{1em} \{\text{Definition of} \ :=_{BM}, :=_{BM} \text{and} \ \sqcup_{BM}\}

\begin{align*}
&= \left( \{ s : \text{State}, ss : \mathbb{P} \text{State}_{\bot} \mid s \oplus (x \mapsto e) \in ss \} \cap \{ s : \text{State}, ss : \mathbb{P} \text{State}_{\bot} \mid s \oplus (x \mapsto e) \in ss \} \right) \hspace{1em} \{\text{Type: } \bot \notin \mathbb{P} \text{State}\} \\
&= \left( \{ s : \text{State}, ss : \mathbb{P} \text{State}_{\bot} \mid s \oplus (x \mapsto e) \in ss \} \cap \{ s : \text{State}, ss : \mathbb{P} \text{State}_{\bot} \mid s \oplus (x \mapsto e) \in ss \land \bot \notin ss \} \right) \hspace{1em} \{\text{Property of sets and predicate calculus}\}
\\
&= \{ s : \text{State}, ss : \mathbb{P} \text{State}_{\bot} \mid s \oplus (x \mapsto e) \in ss \land \bot \notin ss \} \hspace{1em} \{\text{Type: } \bot \notin \mathbb{P} \text{State}\}
\\
&= \{ s : \text{State}, ss : \mathbb{P} \text{State} \mid s \oplus (x \mapsto e) \in ss \} \hspace{1em} \{\text{Definition of} \ :=_{BM}\}
\\
&= (x :=_{BM} e)
\end{align*}

\[ \square \]

This result can be interpreted as follows: given an assignment which is guaranteed to terminate, adding an equivalent angelic choice which is potentially non-terminating does not in fact introduce any new choices. Termination can still be enforced.

In general, and as expected from the original model of binary multirelations, the angelic choice operator observes the following properties with respect to the extreme points.

**Law 4.5.2**

\[ \top_{BM} \sqcup_{BM} B = \top_{BM} \]

**Proof.**

\begin{align*}
\top_{BM} \sqcup_{BM} B & \hspace{1em} \{\text{Definition of} \ \top_{BM} \text{and} \ \sqcup_{BM}\} \\
= \emptyset \cap B & \hspace{1em} \{\text{Property of sets}\} \\
= \emptyset & \hspace{1em} \{\text{Definition of} \ \top_{BM}\}
\\
& \hspace{1em} \square
\end{align*}
The angelic choice between an everywhere miraculous program and any other program is still miraculous.

**Law 4.5.3**

\[ \perp_{BM \perp} \sqcup_{BM \perp} B = B \]

*Proof.*

\[ \perp_{BM \perp} \sqcup_{BM \perp} B \]
\[ = (State \times \mathbb{P}\text{State}_{\perp}) \cap B \quad \{\text{Definition of } \perp_{BM \perp} \text{ and } \sqcup_{BM \perp}\} \]
\[ = B \]

\[ \square \]

On the other hand, the angelic choice between abort and any other program \( B \) is the same as \( B \). That is, the angel will avoid choosing an aborting program if possible.

### 4.5.3 Demonic choice

The demonic choice operator is defined by set union, exactly as in the original theory of binary multirelations.

**Definition 45**

\[ B_0 \cap_{BM \perp} B_1 \triangleq B_0 \cup B_1 \]

For every initial state, a corresponding set of final states available for demonic choice in either, or both, of \( B_0 \) and \( B_1 \), is included in the result.

In the following paragraphs we present some results regarding the demonic choice operator.

**Properties**

Similar to the angelic choice operator, there is a general result regarding the demonic choice over the two assignment operators, terminating and not necessarily terminating. This is shown in the following Law 4.5.4.
Law 4.5.4

\[(x :=_{BM} e) \cap_{BM\bot} (x :=_{BM\bot} e) = (x :=_{BM\bot} e)\]

Proof.

\[(x :=_{BM} e) \cap_{BM\bot} (x :=_{BM\bot} e) \quad \{\text{Definition of } :=_{BM}, :=_{BM\bot} \text{ and } \cap_{BM\bot}\}\]

\[= \left( \bigcup \left\{ s : \text{State}, ss : \mathbb{P} \text{State} \mid s \oplus (x \mapsto e) \in ss \right\} \right) \cup \left\{ s : \text{State}, ss : \mathbb{P} \text{State}_{\bot} \mid s \oplus (x \mapsto e) \in ss \right\} \quad \{\text{Type: } \bot \notin \mathbb{P} \text{State}\}\]

\[= \left( \bigcup \left\{ s : \text{State}, ss : \mathbb{P} \text{State}_{\bot} \mid s \oplus (x \mapsto e) \in ss \right\} \right) \quad \{\text{Property of sets}\}\]

\[= \left\{ s : \text{State}, ss : \mathbb{P} \text{State} \mid (s \oplus (x \mapsto e) \in ss \wedge \bot \notin \mathbb{P} \text{State}) \lor s \oplus (x \mapsto e) \in ss \right\} \quad \{\text{Predicate calculus: absorption law}\}\]

\[= \left\{ s : \text{State}, ss : \mathbb{P} \text{State} \mid s \oplus (x \mapsto e) \in ss \right\} \quad \{\text{Definition of } :=_{BM\bot}\}\]

\[= (x :=_{BM\bot} e)\]

\[\square\]

This result can be interpreted as follows: if there is an assignment for which termination is not guaranteed, then the demonic choice over this assignment and an equivalent one that is guaranteed to terminate is the same as the assignment that does not require termination. In other words, if it is possible for the demon to choose between two similar sets of final states, one that is possibly non-terminating and one that terminates, then the one for which termination is not guaranteed dominates the choice.

The following two laws show how the demonic choice operator behaves with respect to the extreme points of the theory.

Law 4.5.5

\[\bot_{BM\bot} \cap_{BM\bot} B = \bot_{BM\bot}\]

Proof.

\[\bot_{BM\bot} \cap_{BM\bot} B \quad \{\text{Definition of } \bot_{BM\bot} \text{ and } \cap_{BM\bot}\}\]
\[
= (\text{State} \times \mathbb{P} \text{State}_\bot) \cup B \\
= (\text{State} \times \mathbb{P} \text{State}_\bot) \\
= \bot_{BM_\bot}
\]

\{Property of sets\}

\{Definition of \(\bot_{BM_\bot}\)\}

\[\square\]

**Law 4.5.6**

\[\top_{BM_\bot} \cap_{BM_\bot} B = B\]

**Proof.**

\[
\top_{BM_\bot} \cap_{BM_\bot} B \\
= \emptyset \cup B \\
= B
\]

\{Property of sets\}

\[\square\]

As expected, the demonic choice between abort and some other program is abort. In the case of a miracle, the demon will avoid choosing it if possible.

Since the angelic and demonic choice operators are defined as set intersection and union, respectively, they also distribute through each other. This is exactly the same property as in the original theory of binary multirelations.

### 4.5.4 Sequential composition

The definition of sequential composition is not immediately obvious. In fact, one of the main reasons for developing a new binary multirelational model is that it provides a more intuitive approach to the definition of sequential composition. Consider the following example from the theory of designs.

**Example 10**

\[
(x' = 1 \vdash \text{true}) ; D \Pi_D \\
= (x' = 1 \vdash \text{true}) ; D (\text{true} \vdash x') \\
= (\neg (x' \neq 1 ; \text{true}) \land \neg (\text{true} ; \text{false}) \vdash \text{true} ; x' = x) \\
\{\text{Definition of sequential composition for designs}\}
\]

\[
\{\text{Definition of sequential composition}\}
\]
\[ = (\neg (\exists x_0 \cdot x_0 \neq 1 \land true) \land \neg (\exists x_0 \cdot true \land false) \vdash \exists x_0 \cdot true \land x' = x_0) \]
\[ \quad \{ \text{Predicate calculus and one-point rule} \} \]
\[ = (\neg true \land \neg false \vdash true) \]
\[ = (false \vdash true) \quad \{ \text{Property of designs and predicate calculus} \} \]
\[ = true \quad \{ \text{Definition of } \bot_D \} \]
\[ = \bot_D \]

In this case, a non-H3-design is sequentially composed with \( \Pi_D \), the Skip of the theory. The result is an aborting program. In fact, this result can be generalised for the sequential composition of any non-H3-design.

The behaviour just described provides the motivation for the definition of sequential composition in the new binary multirelational model.

**Definition 46**

\[
B_0 ; \ _{BM\bot} B_1 \\
\cong \\
\begin{cases} 
  s_0 : \text{State}, ss_0 : \mathbb{P}\text{State}_{\bot} \\
  \exists ss : \mathbb{P}\text{State}_{\bot} \cdot (s_0, ss) \in B_0 \land \\
  \begin{cases} 
    \bot \in ss \\
    (\bot \notin ss \land ss \subseteq \{s_1 : \text{State} \mid (s_1, ss_0) \in B_1\}) 
  \end{cases}
\end{cases}
\]

This definition is similar to the one for binary multirelations, except for the case where \( B_0 \) may lead to sets of final states where termination is not guaranteed. For sets of final states where termination is guaranteed, that is, \( \bot \) is not in the set of intermediate states \( ss \), then the definition matches that of the original theory of binary multirelations. If \( \bot \) is in \( ss \), and hence termination is not guaranteed, then the result of the sequential composition is arbitrary as it can include any set of final states.

If we assume that \( B_0 \) is BMH0-healthy, then the definition of sequential composition can be split into the set union of two sets as shown in Law 4.5.7.

**Law 4.5.7** *Provided \( B_0 \) is BMH0-healthy.*

\[
B_0 ; \ _{BM\bot} B_1 \\
= 
\]
\[
\left( \{ s_0 : \text{State}, ss_0 : \mathbb{P} \text{State}_\perp \mid (s_0, \text{State}_\perp) \in B_0 \} \right.
\]
\[
\cup
\left( \{ s_0 : \text{State}, ss_0 : \mathbb{P} \text{State}_\perp \mid (s_0, \{ s_1 : \text{State} \mid (s_1, ss_0) \in B_1 \}) \in B_0 \} \right)
\]

**Proof.**

\[
B_0 \; ; \; _{BM \perp} B_1
\]
\[
= \left\{ \begin{array}{l}
  s_0 : \text{State}, ss_0 : \mathbb{P} \text{State}_\perp \\
  \exists ss : \mathbb{P} \text{State}_\perp \bullet (s_0, ss) \in B_0 \land \\
  \exists \perp \in ss \\
  \perp \notin ss \land ss \subseteq \{ s_1 : \text{State} \mid (s_1, ss_0) \in B_1 \}
\end{array} \right\}
\]
\[
\{ \text{Definition of } ; \; _{BM \perp} \}
\]

\[
= \left\{ \begin{array}{l}
  s_0 : \text{State}, ss_0 : \mathbb{P} \text{State}_\perp \\
  \exists ss : \mathbb{P} \text{State}_\perp \bullet (s_0, ss) \in B_0 \land \perp \in ss \\
\end{array} \right\}
\]
\[
\{ \text{Predicate calculus and property of sets} \}
\]

\[
= \left\{ \begin{array}{l}
  s_0 : \text{State}, ss_0 : \mathbb{P} \text{State}_\perp \\
  \exists ss : \mathbb{P} \text{State}_\perp \bullet (s_0, ss) \in B_0 \land \perp \in ss \land ss \subseteq \text{State}_\perp \\
\end{array} \right\}
\]
\[
\{ \text{Propositional calculus and property of sets} \}
\]

\[
= \left\{ \begin{array}{l}
  s_0 : \text{State}, ss_0 : \mathbb{P} \text{State}_\perp \\
  \exists ss : \mathbb{P} \text{State}_\perp \bullet (s_0, ss) \in B_0 \land \perp \in ss \land ss \subseteq \text{State}_\perp \\
  \perp \notin ss \land ss \subseteq \{ s_1 : \text{State} \mid (s_1, ss_0) \in B_1 \}
\end{array} \right\}
\]
\[
\{ \perp \text{ in State}_\perp \text{ and } \perp \text{ not in State} \}
\]

\[
= \left\{ \begin{array}{l}
  s_0 : \text{State}, ss_0 : \mathbb{P} \text{State}_\perp \\
  \exists ss : \mathbb{P} \text{State}_\perp \bullet (s_0, ss) \in B_0 \land \perp \in ss \land ss \subseteq \text{State}_\perp \land \perp \in \text{State}_\perp
\end{array} \right\}
\]

Revision: 704f887 (2014-02-04 11:14:10 +0000) 68
Assumption: $B_0$ is $\text{BMH}_0$-healthy and Law [B.2.1]

$$ = \left( \left\{ s_0 : \text{State}, ss_0 : \mathbb{P} \text{State}_\bot \mid (s_0, \text{State}_\bot) \in B_0 \right\} \cup \left\{ s_0 : \text{State}, ss_0 : \mathbb{P} \text{State}_\bot \mid (s_0, \{s_1 : \text{State} \mid (s_1, ss_0) \in B_1\}) \in B_0 \right\} \right) $$. \hfill \square

The first set considers the case when $B_0$ leads to sets of final states where termination is not required ($\text{State}_\bot$). The second set considers the case where termination is required.

For a similar example to Example 10 expressed in the new theory, we consider the following example, where a non-terminating assignment is followed by the assignment that requires termination, but does not change the value of $x$.

**Example 11**

$$(x :=_{BM_\bot} e) ;_{BM_\bot} (x :=_{BM} x) \quad \{\text{Definition of } ;_{BM_\bot} \text{ (Law 4.5.7)}\}$$

$$ = \left( \left\{ s_0 : \text{State}, ss_0 : \mathbb{P} \text{State}_\bot \mid (s_0, \text{State}_\bot) \in (x :=_{BM_\bot} e) \right\} \cup \left\{ s_0 : \text{State}, ss_0 : \mathbb{P} \text{State}_\bot \mid (s_0, \{s_1 : \text{State} \mid (s_1, ss_0) \in (x :=_{BM} x)\}) \in (x :=_{BM_\bot} e) \right\} \right) \quad \{\text{Definition of } :=_{BM} \text{ and } :=_{BM_\bot}\}$$

$$ = \left( \left\{ s_0 : \text{State}, ss_0 : \mathbb{P} \text{State}_\bot \mid (s_0, \text{State}_\bot) \in \{s : \text{State}, ss : \mathbb{P} \text{State}_\bot \mid s \oplus (x \mapsto e) \in ss\} \right\} \cup \left\{ s_0 : \text{State}, ss_0 : \mathbb{P} \text{State}_\bot \mid (s_0, \{s_1 : \text{State} \mid (s_1, ss_0) \in (x :=_{BM} x)\}) \in \{s : \text{State}, ss : \mathbb{P} \text{State} \mid s \oplus (x \mapsto e) \in ss\} \right\} \right) \quad \{\text{Property of sets}\}$$

$$ = \left( \left\{ s_0 : \text{State}, ss_0 : \mathbb{P} \text{State}_\bot \mid s_0 \oplus (x \mapsto e) \in \text{State}_\bot \right\} \cup \left\{ s_0 : \text{State}, ss_0 : \mathbb{P} \text{State}_\bot \mid s_0 \oplus (x \mapsto e) \in \{s_1 : \text{State} \mid (s_1, ss_0) \in (x :=_{BM} x)\} \right\} \right) \quad \{\text{Property of sets}\}$$

Revision: 704f887 (2014-02-04 11:14:10 +0000) 69
The result of this sequential composition is an aborting program. If it is possible for the first program not to terminate, then the sequential composition cannot provide any guarantees either. The properties observed by the sequential composition operator are explored in what follows.

Properties

The first property of interest considers the sequential composition of $\top_{BM\bot}$ followed by some program $B$. The result is also a miraculous program as shown in the following Law 4.5.8.

Law 4.5.8

$$\top_{BM\bot} ;_{BM\bot} B = \top_{BM\bot}$$

Proof.

$$\top_{BM\bot} ;_{BM\bot} B$$

$$= \emptyset ;_{BM\bot} B$$

$$= \emptyset$$

$$= \emptyset \cup \emptyset$$

$$= \emptyset$$

Hence, the result of the sequential composition is an aborting program.
The following law expresses that the sequential composition of abort with another program is also abort.

**Law 4.5.9**

\[ \bot_{BM_\perp} \triangleright_{BM_\perp} B = \bot_{BM_\perp} \]

*Proof.*

\[ \bot_{BM_\perp} \triangleright_{BM_\perp} B \]

\[ = (State \times \mathbb{P}State_{\perp}) \triangleright_{BM_\perp} B \]

\[ \quad \text{\{Definition of } \triangleright_{BM_\perp} \text{ as } \bot_{BM_\perp} \text{ is BMH0-healthy}\}} \]

\[ = \left\{ \begin{array}{l} s_0 : State, ss_0 : \mathbb{P}State_{\perp} | (s_0, State_{\perp}) \in (State \times \mathbb{P}State_{\perp}) \end{array} \right\} \]

\[ \cup \left\{ \begin{array}{l} s_0 : State, ss_0 : \mathbb{P}State_{\perp} \quad \text{\{Property of sets\}} \\
| \quad (s_0, \{s_1 : State | (s_1, ss_0) \in B\}) \in (State \times \mathbb{P}State_{\perp}) \end{array} \right\} \]

\[ = \left\{ s_0 : State, ss_0 : \mathbb{P}State_{\perp} | \text{true} \right\} \cup \left\{ s_0 : State, ss_0 : \mathbb{P}State_{\perp} | \text{true} \right\} \]

\[ = (State \times \mathbb{P}State_{\perp}) \quad \text{\{Property of sets\}} \]

\[ = \bot_{BM_\perp} \quad \text{\{Definition of } \bot_{BM_\perp} \text{\}} \]

\[ \square \]

In the following paragraphs we explore some examples with respect to the extreme points of the theory.

**Examples**

The following example describes the general behaviour of some program \( B \) that is BMH0-healthy sequentially composed with a miraculous program.

**Example 12**

\[ B \triangleright_{BM_\perp} \top_{BM_\perp} \]

\[ = \left( \{ s_0 : State, ss_0 : \mathbb{P}State_{\perp} | (s_0, State_{\perp}) \in B \} \right) \]

\[ \cup \left( \{ s_0 : State, ss_0 : \mathbb{P}State_{\perp} | (s_0, \{s_1 : State | (s_1, ss_0) \in \emptyset\}) \in B \} \right) \quad \text{\{Property of sets\}} \]
If \( B \) may not terminate for some set of final states, and it is BMH0-healthy, then the result of the sequential composition is also abort, as \( \text{State}_\perp \) is in \( B \).

If \( B \) aborts for some particular initial state \( s_0 \), then that state is related to the empty set in \( B \) and the result of the sequential composition is also abort. Otherwise, the result is miraculous as the union of both sets if the empty set.

The following example describes the behaviour of a program \( B \) sequentially composed with abort.

**Example 13**

\[
B ; \text{BM}_\perp \perp \text{BM}_\perp = \left( \{ s_0 : \text{State}, ss_0 : \mathbb{P} \text{State}_\perp \mid (s_0, \text{State}_\perp) \in B \} \cup \{ s_0 : \text{State}, ss_0 : \mathbb{P} \text{State}_\perp \mid (s_0, \emptyset) \in B \} \right)
\]

Because \( B \) is upward closed, if it definitely terminates then \( \text{State} \) is a superset of all sets of final states and is in \( B \). If \( B \) may or may not terminate for some particular set of final states, then \( \text{State}_\perp \) is also in \( B \) due to the upward closure guaranteed by BMH0. In either case, the sequential composition behaves as abort. If \( B \) is miraculous, then so is the sequential composition.

### 4.6 Relationship with binary multirelations

In this section we focus our attention on the relationship between the subset of the theory that is BMH3-healthy and the original theory of binary multirelations [15]. In the following Sections 4.6.1 and 4.6.2 we define the
linking functions that relate both models. Finally in Section 4.6.3 we prove that the linking functions form a bijection under the respective healthiness conditions of each theory.

4.6.1 \(bmb2bm\)

The function \(bmb2bm\) maps binary multirelations in the new model, of type \(BM_\bot\), to those in the original model. It is defined by considering every pair in \((s, ss)\) in \(B\) such that \(\bot\) is not in \(ss\).

**Definition 47 (bmb2bm)**

\[
bmb2bm : BM_\bot \rightarrow BM
\]

\[
bmb2bm(B) = \{ \{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \mid ((s, ss) \in B \land \bot \notin ss) \}
\]

In order to show that \(bmb2bm\) yields a binary multirelation that is BMH-healthy, we first calculate the result of applying \(bmb2bm\) to a relation that is BMH0-BMH3-healthy in Lemma 4.6.1. Finally in Theorem 4.6.1 we prove that \(bmb2bm\) yields a BMH-healthy binary multirelation.

**Lemma 4.6.1**

\[
bmb2bm(bmh_{0,1,3,2}(B)) = \left\{ \begin{array}{l}
\{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
\{ ((s, \emptyset) \in B \land (s, \{ \bot \}) \in B) \land \bot \notin ss \\
\lor \\
\left( (s, \{ \bot \}) \notin B \land (s, \emptyset) \notin B \\
\land \\
\exists ss_0 \ ( (s, ss_0) \in B \land ss_0 \subseteq ss \land \bot \notin ss_0 \land \bot \notin ss ) \} \right. \right. 
\right. 
\]

**Proof.**

\[
bmb2bm(bmh_{0,1,3,2}(B)) = \{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \mid ((s, ss) \in bmh_{0,1,3,2}(B) \land \bot \notin ss) \}
\]

{Definition of bmb2bm}

{"revision": 704f887 (2014-02-04 11:14:10 +0000) 73}
Theorem 4.6.1 (bmb2bm-is-bmhupclosed)

\[ \text{bmh}_{\text{upclosed}} \circ \text{bmb2bm}(\text{bmh}_{0,1,3,2}(B)) = \text{bmb2bm}(\text{bmh}_{0,1,3,2}(B)) \]

Proof.

\[ \text{bmh}_{\text{upclosed}} \circ \text{bmb2bm}(\text{bmh}_{0,1,3,2}(B)) = \begin{cases} 
\text{s : State, ss : P State} & \\
(s, ss) \notin (s, \emptyset) \in B \wedge (s, \{\bot\}) \in B \\
\wedge (s, \{\bot\}) \notin B \wedge (s, \emptyset) \notin B \\
\wedge \exists s_{s0} \bullet (s, ss_{0}) \in B \wedge s_{s0} \subseteq ss \wedge \bot \notin s_{s0} \wedge \bot \notin ss \end{cases} \]

\{Definition of bmh\_upclosed\}
\[
\begin{align*}
\exists s_{s_0} \cdot (s, ss_0) &\in \left\{ \\
\left( (s, \emptyset) \in B \land (s, \{\bot\}) \in B \right) \land \bot \notin ss_0 \cup \\
\left( (s, \{\bot\}) \notin B \land (s, \emptyset) \notin B \right) \land \\
\exists ss_0 \cdot (s, s_{s_0}) \in B \land ss_0 \subseteq ss \land \bot \notin ss_0 \land \bot \notin ss \\
\end{align*}
\]

\{Variable renaming and property of sets\}

\[
\begin{align*}
\exists ss_0 \cdot \\
\left( (s, \emptyset) \in B \land (s, \{\bot\}) \in B \right) \land \bot \notin ss_0 \\
\left( (s, \{\bot\}) \notin B \land (s, \emptyset) \notin B \right) \land \\
\exists ss_0 \cdot (s, ss_1) \in B \land ss_1 \subseteq ss \land \bot \notin ss_1 \land \bot \notin ss_0 \\
\end{align*}
\]

\{Predicate calculus: distributivity and quantifier scope\}

\[
\begin{align*}
\left( (s, \emptyset) \in B \land (s, \{\bot\}) \in B \right) \land \bot \notin ss \land \exists ss_0 \cdot \bot \notin ss_0 \land ss_0 \subseteq ss \\
\left( (s, \{\bot\}) \notin B \land (s, \emptyset) \notin B \right) \land \\
\exists ss_1, ss_0 \cdot (s, ss_1) \in B \land ss_1 \subseteq ss \land \bot \notin ss \land ss_0 \subseteq ss \\
\end{align*}
\]

\{Predicate calculus: case-analysis on ss_0\}

\[
\begin{align*}
\left( (s, \emptyset) \in B \land (s, \{\bot\}) \in B \right) \land \bot \notin ss \\
\left( (s, \{\bot\}) \notin B \land (s, \emptyset) \notin B \right) \land \\
\exists ss_1, ss_0 \cdot (s, ss_1) \in B \land ss_1 \subseteq ss \land \bot \notin ss \land ss_0 \subseteq ss \\
\end{align*}
\]

\{Predicate calculus\}
This result establishes that for BMH0-BMH3-healthy relations $bmb2bm(B)$ yields relations that are in the original theory.

4.6.2 $bm2bmb$

The function that maps from relations in the original model, of type $BM$, into the new model is $bm2bmb$ and its definition is presented below.

Definition 48 ($bm2bmb$)
\[
bm2bmb : BM \to BM_\perp
\]
\[
bm2bmb(B) \equiv \left\{ s : State, ss : \mathbb{P} State_\perp \mid ((s, \emptyset) \in B \land (s, \{\perp\}) \in B) \land \perp \notin ss \lor (s, \emptyset) \notin B \land \exists ss_0 \in B \land \perp \notin ss_0 \land ss_0 \subseteq ss \land \perp \notin ss \right\}
\]

It considers every pair $(s, ss)$ in $B$ where $\perp$ is not in the set of final states $ss$, or if $B$ is aborting for a particular initial state $s$, then the result is the universal relation of type $BM_\perp$.

In the following Lemma 4.6.2 we calculate the result of applying $bm2bmb$ to a relation that is BMH-healthy. Finally, Theorem 4.6.2 establishes that $bm2bmb$ yields relations that are BMH0-BMH3-healthy.

Lemma 4.6.2
\[
bm2bmb(bmh_{upclosed}(B)) = \left\{ s : State, ss : \mathbb{P} State_\perp \mid \exists ss_0 \in B \land \perp \notin ss_0 \land ss_0 \subseteq ss \land \perp \notin ss \lor (s, \emptyset) \in B \right\}
\]
Proof.

\(bm2bmb(bmh_{\text{upclosed}}(B))\) \quad \text{(Definition of \(bm2bmb\))}

\[
= \left\{ \begin{array}{l}
  s : \text{State}, ss : \mathbb{P} \text{State}_\perp \\
  ((s, ss) \in bmh_{\text{upclosed}}(B) \land \perp \notin ss) \lor (s, \emptyset) \in bmh_{\text{upclosed}}(B)
\end{array} \right\}
\]

\[
= \left\{ \begin{array}{l}
  s : \text{State}, ss : \mathbb{P} \text{State}_\perp \\
  \left( (s, ss) \in \left\{ \begin{array}{l}
    s : \text{State}, ss : \mathbb{P} \text{State}_\perp \\
    \exists ss_0 \bullet (s, ss_0) \in B \land \perp \notin ss_0 \land ss_0 \subseteq ss \land \perp \notin ss
  \end{array} \right\} \land \perp \notin ss \right)
\end{array} \right\}
\]

\[
= \left\{ \begin{array}{l}
  s : \text{State}, ss : \mathbb{P} \text{State}_\perp \\
  \exists ss_0 \bullet (s, ss_0) \in B \land \perp \notin ss_0 \land ss_0 \subseteq ss \land \perp \notin ss
\end{array} \right\}
\]

\[
= \left\{ \begin{array}{l}
  s : \text{State}, ss : \mathbb{P} \text{State}_\perp \\
  \exists ss_0 \bullet (s, ss_0) \in B \land \perp \notin ss_0 \land ss_0 \subseteq \emptyset
\end{array} \right\}
\]

\[
= \left\{ \begin{array}{l}
  s : \text{State}, ss : \mathbb{P} \text{State}_\perp \\
  \exists ss_0 \bullet (s, ss_0) \in B \land \perp \notin ss_0 \land ss_0 \subseteq ss \land \perp \notin ss
\end{array} \right\}
\]

\[
= \left\{ \begin{array}{l}
  (s, \emptyset) \in B
\end{array} \right\}
\]

\[\square\]

Theorem 4.6.2

\(bmh_{0,1,3,2} \circ bm2bmb(bmh_{\text{upclosed}}(B)) = bm2bmb(bmh_{\text{upclosed}}(B))\)

Proof.

\(bmh_{0,1,3,2} \circ bm2bmb(bmh_{\text{upclosed}}(B))\) \quad \text{(Definition of \(bmh_{0,1,3,2}\))}
\[
\begin{align*}
\{ s : \text{State}, ss : \mathbb{P} \text{State} \perp \\
(s, \emptyset) \in bm2bmb(bmh_{\text{upclosed}}(B)) \land (s, \perp) \in bm2bmb(bmh_{\text{upclosed}}(B)))
\end{align*}
\]

\[
\begin{align*}
\lor & \\
\begin{cases}
(s, \{ \perp \}) \notin bm2bmb(bmh_{\text{upclosed}}(B)) \land (s, \emptyset) \notin bm2bmb(bmh_{\text{upclosed}}(B)) \land \\
\exists ss_0 \cdot (s, ss_0) \in bm2bmb(bmh_{\text{upclosed}}(B)) \land ss_0 \subseteq ss \land \perp \notin ss_0 \land \perp \notin ss
\end{cases}
\end{align*}
\]

{Law B.2.19 and Law B.2.18}

\[
\begin{align*}
\{ s : \text{State}, ss : \mathbb{P} \text{State} \perp \\
(s, \emptyset) \in B \land (s, \emptyset) \in B
\end{align*}
\]

\[
\begin{align*}
\lor & \\
\begin{cases}
((s, \emptyset) \notin B \land (s, \emptyset) \notin B) \land \\
\exists ss_0 \cdot (s, ss_0) \in bm2bmb(bmh_{\text{upclosed}}(B)) \land ss_0 \subseteq ss \land \perp \notin ss_0 \land \perp \notin ss
\end{cases}
\end{align*}
\]

{Predicate calculus and definition of $bm2bmb(bmh_{\text{upclosed}}(B))$ (Law B.2.16)}

\[
\begin{align*}
\{ s : \text{State}, ss : \mathbb{P} \text{State} \perp \\
(s, \emptyset) \in B
\end{align*}
\]

\[
\begin{align*}
\lor & \\
\begin{cases}
\exists ss_0 \cdot (s, ss_0) \in B \land \perp \notin ss_0 \land ss_0 \subseteq ss \land \perp \notin ss
\lor \\
(s, \emptyset) \in B
\end{cases}
\end{align*}
\]

{Variable renaming and property of sets}

\[
\begin{align*}
\{ s : \text{State}, ss : \mathbb{P} \text{State} \perp \\
(s, \emptyset) \in B
\end{align*}
\]

\[
\begin{align*}
\lor & \\
\begin{cases}
\exists ss_0 \cdot (s, ss_0) \in B \land \perp \notin ss_0 \land ss_0 \subseteq ss \land \perp \notin ss
\lor \\
(s, \emptyset) \in B
\end{cases}
\end{align*}
\]

{Predicate calculus}
These results complete the proofs for healthiness regarding both linking functions. In the following section we discuss the isomorphism.

4.6.3 \textbf{bm2bmb and bmb2bm}

Using the results from the previous section we establish that \( \text{bm2bmb} \) and \( \text{bmb2bm} \) form a bijection for healthy relations. Theorem 4.6.3 establishes this for relations that are BMH0-BMH3-healthy, while Theorem 4.6.4 establishes the bijection for relations that are BMH-healthy.

\textbf{Theorem 4.6.3} \textit{Provided }\( B \) \textit{is BMH0-BMH3-healthy.}

\[ \text{bm2bmb} \circ \text{bmb2bm}(B) = B \]
Proof.

\[ \text{Assumption: } B \text{ is BMH0-BMH3-healthy} \]

\[ = \text{Definition of } bm2bmb \]

\[ = \begin{cases} 
  s : \text{State, } ss : \mathbb{P} \text{State} & \begin{aligned} 
  (s, ss) &\in \text{bm2bmb}(\text{bmh}_{0,1,3,2}(B)) \\
  \downarrow &\in ss \\
  (s, \emptyset) &\in \text{bm2bmb}(\text{bmh}_{0,1,3,2}(B)) 
  \end{aligned} 
  \end{cases} \quad \text{(Law [B.2.17])} \]

\[ = \begin{cases} 
  s : \text{State, } ss : \mathbb{P} \text{State} & \begin{aligned} 
  (s, ss) &\in \begin{cases} 
    s : \text{State, } ss : \mathbb{P} \text{State} & \\
    \begin{aligned} 
      (s, ss) &\in \text{bm2bmb}(\text{bmh}_{0,1,3,2}(B)) \\
      \downarrow &\in ss \\
      (s, \emptyset) &\in \text{bm2bmb}(\text{bmh}_{0,1,3,2}(B)) 
    \end{aligned} 
    \end{cases} \\
  \downarrow &\in ss \\
  (s, \emptyset) &\in \begin{cases} 
    s : \text{State, } ss : \mathbb{P} \text{State} & \\
    \begin{aligned} 
      (s, ss) &\in \text{bm2bmb}(\text{bmh}_{0,1,3,2}(B)) \\
      \downarrow &\in ss \\
      (s, \emptyset) &\in \text{bm2bmb}(\text{bmh}_{0,1,3,2}(B)) 
    \end{aligned} 
    \end{cases} 
  \end{aligned} 
  \end{cases} \quad \text{(Property of sets)} \]
\[
s : \text{State}, \ s : \mathbb{P} \ \text{State}_\bot
\]
\[
\begin{align*}
(s, \emptyset) \in B \land (s, \{\bot\}) \in B \land \bot \notin ss \\
\lor \\
(s, \{\bot\}) \notin B \land (s, \emptyset) \notin B \\
\land \\
\exists s_{s_0} \bullet ( (s, s_{s_0}) \in B \land s_{s_0} \subseteq ss \land \bot \notin s_{s_0} \land \bot \notin ss )
\end{align*}
\]
\[
\begin{align*}
(\emptyset, \emptyset) \in B \land (s, \emptyset) \notin \emptyset \\
\lor \\
(s, \emptyset) \notin B \land (s, \emptyset) \notin B \\
\land \\
\exists s_{s_0} \bullet ( (s, s_{s_0}) \in B \land s_{s_0} \subseteq \emptyset \land \bot \notin s_{s_0} \land \bot \notin \emptyset )
\end{align*}
\]
\{Property of sets, predicate calculus and one-point rule\}

\[
s : \text{State}, \ s : \mathbb{P} \ \text{State}_\bot
\]
\[
\begin{align*}
((s, \emptyset) \in B \land (s, \{\bot\}) \in B) \land \bot \notin \emptyset \\
\lor \\
(s, \{\bot\}) \notin B \land (s, \emptyset) \notin B \\
\land \\
\exists s_{s_0} \bullet ( (s, s_{s_0}) \in B \land s_{s_0} \subseteq s_{s_0} \land \bot \notin s_{s_0} \land \bot \notin ss )
\end{align*}
\]
\{Predicate calculus\}

\[
s : \text{State}, \ s : \mathbb{P} \ \text{State}_\bot
\]
\[
\begin{align*}
((s, \emptyset) \in B \land (s, \emptyset) \in B) \\
\lor \\
(s, \emptyset) \notin B \land (s, \emptyset) \notin B \\
\land \\
\exists s_{s_0} \bullet ( (s, s_{s_0}) \in B \land s_{s_0} \subseteq ss \land \bot \notin s_{s_0} \land \bot \notin ss )
\end{align*}
\]
\{Predicate calculus: absorption law\}
\[ \begin{align*}
&\begin{cases}
  s : \text{State}, \ ss : \mathcal{P} \ \text{State}_\bot \\
  (s, \emptyset) \in B \land (s, \{\bot\}) \in B \\
  \lor
  (s, \{\bot\}) \notin B \land (s, \emptyset) \notin B \\
  \land
  \exists \ ss_0 \cdot (s, ss_0) \in B \land ss_0 \subseteq ss \land \bot \notin ss_0 \land \bot \notin ss
\end{cases} \\
&\{\text{Definition of } bm_{0,1,3,2}\}
\\
&= bm_{0,1,3,2}(B) \quad \{\text{Assumption: } B \text{ is } \text{BMH0-BMH3-healthy}\}
\\
&= B
\end{align*} \]

\textbf{Theorem 4.6.4} Provided $B$ is BMH-healthy.

\[ bm_{2}bm \circ bm_{2}bm(B) = B \]

\textbf{Proof.}

\[ \begin{align*}
&bm_{2}bm \circ bm_{2}bm(B) \\
&= bm_{2}bm \circ bm_{2}bm(bm_{\text{upclosed}}(B)) \quad \{\text{Definition of } bm_{2}bm\}
\\
&= \{s : \text{State}, \ ss : \mathcal{P} \ \text{State}_\bot \mid (s, ss) \in bm_{2}bm(bm_{\text{upclosed}}(B)) \land \bot \notin ss\} \\
&\{\text{Law B.2.16}\}
\\
&\begin{cases}
  s : \text{State}, \ ss : \mathcal{P} \ \text{State}_\bot \\
  (s, ss) \in \begin{cases}
    s : \text{State}, \ ss : \mathcal{P} \ \text{State}_\bot \\
    \exists ss_0 \cdot (s, ss_0) \in B \land \bot \notin ss_0 \land ss_0 \subseteq ss \land \bot \notin ss
  \end{cases} \\
  \lor
  (s, \emptyset) \in B \\
  \land
  \bot \notin ss
\end{cases}
\\
&\{\text{Property of sets}\}
\\
&\begin{cases}
  s \cdot \text{State}, \ ss \cdot \mathcal{P} \ \text{State}_\bot \\
  \exists ss_0 \cdot (s, ss_0) \in B \land \bot \notin ss_0 \land ss_0 \subseteq ss \land \bot \notin ss
\end{cases} \\
\}\{\text{Predicate calculus}\}
These results show that the subset of the theory that is \textbf{BMH3}-healthy is isomorphic to the original theory of binary multirelations \cite{15}. This confirms that while our model is more expressive, it is still possible to express every program that could be specified using the original model.

\subsection{4.7 Final considerations}

In this section we have introduced a new binary multirelational model that allows specifying sets of final states for which termination is not required. This model extends that of \cite{15} by using the symbol $\bot$ to denote the possibility for non-termination. The healthiness conditions have been introduced as predicates and subsequently characterised as fixed points of idempotent functions. These functions have been studied at length and their functional composition has been justified.

The operators of the theory have been introduced and their properties studied. The definition of sequential composition is the most unexpected. Its intuition comes from the theory of designs. The full justification for the definition of some of the operators and of the refinement order, is deferred until the study of the equivalent predicative model in the following Chapter 5.
Chapter 5

Designs with angelic nondeterminism

In this chapter we introduce a new [UTP] theory of designs that embodies the notion of angelic nondeterminism. The starting points for this predicative model are the theory of [14] and the binary multirelational model presented in Chapter 4. For this reason we begin this chapter by discussing the choice of alphabet in Section 5.1 and its relationship with that of [14].

In Section 5.2 the healthiness conditions of the theory are defined. These are specified by idempotent and monotonic functions whose fixed points are the designs of interest.

Since this theory is a predicative account of the model of Chapter 4 in Section 5.3 we establish that these models are isomorphic. This is achieved by defining a pair of linking functions and subsequently proving that they form a bijection. This result enables us, for example, to establish the correspondence between the healthiness conditions and operators of both models.

In Section 5.4 we justify that the theory of designs that we propose is a complete lattice. The definition of refinement adopted is the same as in the original theory of designs. Furthermore, we prove that this corresponds exactly to the refinement ordering of the binary multirelational model of Chapter 4 which is defined as subset ordering.

Section 5.5 discusses the main operators of the theory, including assignment and sequential composition. The entire Section 5.6 is dedicated to the main focus of this theory: angelic and demonic nondeterminism. Finally, in Section 5.7 we show that the subset of H3-designs of our theory is isomorphic to the UTP model of [14].
5.1 Alphabet

The result in [14] establishes that demonic and angelic nondeterminism cannot be both directly modelled in the relational setting of the UTP. To address that, Cavalcanti et al. [14] propose a non-homogeneous theory that can encode demonic and angelic choices. Our aim is to build on that model, which is isomorphic to the monotonic predicate transformers [14], and define a theory of designs (that includes the observational variables ok and ok′ and can describe both demonic and angelic nondeterminism). In order to put our choice of alphabet into perspective, we first explain the reasoning behind the alphabet used in [14].

The work of Cavalcanti et al. [14] considers an alphabet that includes the undashed program variables and, as the only dashed variable, ac′. This sole dashed variable represents the set of final states that can be chosen by the angel. A state is a record whose components represent program variables. For example, if we specify a program that uses the program variable x, then each state in ac′ must contain a component of name x′, whose value is one of the possible final values of x.

The non-homogeneous relations can be understood as establishing the relationship between an initial state and a set of possible final states corresponding to the choices available to the angel. For example, in the case of the program specified by \( x := 1 \sqcup x := 2 \), where \( \sqcup \) is the angelic choice operator, the set of outcomes ac′ includes at least two states whose component x′ is set to the possible final values of x, 1 and 2, respectively.

Perhaps, the most surprising observation we can make about the theory in [14] is the absence of variables such as ok and ok′, although it captures termination. In particular, the healthiness conditions of that theory correspond to H1, H2 and in fact H3 as well. However, for our purposes, it is essential to use the variables ok and ok′ as other theories of interest, namely the theory of reactive processes [7], make use of these. Furthermore, as mentioned before, it is absolutely vital that we can consider non-H3 designs.

The theory that we propose is, therefore, a theory of designs: we consider an alphabet that includes ok and ok′. In addition, we introduce two variables s and ac′ as shown below.

**Definition 49 (Alphabet)**

\[
\begin{align*}
s : & \text{State} \\
ac' : & \text{State}
\end{align*}
\]
\[ \text{ac}': \mathbb{P} \text{State} \]
\[ \text{ok}, \text{ok}' : \{\text{true, false}\} \]

We observe that as mentioned in Chapter 2, it is possible to define a non-homogeneous theory of designs with \text{ok} and \text{ok}'.

The variable \( s \) encapsulates the initial values of program variables as record components: each component corresponds to an undashed program variable. The set of final states \text{ac}' is similar to that of [14] with the notable difference that we do not dash the variables in the record components, instead we only consider these as undashed. This simplifies reasoning and proofs. We observe that we still make an explicit distinction between the initial state, which are encoded by \( s \), and the final states, which are encoded in the set defined by \text{ac}'. It is possible to relate the two sets through the following pair of functions.

**Definition 50 (acdash-to-ac)**

\[
\begin{align*}
\text{acdash2ac}(ss) &= \left\{ s_0 : S_{\text{in}a}, s_1 : S_{\text{out}a} \mid s_1 \in ss \land \left( \bigwedge x : \alpha P \bullet s_0.x = s_1.(x') \right) \bullet s_0 \right\} \\
\text{ac2acdesh}(zz) &= \left\{ z_0 : S_{\text{in}a}, z_1 : S_{\text{out}a} \mid z_0 \in ss \land \left( \bigwedge x : \alpha P \bullet z_0.x = z_1.(x') \right) \bullet z_1 \right\}
\end{align*}
\]

The function \text{acdash2ac} maps a set \( ss \) of angelic choices whose record components are dashed variables into a set whose record components are undashed. This is achieved by considering every state \( s_1 \) in \( ss \) and every state \( s_0 \), such that \( s_0 \) is a state on the undashed variables of \( P \) and whose components are exactly the same as those in \( s_1 \), except that those in \( s_1 \) are dashed. Each state is characterised by its alphabet, so in the case of the dashed state \( s_1 : S_{\text{out}a} \) this corresponds to a state whose record components are those in the output alphabet, \( \text{out}a \), for some program.

These two functions are important in the definition of a link between the theories as explained later in Section 5.7. In the following section we introduce the healthiness conditions.

### 5.2 Healthiness conditions

The theory we propose is a theory of designs. Therefore, predicates at the very are fixed points of \text{H1} and \text{H2}. Furthermore, since we seek to integrate
designs with a model similar to [14], a consistent notion of termination must be established. In that model, non-termination is possible if the set of angelic choices can be empty [14]. However, explicit non-termination cannot be required since the theory adopts H2 as a healthiness condition.

In addition to characterising termination appropriately, we also need to ensure that the set of final choices \( ac' \) is upward closed. The reason behind this is further explained in Section 5.2.2. These two concerns are addressed separately by the healthiness conditions A0 and A1, respectively. We introduce A0 in Section 5.2.1 and A1 in Section 5.2.2. Finally in Section 5.2.3 both functions are composed together and their combined properties explored.

5.2.1 A0

The first healthiness condition provides a consistent treatment of termination between the auxiliary variable \( ok' \) and the value of \( ac' \) in the theory. It is defined as follows.

**Definition 51 (A0)** If \( ok' \) holds, then \( ac' \) cannot be empty. Otherwise any value for \( ac' \) is allowed.

\[
A0(P) \equiv P \land (\lnot ok \lor \neg P^f) \Rightarrow (ok' \Rightarrow ac' \neq \emptyset)
\]

A0 states that when a design \( P \) terminates, that is \( ok' \) is true, then it must also be the case that \( ac' \) is not empty. In other words, there must be at least one final state in \( ac' \) available to the angel. If the precondition \( \neg P^f \) is not satisfied then the design aborts and there are no guarantees on the outcome. This embodies the notion of termination as found in [14] and related models, such as binary multirelations [15]. This particular definition ensures that H1 and H2 are preserved as shown in the following section.

**Properties**

In the following laws we show that A0 is closed with respect to designs, idempotent, and monotonic with respect to the refinement ordering.

**Law 5.2.1 (A0-design)** If \( P \) is a design so is \( A0(P) \).

\[
A0(P) = (\neg P^f \vdash P^t \land ac' \neq \emptyset)
\]
Proof.

\[ A_0(P) \]  \{Definition of design and \( A_0 \)\}
\[ = (\neg P_i \vdash P_t) \land ((ok \land \neg P_i) \Rightarrow (ok' \Rightarrow ac' \neq \emptyset)) \]  \{Definition of design and propositional calculus\}
\[ = (ok \land \neg P_i) \Rightarrow (P_t \land ok' \land (ok' \Rightarrow ac' \neq \emptyset)) \]  \{Propositional calculus\}
\[ = (ok \land \neg P_i) \Rightarrow (P_t \land ok' \land ac' \neq \emptyset) \]  \{Definition of design\}
\[ = (\neg P_i \vdash P_t \land ac' \neq \emptyset) \]

Law 5.2.1 establishes that a design in our theory can be stated in the usual manner, with a precondition and a postcondition, but the postcondition must guarantee that \( ac' \) is not equal to the empty set. In other words, once its precondition is satisfied, it establishes the postcondition and terminates.

**Law 5.2.2 (A0-idempotent)**

\[ A_0 \circ A_0(P) = A_0(P) \]

Proof.

\[ A_0 \circ A_0(P) \]  \{Law 5.2.1\}
\[ = A_0(\neg P_i \vdash P_t \land ac' \neq \emptyset) \]  \{Law 5.2.1\}
\[ = (\neg P_i \vdash P_t \land ac' \neq \emptyset \land ac' \neq \emptyset) \]  \{Propositional calculus\}
\[ = (\neg P_i \vdash P_t \land ac' \neq \emptyset) \]  \{Definition of \( A_0 \)\}
\[ = A_0(P) \]

**Law 5.2.3 (A0-monotonic)**

\[ (P \subseteq Q) \Rightarrow (A_0(P) \subseteq A_0(Q)) \]

Proof.

\[ A_0(Q) \]  \{Definition of \( A_0 \)\}
\[ = Q \land ((ok \land \neg Q_i) \Rightarrow (ok' \Rightarrow ac' \neq \emptyset)) \]  \{Assumption: \( [Q \Rightarrow P] \Leftrightarrow [\neg P \Rightarrow \neg Q] \)\}
\[ \Rightarrow P \land ((ok \land \neg P_i) \Rightarrow (ok' \Rightarrow ac' \neq \emptyset)) \]  \{Definition of \( A_0 \)\}
\[ = A_0(P) \]
Law 5.2.2 establishes that the function $A_0$ is idempotent, and Law 5.2.3 establishes that it is monotonic. These results confirm the suitability of $A_0$ as a healthiness condition. In the following section we explore the closure properties of $A_0$.

**Closure properties**

In the following laws we show that $A_0$ is closed with respect to disjunction and conjunction.

**Law 5.2.4 (A0-conjunction-closure)** Provided $P$ and $Q$ are $A_0$-healthy.

$$A_0(P \land Q) = P \land Q$$

*Proof.*

$$P \land Q \quad \{\text{Assumption: } P \text{ and } Q \text{ are } A_0\text{-healthy}\}$$

$$= A_0(P) \land A_0(Q) \quad \{\text{Definition of } A_0\}$$

$$= (P \land ((ok \land \neg P_f) \Rightarrow (ok' \Rightarrow ac' \neq \emptyset))) \land (Q \land ((ok \land \neg Q_f) \Rightarrow (ok' \Rightarrow ac' \neq \emptyset))) \quad \{\text{Propositional calculus}\}$$

$$= (P \land Q) \land (((ok \land \neg P_f) \lor (ok \land \neg Q_f)) \Rightarrow (ok' \Rightarrow ac' \neq \emptyset)) \quad \{\text{Propositional calculus}\}$$

$$= A_0(P \land Q) \quad \{\text{Definition of } A_0\}$$

**Law 5.2.5 (A0-disjunction-closure)** Provided $P$ and $Q$ are $A_0$-healthy.

$$A_0(P \lor Q) = P \lor Q$$

*Proof.*

$$P \lor Q \quad \{\text{Assumption: } P \text{ and } Q \text{ are } A_0\text{-healthy}\}$$

$$= A_0(P) \lor A_0(Q) \quad \{\text{Definition of } A_0\}$$
= (\neg P^f \vDash P^t \land ac' \neq \emptyset) \lor (\neg Q^f \vDash Q^t \land ac' \neq \emptyset) \quad \text{\{Disjunction of designs\}}

= (\neg P^f \land \neg Q^f \vDash (P^t \land ac' \neq \emptyset) \lor (Q^t \land ac' \neq \emptyset)) \quad \text{\{Propositional calculus\}}

= (\neg (P^f \lor Q^f) \vDash (P^t \lor Q^t) \land ac' \neq \emptyset) \quad \text{\{Property of substitution\}}

= (\neg (P \lor Q)^f \vDash (P \lor Q)^t \land ac' \neq \emptyset) \quad \text{\{Definition of A0\}}

= A0(P \lor Q)

\square

We observe that the proofs for both Law 5.2.4 and Law 5.2.5 also show that A0 distributes over conjunction and disjunction, irrespective of satisfying their provisos. This concludes our discussion of the basic properties of A0.

5.2.2 A1

In addition to requiring a consistent treatment of termination, our theory of designs requires that both pre and postcondition observe the upward closure of the set of final states, ac'. When this requirement is applied on simple predicates, this corresponds exactly to the healthiness condition PBMH of [14]. The definition is reproduced below.

Definition 52 (PBMH)

\[ \text{PBMH}(P) \equiv P ; ac \subseteq ac' \]

For every fixed point \( P \) of PBMH, the value of \( ac' \) must be upward closed. We observe that the function PBMH is idempotent. Other properties of interest are established in Appendix D.

The requirement upon our theory of designs regarding upward closure concerns both pre and postcondition. This is specified by the following healthiness condition A1.

Definition 53 (A1)

\[ A1(P_0 \vDash P_1) \equiv (\neg \text{PBMH}(\neg P_0) \vDash \text{PBMH}(P_1)) \]

The upward closure of \( ac' \) in the postcondition is enforced exactly as in [14]. However, the precondition is treated differently. In this case we ensure that
it is the negation of the precondition that is upward closed, since it is the
negation that actually establishes the value of \( ac' \) for designs that do not
require termination. This can be illustrated by the following Lemma 5.2.1.

Lemma 5.2.1

\[
A_1(P_0 \vdash P_1) = ok \Rightarrow (((P_1 \mid; ac \subseteq ac') \land ok') \lor (\neg P_0 \mid; ac \subseteq ac'))
\]

Proof.

\[
A_1(P_0 \vdash P_1) \quad \{\text{Definition of } A_1\}
\]

\[
= (\neg (\neg P_0 \mid; ac \subseteq ac') \vdash P_1 \mid; ac \subseteq ac') \quad \{\text{Definition of designs}\}
\]

\[
= (ok \land (\neg (\neg P_0 \mid; ac \subseteq ac'))) \Rightarrow ((P_1 \mid; ac \subseteq ac') \land ok') \\
\quad \{\text{Predicate calculus}\}
\]

\[
= ok \Rightarrow (((P_1 \mid; ac \subseteq ac') \land ok') \lor (\neg P_0 \mid; ac \subseteq ac'))
\]

When the program is started it can either terminate, in which case \( ok' \) is
ture and \( P_1 \) is established, or \( \neg P_0 \) is established and termination is then not
required. In either case we enforce the upward closure of \( ac' \).

This concludes our discussion of the definition of \( A_1 \). In the sequel we
show how it satisfies some basic properties.

Properties

In the following Laws 5.2.6 and 5.2.7 we establish that \( A_1 \) is an idempotent
and monotonic function.

Law 5.2.6 (A1-idempotent)

\[
A_1 \circ A_1(P_0 \vdash P_1)
\]

Proof.

\[
A_1 \circ A_1(P_0 \vdash P_1) \quad \{\text{Definition of } A_1\}
\]

\[
= A_1 \circ (\neg \text{PBMH}(\neg P_0) \vdash \text{PBMH}(P_1)) \quad \{\text{Definition of } A_1\}
\]

\[
= (\neg (\text{PBMH}(\neg \neg \text{PBMH}(\neg P_0))) \vdash \text{PBMH} \circ \text{PBMH}(P_1)) \\
\quad \{\text{Propositional calculus}\}
\]
\[
(\neg ( PBMH \circ PBMH (\neg P_0)) \vdash PBMH \circ PBMH (P_1)) \quad \{\text{Law D.1.1}\}
\]
\[
(\neg ( PBMH (\neg P_0)) \vdash PBMH (P_1)) \quad \{\text{Definition of A1}\}
\]
\[
A1(P_0 \vdash P_1)
\]

\textbf{Law 5.2.7 (A1-monotonic)}

\[(P \sqsubseteq Q) \Rightarrow A1(P) \sqsubseteq A1(Q)\]

\textit{Proof.}

\[
A1(Q)
\]
\[
= A1(\neg Q^f \vdash Q^t) \quad \{\text{Definition of design and propositional calculus}\}
\]
\[
= A1((\neg ok \vee Q^f) \vee (Q^t \land ok')) \quad \{\text{Assumption: } [Q \Rightarrow P] \text{ holds}\}
\]
\[
= A1((\neg ok \vee (Q^f \land (Q^f \Rightarrow P^f))) \vee (Q^t \land (Q^t \Rightarrow P^t) \land ok')) \quad \{\text{Predicate calculus and definition of design}\}
\]
\[
= A1((\neg Q^f \land P^f) \vdash Q^t \land P^t) \quad \{\text{Definition of A1}\}
\]
\[
= (\neg PBMH(Q^f \land P^f) \vdash PBMH(Q^t \land P^t)) \quad \{\text{Definition of PBMH}\}
\]
\[
= (\neg PBMH(Q^f \land P^f) \vdash PBMH(Q^t \land P^t)) \quad \{\text{Definition of sequential composition}\}
\]
\[
= (\neg \exists ac_0 \bullet Q^f[ac_0/ac'] \land P^f[ac_0/ac'] \land ac_0 \subseteq ac' \vdash (Q^t \land P^t) \land ac \subseteq ac') \quad \{\text{Predicate calculus}\}
\]
\[
= \left( \forall ac_0 \bullet \neg Q^f[ac_0/ac'] \lor \neg P^f[ac_0/ac'] \lor \neg (ac_0 \subseteq ac') \right) \vdash (Q^t \land P^t) \land ac \subseteq ac' \quad \{\text{Predicate calculus}\}
\]
\[
= \left( \forall ac_0 \bullet \left( \neg Q^f[ac_0/ac'] \lor \neg (ac_0 \subseteq ac') \lor \neg P^f[ac_0/ac'] \lor \neg (ac_0 \subseteq ac') \right) \right) \vdash (Q^t \land P^t) \land ac \subseteq ac' \quad \{\text{Weaken precondition}\}
\]
\[
\begin{align*}
\forall ac_0 \cdot (\neg Q^f[ac_0/ac'] \lor \neg (ac_0 \subseteq ac')) \\
\lor
\forall ac_0 \cdot (\neg P^f[ac_0/ac'] \lor \neg (ac_0 \subseteq ac')) \quad \{\text{Weaken precondition}\}
\end{align*}
\]
\[
\begin{align*}
\exists \forall ac_0 \cdot (Q^t \land P^t) \quad \{\text{Predicate calculus}\}
\end{align*}
\]
\[
\begin{align*}
\equiv (\forall ac_0 \cdot (\neg P^f[ac_0/ac'] \lor \neg (ac_0 \subseteq ac')) \lor (Q^t \land P^t) ; ac \subseteq ac') \\
\{\text{Definition of sequential composition}\}
\end{align*}
\]
\[
\begin{align*}
\equiv (\neg \exists ac_0 \cdot (P^f \land ac_0 \subseteq ac') \lor (Q^t \land P^t) ; ac \subseteq ac') \\
\{\text{Strength of postcondition}\}
\end{align*}
\]
\[
\begin{align*}
\equiv (\neg PBMH(P^f) \vdash PBMH(P^t)) \\
\{\text{Definition of A1}\}
\end{align*}
\]
\[
\begin{align*}
\equiv A1(-P^f \vdash P^t) \\
\{\text{Definition of designs}\}
\end{align*}
\]
\[
\equiv A1(P)
\]

These results establish the suitability of $A1$ as a healthiness condition. We tackle the commutativity of $A1$ and $A0$ in Section 5.2.3, where we define $A$. In the following section we show the closure properties satisfied by $A1$.

**Closure properties**

The function $A1$ is closed with respect to disjunction. In fact it also distributes through disjunction. This is expected as $PBMH$ is defined by the standard sequential composition operator that distributes over disjunction [1].

**Law 5.2.8 (A1-distribute-disjunction)**

\[A1(P \lor Q) = A1(P) \lor A1(Q)\]

**Proof.**

\[
\begin{align*}
A1(P) \lor A1(Q) \\
= A1(-P^f \vdash P^t) \lor A1(-Q^f \vdash Q^t) \\
= (\neg PBMH(P^f) \vdash PBMH(P^t)) \lor (\neg PBMH(Q^f) \vdash PBMH(Q^t)) \\
\{\text{Disjunction of designs}\}
\end{align*}
\]
= (¬ PBMH(P^f) ∧ ¬ PBMH(Q^f) ⊨ PBMH(P^t) ∨ PBMH(Q^t))
{Propositional calculus}

= (¬ (PBMH(P^f) ∨ PBMH(Q^f)) ⊨ PBMH(P^t) ∨ PBMH(Q^t))
{Disjunction closure of PBMH (Law D.3.1)}

= (¬ (PBMH(P^f) ⊨ PBMH(P^t) ∨ Q^t))
{Definition of A1}

= A1(¬ (P^f ∨ Q^f) ⊨ P^t ∨ Q^t)
{Propositional calculus}

= A1(¬ P^f ∧ ¬ Q^f ⊨ P^t ∨ Q^t)
{Disjunction of designs}

= A1(¬ P^f ⊨ P^t) ∨ (¬ Q^f ⊨ Q^t)
{Definition of design}

= A1(P ∨ Q)

\[\square\]

Law 5.2.9 (A-closure-disjunction) Provided P and Q are A1-healthy.

A1(P ∨ Q) = P ∨ Q

Proof.

A1(P ∨ Q) = P ∨ Q
{Law 5.2.8}

= A1(P) ∨ A1(Q)
{Assumption: P and Q are A1-healthy}

= P ∨ Q

\[\square\]

This concludes our discussion regarding the properties observed by A1. In the following section we discuss the functional composition of A0 and A1.

5.2.3 A

The proposed theory of designs is characterised by the two healthiness conditions A0 and A1. The order in which these functions are composed is important since they do not always necessarily commute. In order to see the reason behind this consider the following counter-example.

Counter-example 1

A0 ◦ A1(true ⊨ ac' = ∅)
{Definition of A1}
\[ \text{Definition of sequential composition} \]
\[ \text{One-point rule and predicate calculus} \]
\[ \text{Definition of } A_0 \]
\[ \text{Predicate calculus} \]
\[ \text{Definition of } A_1 \]

In this example we apply the healthiness conditions to an unhealthy design whose postcondition requires non-termination: \( ac' = \emptyset \). In the first case \( A_1 \) changes the postcondition into \( true \), followed by the application of \( A_0 \). While in the second case, \( A_0 \) is applied in the first place, making the postcondition \( false \), a predicate that satisfies \( \text{PBMH} \). The resulting predicate conforms to the definition of \text{Miracle}. Thus the functions do not always commute.

If instead we consider healthy predicates, then we can ensure that \( A_0 \) and \( A_1 \) commute. The following Law 5.2.10 establishes this result for predicates that are \( A_1 \) healthy. In fact the only requirement is for the postcondition, \( P^t \) to satisfy \( \text{PBMH} \).

\textbf{Law 5.2.10 (A0-A1-commutative)} \quad \text{Provided } P^t \text{ satisfies PBMH.}

\[ A_0 \circ A_1(P) = A_1 \circ A_0(P) \]

\textbf{Proof.}
\[ A_0 \circ A_1(P) \quad \text{Definition of design} \]
\[ = A_0 \circ A_1(\neg P^f \vdash P^t) \quad \text{Definition of } A_1 \]
\[ = A_0(\neg \text{PBMH}(P^f) \vdash \text{PBMH}(P^t)) \quad \text{Law 5.2.1} \]
\[ = (\neg \text{PBMH}(P^f) \vdash \text{PBMH}(P^t) \land ac' \neq \emptyset) \quad \text{ac' \neq \emptyset satisfies PBMH (Lemma D.4.5)} \]
\[
= (\neg \text{PBMH}(P^f) \vdash \text{PBMH}(P^t) \land \text{PBMH}(ac' \neq \emptyset)) \\
\quad \{\text{Closure of PBMH w.r.t. conjunction (Law D.3.2)}\}
\]
\[
= (\neg \text{PBMH}(P^f) \vdash \text{PBMH}(\text{PBMH}(P^t) \land \text{PBMH}(ac' \neq \emptyset))) \\
\quad \{ac' \neq \emptyset \text{ satisfies PBMH (Lemma D.4.5)}\}
\]
\[
= (\neg \text{PBMH}(P^f) \vdash \text{PBMH}(\text{PBMH}(P^t) \land ac' \neq \emptyset)) \\
\quad \{\text{Assumption: } P^t \text{ satisfies PBMH}\}
\]
\[
= (\neg \text{PBMH}(P^f) \vdash \text{PBMH}(P^t \land ac' \neq \emptyset)) \quad \{\text{Definition of } A_1\}
\]
\[
= A_1(\neg P^f \vdash P^t \land ac' \neq \emptyset) \quad \{\text{Definition of } A_0\}
\]
\[
= A_1 \circ A_0(\neg P^f \vdash P^t) \quad \{\text{Definition of design}\}
\]
\[
= A_1 \circ A_0(P)
\]

Following this discussion it is safe to introduce the definition of \(A\) as the functional composition of \(A_1\) followed by \(A_0\).

**Definition 54 (A)**

\[
A(P) \doteq A_0 \circ A_1(P)
\]

Law 5.2.10 establishes that once the postcondition of \(P\) satisfies PBMH then the functions commute. Therefore by functionally composing first \(A_1\) we guarantee that this is always the case. In the following section we explore some of the basic properties of \(A\) as expected of a healthiness condition.

**Properties**

In the following laws we prove that \(A\) is idempotent, monotonic and that it commutes with \(H_1 \circ H_2\). These results establish the suitability of \(A\) as a healthiness condition for a theory of designs.

**Law 5.2.11 (A-idempotent)**

\[
A \circ A(P) = A(P)
\]

*Proof.*

\[
A \circ A(P) \quad \{\text{Definition of } A \text{ twice}\}
\]
\[
= A_0 \circ A_1 \circ A_0 \circ A_1(P)
\]
\[\text{Law 5.2.10 and } A_1(P) \text{ ensures } P^t \text{ satisfies PBMH}\]
\[
= A_0 \circ A_0 \circ A_1 \circ A_1(P)
\]
\[\text{A0-idempotent (Law 5.2.2) and A1-idempotent (Law 5.2.6)}\]
\[
= A_0 \circ A_1(P)
\]
\[\text{Definition of } A\]
\[
= A(P)
\]

The proof of Law 5.2.11 relies on the fact that once \(A_1(P)\) is applied, then \(P^t\) is guaranteed to satisfy PBMH. In turn this means that \(A_0\) and \(A_1\) commute according to Law 5.2.10. Finally both idempotents allow us to establish that the result of applying \(A\) twice is indeed \(A\).

**Law 5.2.12 (A-monotonic)**

\[P \sqsubseteq Q \Rightarrow A(P) \sqsubseteq A(Q)\]

*Proof.* Follows from \(A_0\)-monotonic (Law 5.2.3) and \(A_1\)-monotonic (Law 5.2.7).

As expected, the function \(A\) is monotonic as established by Law 5.2.12. This follows from the monotonicity of both \(A_0\) and \(A_1\).

**Law 5.2.13 (A-H-commutative)**

\[H_1 \circ H_2 \circ A(P) = A \circ H_1 \circ H_2(P)\]

*Proof.*

\[
H_1 \circ H_2 \circ A
\]
\[\text{Definition of } A\]
\[
= H_1 \circ H_2(\neg \text{PBMH}(P^f) \vdash \text{PBMH}(P^t) \land ac' \neq \emptyset)
\]
\[\text{Property of designs}\]
\[
= (\neg \text{PBMH}(P^f) \vdash \text{PBMH}(P^t) \land ac' \neq \emptyset)
\]
\[\text{Definition of } A\]
\[
= A(\neg P^t \vdash P^t)
\]
\[\text{Definition of } H_1 \circ H_2\]
\[
= A \circ H_1 \circ H_2(P)
\]

\[\square\]
The healthiness condition of our theory is $H \circ A$. Since $H$ and $A$ commute, and $H$ and $A$ are idempotents, so is $H \circ A$ [1]. Furthermore, monotonicity also follows from monotonicity of $H$ and $A$.

This concludes our discussion of the healthiness conditions of the theory of designs with angelic nondeterminism. The designs of interest are characterised as fixed points of $A$, an idempotent and monotonic function.

5.3 Relationship with the binary multirelational model

In this section we prove that the predicative model of $A$-healthy designs is isomorphic to the binary multirelational model presented in Chapter 4. As mentioned previously, this allows us to establish the correspondence of the healthiness conditions and operators of both models.

In order to do so, we define a pair of linking functions: $bmb2d$, that maps from binary multirelations to predicates, and $d2bmb$ that maps in the opposite direction. The latter is defined in the following Section 5.3.1 while the former is defined in Section 5.3.2. Finally, in Section 5.3.3 the isomorphism is established by proving that both functions form a bijection.

5.3.1 From designs to binary multirelations ($d2bmb$)

The first linking function of interest is $d2bmb$. It maps from $A$-healthy designs into relations of type $BM_{\bot}$. It is defined as follows.

**Definition 55 (d2bmb)** Provided $P$ is a design.

$$d2bmb : A \rightarrow BM_{\bot}$$

$$d2bmb(P) \equiv \left\{ \begin{array}{c}
s : State, ss : \mathbb{P} State_{\bot} \\
(\neg P^f \Rightarrow P^t)[ss/ac'] \land \bot \notin ss \\
\lor \\
(P^f[ss \setminus \{\bot\}]/ac') \land \bot \in ss \end{array} \right\}$$

For a given design $P = (\neg P^f \vdash P^t)$, the set construction of $d2bmb(P)$ is split into two disjuncts.

In the first disjunction we consider the case where $P$ is guaranteed to terminate, with $ok$ and $ok'$ both being substituted for $true$. The resulting
set of final states \( ss \), for which termination is required \(( \bot \notin ss \)\) is obtained by substituting \( ss \) for \( ac' \) in \( P \).

The second disjunct considers the case where \( ok \) is also \( true \), but \( ok' \) is \( false \). This corresponds to the situation where \( P \) does not terminate. In this case, the set of final states is obtained by substituting \( ss \) \( \{ \bot \} \) for \( ac' \) and requiring \( \bot \) to be in the set of final states \( ss \).

As a consequence of \( P \) satisfying \( H2 \), we ensure that if there is some set of final states captured by the second disjunct with \( \bot \), then there is also an equivalent set of final states without \( \bot \) that is captured by the first disjunct.

In the following Theorem 5.3.1 we prove that the application of \( d2bmb \) to \( A \)-healthy designs yields relations that are \( BMH0\)-BMH2-healthy.

**Theorem 5.3.1**

\[
\text{bmh}_{0,1,2} \circ d2bmb(A(P)) = d2bmb(A(P))
\]

*Proof.*

\[
\text{bmh}_{0,1,2} \circ d2bmb(A(P)) = \left\{ \begin{array}{l}
\exists s_0 : \text{P State}_\bot \\
\quad \left( (s, s_0) \in d2bmb(A(P)) \lor (s, s_0 \cup \{ \bot \}) \in d2bmb(A(P)) \right) \\
\quad \land (s, \{ \bot \}) \in d2bmb(A(P)) \leftrightarrow (s, \emptyset) \in d2bmb(A(P)) \\
\quad \land s_0 \subseteq ss \land (\bot \in ss_0 \Leftrightarrow \bot \in ss) \\
\end{array} \right. 
\]

\{Lemma C.1.4\}

\[
\exists s_0 : \text{P State}_\bot \\
\left( (s, s_0) \in d2bmb(A(P)) \lor (s, s_0 \cup \{ \bot \}) \in d2bmb(A(P)) \right) \\
\land s_0 \subseteq ss \land (\bot \in s_0 \Leftrightarrow \bot \in ss) \\
\}

\{Predicate calculus\}

\[
\exists s_0 : \text{P State}_\bot \\
\left( (s, s_0) \in d2bmb(A(P)) \right) \\
\land s_0 \subseteq ss \land (\bot \in s_0 \Leftrightarrow \bot \in ss) \\
\lor \\
\left( (s, s_0 \cup \{ \bot \}) \in d2bmb(A(P)) \right) \\
\land s_0 \subseteq ss \land (\bot \in s_0 \Leftrightarrow \bot \in ss) \\
\}

\{Lemmas C.1.2 and C.1.3\}
This result, whose proof relies on a number of lemmas proved in Appendix C.1, establishes the suitability of $d_2bmb$ as a linking function.

In order to understand the result of applying $d_2bmb$ better, we consider the following Example 14. It specifies a program that either assigns the value 1 to the sole program variable $x$ and successfully terminates, or assigns the value 2 to $x$, in which case termination is not required.

**Example 14**

$$d_2bmb((x \mapsto 2) \notin ac' \vdash (x \mapsto 1) \in ac') \quad \{ \text{Definition of } d_2bmb \}$$

$$= \begin{cases} 
  s : \text{State}, ss : \mathbb{P} \text{State}_\perp \\
  (\exists ac_0 : \mathbb{P} \text{State} \cdot \\
   (Pf[ac_0/ac'] \lor Pf[ac_0/ac'] \land ss \neq \emptyset \lor \perp \notin ss)) \land ac_0 \subseteq ss \\
   (\exists ac_0 : \mathbb{P} \text{State} \cdot Pf[ac_0/ac'] \land ac_0 \subseteq ss)
  \end{cases}$$

\{Predicate calculus and Lemma C.1.1\}

$$= d_2bmb(A(P))$$

\[ \square \]

Revision: 704f887 (2014-02-04 11:14:10 +0000) 100
\[
\begin{align*}
\{ s : State, ss : \mathbb{P} State \} \\
\quad | ((x \mapsto 2) \in ss \land \bot \notin ss) \\
\quad \lor ((x \mapsto 1) \in ss \land \bot \notin ss) \\
\quad \lor ((x \mapsto 2) \in ss \land \bot \in ss)
\end{align*}
\]

\{ Predicate calculus\}

\[
\begin{align*}
\{ s : State, ss : \mathbb{P} State \} \\
\quad | (x \mapsto 2) \in ss \lor ((x \mapsto 1) \in ss \land \bot \notin ss)
\end{align*}
\]

\{ Definition of \( \sqcap_{BM} \bot \) and \( :={BM} \sqcup \) \}

\[
\begin{align*}
=x :=_{BM} 2) \sqcap_{BM} (x :=_{BM} 1)
\end{align*}
\]

As expected, the function \( d2bmb \) yields a program with the same behaviour specified using the binary multirelational model. It is the demonic choice over two assignments, one requires termination while the other does not.

5.3.2 From binary multirelations to designs (\( bmb2d \))

The second linking function of interest is \( bmb2d \) that maps binary multirelations to \( A \)-healthy predicates. Its definition is presented below.

**Definition 56**

\[
bmb2d : BM \rightarrow A
\]

\[
bmb2d(B) \equiv ((s, ac' \cup \{ \bot \}) \notin B \vdash (s, ac') \in B)
\]

It is defined as a design, such that for a particular initial state \( s \), the precondition requires \( (s, ac' \cup \{ \bot \}) \) not to be in \( B \), while the postcondition establishes that \( (s, ac') \) is in \( B \). This definition can be expanded into a more intuitive representation according to the following Lemma 5.3.1.

**Lemma 5.3.1**

\[
bmb2d(B) = ok \Rightarrow \left( ((s, ac') \in B \land \bot \notin ac' \land ok') \lor ((s, ac' \cup \{ \bot \}) \in B \right)
\]

**Proof.** Follows from the definition of design and type restriction on \( ac' \). □

The behaviour of \( bmb2d \) is split into two disjuncts. The first one considers the case where \( B \) requires termination, and hence \( \bot \) is not part of the set of
final states of the pair in $B$. While the second disjunct considers sets of final states that do not require termination, in which case $ok'$ can be either $true$ or $false$.

The following Theorem 5.3.2 establishes that $bmb2d(B)$ yields $A$-healthy designs provided that $B$ is $BMH0-BMH2$-healthy.

**Theorem 5.3.2** Provided $B$ satisfies $bmh_{0,1,2}$.

$$A \circ bmb2d(B) = bmb2d(B)$$

**Proof.**

$$A \circ bmb2d(B) \quad \{\text{Assumption: } B = bmh_{0,1,2}(B) \text{ and Lemma }[C.2.2]\}$$

$$= A \left( \ \neg ((s, ac' \cup \{\bot\}) \in B; ac \subseteq ac') \right) \quad \{\text{Lemma }[C.2.1]\}$$

$$= A \left( \ \neg ((s, ac') \in B; ac \subseteq ac') \wedge (s, \emptyset) \notin B \right) \quad \{\text{Definition of } PBMH\}$$

$$= A \left( \ \neg \ PBMH((s, ac' \cup \{\bot\}) \in B) \right) \quad \{\text{Definition of } A\}$$

$$= A \left( \ \neg \ PBMH((s, ac') \in B) \wedge ac' \neq \emptyset \wedge (s, \emptyset) \notin B \right) \quad \{(\text{PBMH-idempotent}) \text{ Law }[D.1.1]\}$$

$$= A \left( \ \neg \ PBMH((s, ac' \cup \{\bot\}) \in B) \right) \quad \{\text{Lemma }[D.4.5] \text{ and Lemma }[D.4.6]\}$$

Revision: 701f887 (2014-02-04 11:14:10 +0000) 102
\[
\begin{align*}
\neg \text{PBMH}((s, ac' \cup \{\bot\}) \in B) \\
\vdash \text{PBMH}((s, ac') \in B) \land \text{PBMH}(ac' \neq \emptyset) \\
\land \text{PBMH}((s, \emptyset) \notin B) \land ac' \neq \emptyset \\
\{\text{Lemma D.4.5 and Lemma D.4.6 and predicate calculus}\}
\end{align*}
\]

\[
\begin{align*}
\neg \text{PBMH}((s, ac' \cup \{\bot\}) \in B) \\
\vdash \text{PBMH}((s, ac') \in B) \land ac' \neq \emptyset \land (s, \emptyset) \notin B \\
\{\text{Definition of PBMH and Lemma C.2.1}\}
\end{align*}
\]

\[
\begin{align*}
\neg ((s, ac' \cup \{\bot\}) \in B \land ac' \subseteq ac') \\
\vdash ((s, ac') \in B \land ac' \subseteq ac') \land (s, \emptyset) \notin B \\
\{\text{Assumption: } B = \text{bmh}_{0,1,2}(B) \text{ and Lemma C.2.2}\}
\end{align*}
\]

\[
bmb2d(B)
\]

This result confirms that \(bmb2d\) is closed with respect to \(A\) when applied to relations that are \(\text{BMH0-BMH2}\)-healthy. This concludes our discussion of \(bmb2d\). In the following section we focus our attention on the isomorphism.

\textbf{5.3.3 Isomorphism:} \(d2bmb\) and \(bmb2d\)

In this section we show that \(d2bmb\) and \(bmb2d\) form a bijection. The following Theorem \textbf{5.3.3} establishes that \(d2bmb\) is the inverse function of \(bmb2d\) for relations that are \(\text{BMH0-BMH2}\)-healthy. While Theorem \textbf{5.3.4} establishes that \(bmb2d\) is the inverse function of \(d2bmb\) for designs that are \(A\)-healthy. Together these results establish that the models are isomorphic.

\textbf{Theorem 5.3.3} \textit{Provided} \(B\) \textit{is} \(\text{BMH0-BMH2}\)-\textit{healthy}.

\[
d2bmb \circ bmb2d(B) = B
\]

\textit{Proof.}

\[
d2bmb \circ bmb2d(B) \quad \{\text{Assumption: } B \text{ is } \text{BMH0-BMH2}\text{-healthy}\}
\]

\[
= d2bmb \circ bmb2d(\text{bmh}_{0,1,2}(B)) \quad \{\text{Lemma C.2.2}\}
\]

Revision: 704f887 (2014-02-04 11:14:10 +0000)
\[ d2bmb\left( \begin{align*}
& \left( \neg ((s, \{\bot\}) \in B \land (s, \emptyset) \in B) \\
& \land \left( \neg \left( ((s, ac' \cup \{\bot\}) \in B ; ac \subseteq ac' \right) \right) \right) \\
& \vdash \left( (s, \{\bot\}) \notin B \land (s, \emptyset) \notin B \\
& \left( (s, ac') \in B ; ac \subseteq ac' \right) \land (s, \{\bot\}) \notin B \land (s, \emptyset) \notin B \\
\end{align*} \right) \}
\]

\[ = \left\{ \begin{align*}
& s : State, ss : \mathbb{P} State_{\bot} \\
& \left( \begin{align*}
& \neg ((s, \{\bot\}) \in B \land (s, \emptyset) \in B) \\
& \land \left( \neg \left( ((s, ac' \cup \{\bot\}) \in B ; ac \subseteq ac' \right) \right) \right) \\
& \vdash \left( (s, \{\bot\}) \notin B \land (s, \emptyset) \notin B \\
& \left( (s, ac') \in B ; ac \subseteq ac' \right) \land (s, \{\bot\}) \notin B \land (s, \emptyset) \notin B \\
& \land \bot \notin ss \\
\end{align*} \right) \} \\
\cup \left\{ \begin{align*}
& \left( \begin{align*}
& \neg ((s, \{\bot\}) \in B \land (s, \emptyset) \in B) \\
& \land \left( \neg \left( ((s, ac' \cup \{\bot\}) \in B ; ac \subseteq ac' \right) \right) \right) \right) \\
& \vdash \left( (s, \{\bot\}) \notin B \land (s, \emptyset) \notin B \\
& \left( (s, ac') \in B ; ac \subseteq ac' \right) \land (s, \{\bot\}) \notin B \land (s, \emptyset) \notin B \\
& \land \bot \in ss \\
\end{align*} \right) \} \\
\]

\{Definition of \( d2bmb \)}

\{Substitution\}

Revision: 704f887 (2014-02-04 11:14:10 +0000) 104
\[
\begin{align*}
\{ \text{Predicate calculus} \} \\
\end{align*}
\]
\[
s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
((\{\bot\}) \in B \land (s, \emptyset) \notin B \land \bot \notin ss)
\]

\[
\lor
\]

\[
((\{s, ac' \cup \{\bot\}\}) \in B \land (s, \emptyset) \notin B \land \bot \notin ss)
\]

\[
\lor
\]

\[
((s, \{\bot\}) \in B \land (s, \emptyset) \in B \land \bot \in ss)
\]

\[
\lor
\]

\[
((s, \{\bot\}) \in B \land (s, \emptyset) \notin B \land \bot \notin ss)
\]

\[
\lor
\]

\[
((s, ss \setminus \{\bot\}) \land (s, \{\bot\}) \notin B \land (s, \emptyset) \notin B \land \bot \in ss)
\]

\[
\{\text{Predicate calculus}\}
\]

\[
s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
((\{\bot\}) \in B \land (s, \emptyset) \in B)
\]

\[
\lor
\]

\[
((s, ac' \cup \{\bot\}) \in B \land (s, \emptyset) \notin B \land \bot \notin ss)
\]

\[
\lor
\]

\[
((s, ac') \in B \land (s, \emptyset) \notin B \land \bot \notin ss)
\]

\[
\lor
\]

\[
((s, ss \setminus \{\bot\}) \land (s, \{\bot\}) \notin B \land (s, \emptyset) \notin B \land \bot \in ss)
\]

\[
\{\text{Type: } \bot \notin ac\}
\]

\[
s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
((\{\bot\}) \in B \land (s, \emptyset) \in B)
\]

\[
\lor
\]

\[
((s, ac' \cup \{\bot\}) \in B \land (s, \emptyset) \notin B \land \bot \notin ss)
\]

\[
\lor
\]

\[
((s, ac') \in B \land (s, \emptyset) \notin B \land \bot \notin ss)
\]

\[
\lor
\]

\[
((s, ss \setminus \{\bot\}) \land (s, \{\bot\}) \notin B \land (s, \emptyset) \notin B \land \bot \in ss)
\]

\[
\{\text{Predicate calculus}\}
\]

\[
s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
((\{\bot\}) \in B \land (s, \emptyset) \in B)
\]

\[
\lor
\]

\[
((s, ac' \cup \{\bot\}) \in B \land (s, \emptyset) \notin B \land \bot \notin ss)
\]

\[
\lor
\]

\[
((s, ac') \in B \land (s, \emptyset) \notin B \land \bot \notin ss)
\]

\[
\lor
\]

\[
((s, ss \setminus \{\bot\}) \land (s, \{\bot\}) \notin B \land (s, \emptyset) \notin B \land \bot \in ss)
\]

\[
\{\text{Predicate calculus}\}
\]
\[
\begin{align*}
\{ \text{s : State, ss : } & \mathbb{P} \text{ State}_{\bot} \\
& \left. \begin{array}{l}
(s, \{\bot\}) \in B \land (s, \emptyset) \in B \\
\lor \\
(s, \{\bot\}) \notin B \land (s, \emptyset) \notin B
\end{array} \right) \} \quad \{ \text{Law B.2.2} \}
\end{align*}
\]

\[
\begin{align*}
&= \text{bmh}_{0,1,2}(B) \quad \{ \text{Assumption: } B \text{ is BMH0-BMH2-healthy} \} \\
&= B
\end{align*}
\]

\textbf{Theorem 5.3.4} \textit{Provided P is A-healthy.}

\[
bmb2d \circ d2bmb(P) = P
\]

\textit{Proof.}

\[
\begin{align*}
bmb2d \circ d2bmb(P) & \quad \{ \text{Assumption: } P \text{ is A-healthy} \} \\
&= bmb2d \circ d2bmb(A(P)) \quad \{ \text{Definition of bmb2d} \} \\
&= ok \Rightarrow \left( \begin{array}{c}
((s, ac') \in d2bmb(A(P)) \land \bot \notin ac' \land ok') \\
\lor \\
((s, ac' \cup \{\bot\}) \in d2bmb(A(P)) \land \bot \notin ac')
\end{array} \right) \quad \{ \text{Definition of d2bmb(A(P)) Lemma C.1.1} \}
\end{align*}
\]
\[
\begin{align*}
(s, ac') &\in \left\{ \begin{array}{l}
\exists ac_0 : \mathbb{P}State \bullet \\
\left( P^f[ac_0/ac'] \\
\lor (P^f[ac_0/ac'] \land \bot \notin ss \land ss \neq \emptyset)
\right) \\
\land\ ac_0 \subseteq ss \\
\land \bot \notin ac' \land ok'
\end{array} \right\} \\
\land \bot \notin ac' \land \bot \notin ac' \land ok'
\end{align*}
\]

\[
\begin{align*}
\exists ac_0 : \mathbb{P}State \bullet \\
\left( P^f[ac_0/ac'] \\
\lor (P^f[ac_0/ac'] \land \bot \notin ac' \land ac' \neq \emptyset)
\right) \\
\land\ ac_0 \subseteq ac' \land \bot \notin ac' \land ok'
\end{align*}
\]

\[
\begin{align*}
\exists ac_0 : \mathbb{P}State \bullet \\
\left( P^f[ac_0/ac'] \\
\lor (P^f[ac_0/ac'] \land \bot \notin (ac' \cup \{\bot\}) \land (ac' \cup \{\bot\}) \neq \emptyset)
\right) \\
\land\ ac_0 \subseteq (ac' \cup \{\bot\}) \land \bot \notin ac'
\end{align*}
\]

\[
\begin{align*}
\exists ac_0 : \mathbb{P}State \bullet P^f[ac_0/ac'] \land ac_0 \subseteq ac' \land \bot \notin ac' \land ok'
\lor \left( \exists ac_0 : \mathbb{P}State \bullet P^f[ac_0/ac'] \land ac_0 \subseteq ac' \land \bot \notin ac' \land ac' \neq \emptyset \land ok'ight)
\lor \left( \exists ac_0 : \mathbb{P}State \bullet P^f[ac_0/ac'] \land ac_0 \subseteq (ac' \cup \{\bot\}) \land \bot \notin ac'ight)
\end{align*}
\]
This result is of fundamental importance since it allows the same programs to be characterised using two different approaches. The binary multirelational model provides a set-theoretic approach, while the predicative theory proposed can easily be linked with other UTP theories of interest, namely the theory of reactive processes.

Furthermore, this dual approach enables us to justify the definition of certain aspects of our theory. This includes the healthiness conditions and the
definition of certain operators such as sequential composition. The most intuitive and appropriate model can be used in each case. The results obtained in either model can then be related using the linking functions.

5.4 Refinement

The healthiness condition $A$ can be understood as a function from the theory of designs into our theory. The theory of designs is a complete lattice [1]. Since $A$ is idempotent and monotonic, a result in [1] establishes that such a function also yields a complete lattice. Therefore we can assert that the theory we propose is also a complete lattice under the implication ordering.

In the following Section 5.4.1 we define the extreme points of the lattice and explore basic properties. Finally, in Section 5.4.2 we prove that the refinement order of our theory corresponds to subset inclusion in the binary multirelational model of Chapter 4.

5.4.1 Extreme points

The extreme points of interest are $\text{Abort} (\perp_{\text{DAC}})$ and $\text{Miracle} (\top_{\text{DAC}})$ as expected of a theory of designs. In what follows we explore these two points and prove that they are $A$-healthy.

Abort

In the original theory of designs the bottom of the lattice is $\text{true}$ and this can be expressed as a design, either $(\text{false} \vdash \text{true})$ or $(\text{false} \vdash \text{false})$. In the theory proposed in [14] the bottom of the lattice is also $\text{true}$. In the theory that we propose, the definition is also $\text{true}$.

Definition 57 (Abort)

$$\perp_{\text{DAC}} \equiv \text{true}$$

A program that aborts provides no guarantees about termination. Indeed it also leaves the set of angelic choices $\text{ac}'$ unrestricted, so the empty set is a possibility. The following Law 5.4.1 establishes that $\text{true}$ is an $A$-healthy predicate.
Law 5.4.1 ($\bot_{\mathcal{D}_A}$-A-healthy)

\[ A(\bot_{\mathcal{D}_A}) = \bot_{\mathcal{D}_A} \]

Proof.

\[ A(\bot_{\mathcal{D}_A}) = A(true) \quad \{\text{Definition of } \bot_{\mathcal{D}_A}\} \]
\[ = A(false \vdash true) \quad \{\text{Property of designs}\} \]
\[ = (\neg \text{PBMH}(true) \vdash \text{PBMH}(true) \land ac' \neq \emptyset) \quad \{\text{Definition of } A\} \]
\[ = (\exists ac_0, ok_0 \cdot true[ac_0, ok_0/\text{ac'}, ok'] \land ac_0 \subseteq \text{ac'}) \quad \{\text{Definition of } \text{PBMH} \text{ and sequential composition}\} \]
\[ = (false \vdash ac' \neq \emptyset) \quad \{\text{Property of substitution and propositional calculus}\} \]
\[ = \bot_{\mathcal{D}_A} \quad \{\text{Definition of design and propositional calculus}\} \]

This result establishes that $\bot_{\mathcal{D}_A}$ is indeed a design in the theory.

Miracle

As explained previously, in the lattice of designs, the top of the lattice is Miracle ($\neg \text{ok}$). In the theory proposed in [14], the top is false. Since in our theory we include the observational variables ok and ok', the top is also $\neg \text{ok}$. This is shown in the following definition.

Definition 58 (Miracle)

\[ \top_{\mathcal{D}_A} \equiv \neg \text{ok} \]

The program $\top_{\mathcal{D}_A}$ corresponds to the design specified as $(true \vdash false)$. The following Law 5.4.2 establishes that $\neg \text{ok}$ is an A-healthy predicate.

Law 5.4.2 ($\top_{\mathcal{D}_A}$-A-healthy)

\[ A(\top_{\mathcal{D}_A}) = \top_{\mathcal{D}_A} \]
Proof.

\[ A(\top_{\text{Dac}}) \]

\[ = A(\neg \text{ok}) \quad \text{\{Definition of } \top_{\text{Dac}}\text{\}} \]

\[ = A(\text{true} \vdash \text{false}) \quad \text{\{Property of designs\}} \]

\[ = (\neg \text{PBMH}(\text{false}) \vdash \text{PBMH}(\text{false}) \land \text{ac'} \neq \emptyset) \quad \text{\{Definition of } \text{PBMH}\text{\}} \]

\[ = (\exists \text{ac}_0, \text{ok}_0 \bullet \text{false}[\text{ac}_0, \text{ok}_0/\text{ac'}, \text{ok}] \land \text{ac}_0 \subseteq \text{ac'}) \quad \text{\{Definition of } \text{PBMH}\text{\}} \]

\[ = (\text{true} \vdash \text{false}) \quad \text{\{Property of designs and propositional calculus\}} \]

\[ = \top_{\text{Dac}} \]

\[ \square \]

The program \( \neg \text{ok} \) is the top of the lattice since it refines any \( A \)-healthy predicate. The proof for the bottom of the lattice, \( \text{true} \), follows directly from the implication ordering. Thus we can establish the following property.

**Law 5.4.3 (Ordering)** For any predicate \( P \) that is \( A \)-healthy.

\[ \bot_{\text{Dac}} \subseteq_{\mathcal{D}} P \subseteq_{\mathcal{D}} \top_{\text{Dac}} \]

**Proof.** Follows from \( A \) monotonic, the definition of \( \top_{\text{Dac}}, \bot_{\text{Dac}} \) and the implication ordering. \( \square \)

This concludes our introduction to the extreme points of the theory. In the following Section 5.4.2 we establish the relationship between the refinement order of this theory and that of the binary multirelational model.

### 5.4.2 Relationship with binary multirelations

The development in Chapter 4 was meant to keep the model as similar as possible to the original model of binary multirelations. In Section 4.4, the refinement order was defined as subset inclusion, like in the original theory. The following Theorem 5.4.1 establishes that in fact the refinement order \( \subseteq_{BM} \) corresponds to the refinement order of designs \( \subseteq_{\mathcal{D}} \) in this theory.
Theorem 5.4.1 Provided $B_0$ and $B_1$ are BMH0-BMH2-healthy.

$bmb2d(B_0) \sqsubseteq_d bmb2d(B_1) \iff B_0 \sqsubseteq_{BMH} B_1$

Proof.

$bmb2d(B_0) \sqsubseteq_d bmb2d(B_1)$ \hspace{1cm} \{Definition of $bmb2d$\}

\[
\begin{align*}
&= \left( (s, ac' \cup \{ \bot \}) \notin B_0 \Rightarrow (s, ac') \in B_0 \right) \\
&\quad \sqsubseteq_d \\
&\quad \left( (s, ac' \cup \{ \bot \}) \notin B_1 \Rightarrow (s, ac') \in B_1 \right) \\
&\quad \left( (s, ac') \notin B_0 \land (s, ac') \in B_1 \Rightarrow (s, ac') \in B_0 \right) \\
&\quad (s, ac' \cup \{ \bot \}) \notin B_0 \Rightarrow (s, ac' \cup \{ \bot \}) \notin B_1 \\
&\quad (s, ac') \notin B_0 \Rightarrow (s, ac') \notin B_1 \\
\end{align*}
\]

\{Refinement of designs\}

\{Predicate calculus\}

Assumption: $B_0$ is BMH1-healthy

\[
\begin{align*}
&= \left( (s, ac' \cup \{ \bot \}) \notin B_0 \Rightarrow (s, ac') \in B_0 \right) \\
&\quad \lor \\
&\quad (s, ac') \notin B_0 \Rightarrow (s, ac') \notin B_1 \\
\end{align*}
\]

\{Predicate calculus\}

\[
\begin{align*}
&= \left( (s, ac' \cup \{ \bot \}) \notin B_0 \lor (s, ac' \cup \{ \bot \}) \in B_0 \Rightarrow (s, ac') \in B_0 \right) \\
&\quad (s, ac') \notin B_0 \Rightarrow (s, ac') \notin B_1 \\
\end{align*}
\]

\{Predicate calculus\}

\[
\begin{align*}
&= \left( (s, ac') \notin B_0 \lor (s, ac') \in B_0 \\
&\quad (s, ac' \cup \{ \bot \}) \notin B_0 \Rightarrow (s, ac') \notin B_1 \\
\end{align*}
\]

\{Predicate calculus\}
It is reassuring to find that the refinement order in our theory of designs with angelic nondeterminism corresponds to subset ordering in the binary multirelational model. This is particularly important as it confirms the intuitive definition of the binary multirelational model.

5.5 Operators

In this section we define the main operators of the theory. This includes the definition of assignment in the following Section 5.5.1 and sequential composition in Section 5.5.2.

5.5.1 Assignment

Similarly to the theory of \[14\], the assignment operator is defined as follows.

**Definition 59 (Assignment)**

\[(x :=_{Dac} e) \triangleq (true \vdash s \oplus (x \mapsto e) \in ac')\]

It is defined by considering the design whose precondition is true, and whose postcondition establishes that every set of final states in \(ac'\) has a component where \(x\) is assigned the value of expression \(e\). This is defined by considering the initial state \(s\) with the value of program variable \(x\) overridden.

5.5.2 Sequential composition

The most challenging aspect of the theory that we propose is its reliance on non-homogeneous relations. This means that sequential composition cannot simply be defined as relational composition like in other \([\text{UTP}]\) theories. This
is an unfortunate consequence. The definition we propose is layered upon that of the sequential composition operator defined in [14].

The definition of sequential composition for designs is defined by considering the auxiliary variables $ok$ and $ok'$ separately. The sequential composition of $P$ and $Q$ is defined as follows.

**Definition 60 ($P \triangle Q$-sequence)**

$$P \triangle Q \equiv \exists ok_0 \cdot P[ok_0/ok] \triangle A Q[ok_0/ok]$$

This definition resembles relational composition with the notable difference that instead of conjunction we use another operator ($\triangle A$) that handles the non-homogeneous alphabet of the relations. This operator corresponds to the definition of sequential composition as introduced in [14], bearing in mind that we have a slightly different alphabet. We present our definition.

**Definition 61 ($P \triangledown Q$-sequence)**

$$P \triangledown Q \equiv P[\{z : \text{State} | Q[z/s]\}/ac']$$

The operator $\triangledown$ handles sequential composition in the relational world with angelic choices [14]. The composition can be understood as follows: a final state of $P \triangledown Q$ is a final state of $Q$ that can be reached from a set of input states $z$ of $Q$ that is available to $P$ as a set $ac'$ of angelic choices.

Perhaps a more intuitive interpretation can be given by considering the operator $\triangledown$ as back propagating the information concerning the valid final states, thus resembling a backtracking operation. In order to understand this definition we introduce the following example from [14].

**Example 15**

$$(s \oplus (x \mapsto 1)) \in ac' \triangledown \begin{cases} (s \oplus (x \mapsto s.x + 1)) \in ac' \\ \land \\ (s \oplus (x \mapsto s.x + 2)) \in ac' \end{cases} \{\text{Definition of } \triangledown \text{ and substitution}\}$$

$$= (s \oplus (x \mapsto 1)) \in \left\{ z \begin{cases} (s \oplus (x \mapsto s.x + 1)) \in ac' \\ \land \\ (s \oplus (x \mapsto s.x + 2)) \in ac' \end{cases} [z/s] \right\} \{\text{Substitution}\}$$
\[
\begin{aligned}
&= (s \oplus (x \mapsto 1)) \in \left\{ z \left| \begin{array}{l}
(z \oplus (x \mapsto z + 1)) \in ac' \\
\land
(z \oplus (x \mapsto z + 2)) \in ac'
\end{array} \right. \right\} \\
&\quad \{\text{Property of sets}\}
&= \left( (s \oplus (x \mapsto 1)) \oplus (x \mapsto (s \oplus (x \mapsto 1)).x + 1)) \in ac' \right) \\
&\quad \land
\left( (s \oplus (x \mapsto 1)) \oplus (x \mapsto (s \oplus (x \mapsto 1)).x + 2)) \in ac' \right)
&\quad \{\text{Record component}\}
&= \left( (s \oplus (x \mapsto 1)) \oplus (x \mapsto 2)) \in ac' \right)
&\quad \land
\left( (s \oplus (x \mapsto 1)) \oplus (x \mapsto 3)) \in ac' \right)
&\quad \{\text{Property of } \oplus\}
&= (s \oplus (x \mapsto 2)) \in ac' \land (s \oplus (x \mapsto 3)) \in ac'
\end{aligned}
\]

In this example we consider the sequential composition of a predicate that assigns 1 to \(x\), followed by the conjunction of two predicates: one that increments the initial value of \(x\) by one, and the other by two. We observe that in [14] conjunction corresponds to angelic choice. If we take that interpretation, then the sequential composition yields two choices for assigning a value to \(x\) in \(ac'\) available to the angel.

In Appendix E we explore and prove the properties observed by the \(;_{\mathcal{A}}\) operator. These results are important for proving and characterising the sequential composition of \(\mathcal{A}\)-healthy designs. The follow theorem establishes this relationship.

**Theorem 5.5.1 (Sequential composition)** Provided ok and ok' are not free in \(P\), \(Q\), \(R\) and \(S\), and that \(\neg P\) and \(Q\) are \(\text{PBMH}\)-healthy.

\[
(P \vdash Q) \; ;_{\mathcal{A}} (R \vdash S)
\]

\[
= (\neg (\neg P \; ;_{\mathcal{A}} \text{true}) \land \neg (Q \; ;_{\mathcal{A}} \neg R) \vdash Q \; ;_{\mathcal{A}} (R \Rightarrow S))
\]

**Proof.**

\[
(P \vdash Q) \; ;_{\mathcal{A}} (R \vdash S) \quad \{\text{Definition of } ;_{\mathcal{A}}\}
\]

\[
= \exists ok_0 \bullet (P \vdash Q)[ok_0/ok'] \; ;_{\mathcal{A}} (R \vdash S)[ok_0/ok] \quad \{\text{Definition of design}\}
\]

\[
= \exists ok_0 \bullet (ok \land P) \Rightarrow (Q \land ok')[ok_0/ok'] \; ;_{\mathcal{A}} ((ok \land R) \Rightarrow (S \land ok'))[ok_0/ok] \quad \{\text{Substitution and assumption}\}
\]

Revision: 704f887 (2014-02-04 11:14:10 +0000) 116
\[
\exists \, \text{ok}_0 \bullet ((\text{ok} \land P) \Rightarrow (Q \land \text{ok}_0)) \quad ; \quad \mathcal{A} ((\text{ok}_0 \land R) \Rightarrow (S \land \text{ok}')) \\
\quad \text{Case-analysis on } \text{ok}_0 \text{ and predicate calculus}
\]

= \begin{align*}
&= \left( (((\text{ok} \land P) \Rightarrow Q) \quad ; \quad \mathcal{A} (R \Rightarrow (S \land \text{ok}'))) \right) \\
&\quad \lor \left( (\neg (\text{ok} \land P) \quad ; \quad \mathcal{A} \text{ true}) \right) \\
&\quad \lor \left( ((\neg \text{ok} \lor \neg P \lor Q) \quad ; \quad \mathcal{A} (R \Rightarrow (S \land \text{ok}'))) \right) \\
&\quad \lor \left( (\neg \text{ok} \lor \neg P) \quad ; \quad \mathcal{A} \text{ true} \right) \\
&\quad \text{Right-distributivity of } ; \quad \mathcal{A} \text{ (Law E.3.1)}
\end{align*}

= \begin{align*}
&= \left( (\neg \text{ok} \quad ; \quad \mathcal{A} (R \Rightarrow (S \land \text{ok}'))) \right) \\
&\quad \lor \left( (\neg P \quad ; \quad \mathcal{A} (R \Rightarrow (S \land \text{ok}'))) \right) \\
&\quad \lor \left( (Q \quad ; \quad \mathcal{A} (R \Rightarrow (S \land \text{ok}'))) \right) \\
&\quad \lor \left( (\neg \text{ok} \quad ; \quad \mathcal{A} \text{ true} \right) \lor (\neg P \quad ; \quad \mathcal{A} \text{ true}) \\
&\quad \text{Law E.1.1 and predicate calculus}
\end{align*}

= \begin{align*}
&= \left( (\neg \text{ok} \lor (\neg P \quad ; \quad \mathcal{A} (R \Rightarrow (S \land \text{ok}'))) \right) \\
&\quad \lor \left( (Q \quad ; \quad \mathcal{A} (R \Rightarrow (S \land \text{ok}'))) \right) \\
&\quad \lor \left( (\neg P \quad ; \quad \mathcal{A} \text{ true}) \right) \\
&\quad \text{Assumption: } \neg P \text{ is PBMH-healthy and Theorem E.8.1}
\end{align*}

= \begin{align*}
&= \left( (\neg \text{ok} \lor (Q \quad ; \quad \mathcal{A} (R \Rightarrow (S \land \text{ok}'))) \right) \\
&\quad \lor \left( (\neg P \quad ; \quad \mathcal{A} \text{ true}) \right) \\
&\quad \text{Assumption: } Q \text{ is PBMH-healthy and Lemma E.8.3}
\end{align*}

= \begin{align*}
&= \left( (\neg \text{ok} \lor (Q \quad ; \quad \mathcal{A} (R \Rightarrow (S \land \text{ok}'))) \right) \\
&\quad \lor \left( (\neg P \quad ; \quad \mathcal{A} \text{ true}) \right) \\
&\quad \text{Predicate calculus}
\end{align*}

= \begin{align*}
&= (ok \land \neg (\neg P \quad ; \quad \mathcal{A} \text{ true}) \land \neg (Q \quad ; \quad \mathcal{A} \neg R)) \\
&\quad \Rightarrow (((Q \quad ; \quad \mathcal{A} (R \Rightarrow S)) \land \text{ok}')) \\
&\quad \text{Definition of design}
\end{align*}
\[
\begin{align*}
\neg (\neg P ; A \text{ true}) \land \neg (Q ; A \neg R) \\
\vdash Q ; A (R \Rightarrow S)
\end{align*}
\]

The result obtained is very similar to that of sequential composition for the original theory of designs [1, 22], except for postcondition and the fact that we use the operator \( ; A \) instead of the sequential composition operator for relations [1]. The implication in the postcondition acts as a filter that removes final states of \( Q \) that fail to satisfy \( R \). We consider the following example.

**Example 16**

\[
(true \vdash \{x \mapsto 1\} \in ac' \land \{x \mapsto 2\} \in ac') ; D_{ac} (s.x \neq 1 \vdash s \in ac')
\]

\[
= \left( \neg (\neg true ; A \text{ true}) \land \neg ((\{x \mapsto 1\} \in ac' \land \{x \mapsto 2\} \in ac') ; A s.x = 1) \right)
\]

\[
\vdash (\{x \mapsto 1\} \in ac' \land \{x \mapsto 2\} \in ac') ; A (s.x \neq 1 \Rightarrow s \in ac')
\]

\{Predicate calculus\}

\[
= \left( \neg (false ; A \text{ true}) \land \neg ((\{x \mapsto 1\} \in ac' \land \{x \mapsto 2\} \in ac') ; A s.x = 1) \right)
\]

\[
\vdash (\{x \mapsto 1\} \in ac' \land \{x \mapsto 2\} \in ac') ; A (s.x \neq 1 \Rightarrow s \in ac')
\]

\{Property of \( ; A \)\}

\[
= \left( \neg false \land \neg ((\{x \mapsto 1\} \in ac' \land \{x \mapsto 2\} \in ac') ; A s.x = 1) \right)
\]

\[
\vdash (\{x \mapsto 1\} \in ac' \land \{x \mapsto 2\} \in ac') ; A (s.x \neq 1 \Rightarrow s \in ac')
\]

\{Predicate calculus\}

\[
= \left( \neg ((\{x \mapsto 1\} \in ac' \land \{x \mapsto 2\} \in ac') ; A s.x = 1) \right)
\]

\[
\vdash (\{x \mapsto 1\} \in ac' \land \{x \mapsto 2\} \in ac') ; A (s.x \neq 1 \Rightarrow s \in ac')
\]

\{Definition of \( ; A \) and substitution\}

\[
= \left( \neg ((\{x \mapsto 1\} \in \{s \mid s.x = 1\} \land \{x \mapsto 2\} \in \{s \mid s.x = 1\}) \right)
\]

\[
\vdash (\{x \mapsto 1\} \in \{s \mid s.x \neq 1 \Rightarrow s \in ac' \} \land \{x \mapsto 2\} \in \{s \mid s.x \neq 1 \Rightarrow s \in ac' \})
\]

\{Property of sets\}
\[
\begin{align*}
&= \left( \neg \{x \mapsto 1\}.x = 1 \land \{x \mapsto 2\}.x = 1 \right) \\
&\quad \vdash (\{x \mapsto 1\}.x \neq 1 \Rightarrow \{x \mapsto 1\} \in ac') \land (\{x \mapsto 2\}.x \neq 1 \Rightarrow \{x \mapsto 2\} \in ac') \text{ \{Value of component } x \text{\}} \\
&= \left( \neg (1 = 1 \land 2 = 1) \\
&\quad \vdash (1 \neq 1 \Rightarrow \{x \mapsto 1\} \in ac') \land (2 \neq 1 \Rightarrow \{x \mapsto 2\} \in ac') \right) \text{ \{Predicate calculus\}} \\
&= \left( \text{true} \\
&\quad \vdash (false \Rightarrow \{x \mapsto 1\} \in ac') \land (true \Rightarrow \{x \mapsto 2\} \in ac') \right) \text{ \{Predicate calculus\}} \\
&= (\text{true} \vdash \{x \mapsto 2\} \in ac')
\end{align*}
\]

In this case, there is an angelic choice between the assignment of the value 1 and 2 to the program variable \(x\), sequentially composed with the program that aborts if \(x\) is 1 and that otherwise behaves as \(\text{Skip}\). The resulting design is just the assignment of 2 to \(x\) that avoids aborting. In the following section we establish closure of the sequential composition operator with respect to \(A\).

If we consider designs that observe \(H3\), we can simplify the result further as there are no dashed variables in the precondition.

\[
(P \vdash Q) \quad ; \quad \text{\(\mathcal{Dac}\)} (R \vdash S) = (P \land (\neg R \; A \neg Q) \vdash (Q \; A (R \Rightarrow S)))
\]

This is similar to the definition of sequential composition for designs where the precondition is a condition [22], except for the use of the operator \(; \quad A\) instead of sequential composition.

**Closure**

It is important that we establish closure of sequential composition \((; \; \text{\(\mathcal{Dac}\)}\) with respect to \(A\). The following closure proof relies on laws established in Appendices [D] and [E].

**Law 5.5.1 ( ; \; \text{\(\mathcal{Dac}\)-A-closure})** Provided \(P\) and \(Q\) are \(A\)-healthy and \(ok\), \(ok'\) are not free in \(P\) and \(Q\).

\[
A(P \; ; \; \text{\(\mathcal{Dac}\)} Q) = P \; ; \; \text{\(\mathcal{Dac}\)} Q
\]

Revision: 704f887 (2014-02-04 11:14:10 +0000) 119
Proof.

\[ P \cap \mathcal{D}_ac \ Q \quad \{ \text{Assumption: } P \text{ and } Q \text{ are } \mathcal{A}-\text{healthy} \} \]
\[ = \mathcal{A}(\neg \ P_f \vdash \ P_t) \cap \mathcal{D}_ac \ \mathcal{A}(\neg \ Q_f \vdash \ Q_t) \quad \{ \text{Definition of } \mathcal{A} \} \]
\[ = \left( \neg PBMH(P_f) \vdash PBMH(P_t) \land ac' \neq \emptyset \right) \cap \ \mathcal{D}_ac \ \left( \neg PBMH(Q_f) \vdash PBMH(Q_t) \land ac' \neq \emptyset \right) \quad \{ \text{Definition of } \cap \mathcal{D}_ac \} \]
\[ = \exists \ ok_0 \cdot \ \left( \left( \neg PBMH(P_f) \vdash PBMH(P_t) \land ac' \neq \emptyset \right)[ok_0/ok'] \cap \ \mathcal{A} \ \left( \neg PBMH(Q_f) \vdash PBMH(Q_t) \land ac' \neq \emptyset \right)[ok_0/ok] \right) \quad \{ \text{Definition of design} \} \]
\[ = \exists \ ok_0 \cdot \ \left( \left( (ok \land \neg PBMH(P_f)) \Rightarrow (PBMH(P_t) \land ac' \neq \emptyset \land ok') \right)[ok_0/ok'] \cap \ \mathcal{A} \ \left( (ok \land \neg PBMH(Q_f)) \Rightarrow (PBMH(Q_t) \land ac' \neq \emptyset \land ok') \right)[ok_0/ok] \right) \quad \{ \text{Substitution and assumption} \} \]
\[ = \exists \ ok_0 \cdot \ \left( \left( (ok \land \neg PBMH(P_f)) \Rightarrow (PBMH(P_t) \land ac' \neq \emptyset \land ok) \right) \cap \ \mathcal{A} \ \left( (ok_0 \land \neg PBMH(Q_f)) \Rightarrow (PBMH(Q_t) \land ac' \neq \emptyset \land ok') \right) \right) \quad \{ \text{Case-analysis on } ok_0 \text{ and predicate calculus} \} \]
\[ = \left( \left( ((ok \land \neg PBMH(P_f)) \Rightarrow (PBMH(P_t) \land ac' \neq \emptyset)) \cap \ \mathcal{A} \ \left( (ok \land \neg PBMH(Q_f)) \Rightarrow (PBMH(Q_t) \land ac' \neq \emptyset) \right) \right) \right) \quad \{ \text{Predicate calculus} \} \]
\[ = \left( \left( (\neg ok \lor PBMH(P_f) \lor (PBMH(P_t) \land ac' \neq \emptyset)) \cap \ \mathcal{A} \ \left( (\neg PBMH(Q_f) \Rightarrow (PBMH(Q_t) \land ac' \neq \emptyset) \right) \right) \right) \quad \{ \text{Right-distributivity of } \cap \mathcal{A} \ (\text{Law E.3.1}) \} \]
\[
\begin{align*}
\text{Law E.1.1 and predicate calculus} & \Rightarrow \\
\text{Theorem E.8.1} & \Rightarrow \\
\text{Lemma E.8.1} & \Rightarrow 
\end{align*}
\]
\[
\begin{align*}
&\neg \text{ok} \\
&\lor ((\text{PBMH}(P^t) \land ac' \neq \emptyset) \land \text{A} \text{PBMH}(Q^t)) \\
&\lor \left( (\text{PBMH}(P^t) \land ac' \neq \emptyset) \land \text{A} (\neg \text{PBMH}(Q^t) \Rightarrow \text{PBMH}(Q^t))) \land ac' \neq \emptyset \land \text{ok}' \right) \\
&\lor (\text{PBMH}(P^f) \land \text{A true}) \\
&\Rightarrow \left( (\text{PBMH}(P^t) \land ac' \neq \emptyset) \land \text{A} (\neg \text{PBMH}(Q^f) \Rightarrow \text{PBMH}(Q^t))) \land ac' \neq \emptyset \land \text{ok}' \right) \\
&\Rightarrow \left( (\text{PBMH}(P^t) \land ac' \neq \emptyset) \land \text{A} (\neg \text{PBMH}(Q^f) \Rightarrow \text{PBMH}(Q^t))) \land ac' \neq \emptyset \right) \\
&\Rightarrow (\neg \text{PBMH}(P^f) \land \text{A true}) \\
&\land (\neg ((\text{PBMH}(P^t) \land ac' \neq \emptyset) \land \text{A} \text{PBMH}(Q^f))) \\
&\land (\neg (((\text{PBMH}(P^t) \land ac' \neq \emptyset) \land \text{A} (\neg \text{PBMH}(Q^f) \Rightarrow \text{PBMH}(Q^t))) \land ac' \neq \emptyset \land \text{ok}' \right) \\
&\Rightarrow (\neg \text{PBMH}(P^f) \land \text{A true}) \\
&\land (\neg ((\text{PBMH}(P^t) \land ac' \neq \emptyset) \land \text{A} \text{PBMH}(Q^f))) \\
&\land (\neg (((\text{PBMH}(P^t) \land ac' \neq \emptyset) \land \text{A} (\neg \text{PBMH}(Q^f) \Rightarrow \text{PBMH}(Q^t))) \land ac' \neq \emptyset \land \text{ok}' \right) \\
&\Rightarrow (\neg \text{PBMH}(P^f) \land \text{A true}) \\
&\land (\neg ((\text{PBMH}(P^t) \land ac' \neq \emptyset) \land \text{A} \text{PBMH}(Q^f))) \\
&\land (\neg (((\text{PBMH}(P^t) \land ac' \neq \emptyset) \land \text{A} (\neg \text{PBMH}(Q^f) \Rightarrow \text{PBMH}(Q^t))) \land ac' \neq \emptyset \land \text{ok}' \right) \\
&\Rightarrow (\neg \text{PBMH}(P^f) \land \text{A true}) \\
&\land (\neg ((\text{PBMH}(P^t) \land ac' \neq \emptyset) \land \text{A} \text{PBMH}(Q^f))) \\
&\land (\neg (((\text{PBMH}(P^t) \land ac' \neq \emptyset) \land \text{A} (\neg \text{PBMH}(Q^f) \Rightarrow \text{PBMH}(Q^t))) \land ac' \neq \emptyset \land \text{ok}' \right) \\
&\Rightarrow (\neg \text{PBMH}(P^f) \land \text{A true}) \\
&\land (\neg ((\text{PBMH}(P^t) \land ac' \neq \emptyset) \land \text{A} \text{PBMH}(Q^f))) \\
&\land (\neg (((\text{PBMH}(P^t) \land ac' \neq \emptyset) \land \text{A} (\neg \text{PBMH}(Q^f) \Rightarrow \text{PBMH}(Q^t))) \land ac' \neq \emptyset \land \text{ok}' \right) \\
\end{align*}
\]

\{Predicate calculus\}

\{Definition of design\}

\{Definition of A0\}

\{Lemma D.4.5 and Laws D.3.1, D.3.2 and E.2.1\}
\[
\begin{align*}
&= A_0 \left( \neg \text{PBMH}(\text{PBMH}(P^f) \ ; A \text{ true}) \right. \\
&\quad \land \left( \neg \text{PBMH}((\text{PBMH}(P^t) \land ac' \neq \emptyset) \ ; A \text{ PBMH}(Q^f)) \right) \\
&\quad \vdash \text{PBMH}((((\text{PBMH}(P^t) \land ac' \neq \emptyset) \ ; A \ (\neg \text{PBMH}(Q^f) \Rightarrow \text{PBMH}(Q^t))) ) \\
&\quad \{\text{Predicate calculus and Law } \text{D.3.1}\} \\
&= A_0 \left( \neg \text{PBMH}((\text{PBMH}(P^f) \ ; A \text{ true}) \right. \\
&\quad \lor \left( ((\text{PBMH}(P^t) \land ac' \neq \emptyset) \ ; A \text{ PBMH}(Q^f)) \right) \\
&\quad \vdash \text{PBMH}((((\text{PBMH}(P^t) \land ac' \neq \emptyset) \ ; A \ (\neg \text{PBMH}(Q^f) \Rightarrow \text{PBMH}(Q^t))) ) \\
&\quad \{\text{Definition of } A_1 \text{ and predicate calculus}\} \\
&= A_0 \circ A_1 \left( \neg \text{PBMH}(\text{PBMH}(P^f) \ ; A \text{ true}) \right. \\
&\quad \land \left( \neg (((\text{PBMH}(P^t) \land ac' \neq \emptyset) \ ; A \text{ PBMH}(Q^f)) \right) \\
&\quad \vdash ((\text{PBMH}(P^t) \land ac' \neq \emptyset) \ ; A \ (\neg \text{PBMH}(Q^f) \Rightarrow \text{PBMH}(Q^t))) ) \\
&\quad \{\text{Theorem } 5.5.1\} \\
&= A_0 \circ A_1 \left( \neg \text{PBMH}(P^f) \vdash \text{PBMH}(P^t) \land ac' \neq \emptyset \right) \\
&\quad \{\text{Definition of } A\} \\
&= A(A(\neg P^f \vdash P^t) \ ; \text{dac } A(\neg Q^f \vdash Q^t)) \\
&\quad \{\text{Assumption: } P \text{ and } Q \text{ are } A\text{-healthy}\} \\
&= A(P \ ; \text{dac } Q) \\
\end{align*}
\]

This result establishes that \(; \text{dac} \) is closed with respect to \( A \) provided both operands are also \( A \)-healthy.

In the following section we justify the definition of the sequential composition operator by proving that it corresponds to the definition of sequential composition for \( BM_\bot \) relations.
Sequential composition in the binary multirelational model

The following Theorem 5.5.2 establishes that for designs that are A-healthy the definitions of sequential composition in both models correspond.

**Theorem 5.5.2**  Provided P and Q are A-healthy.

\[ bmb2d(d2bmb(P) \ ; \ BM \downarrow d2bmb(Q)) = P \ ; \ Dac \ Q \]

**Proof.**

\[ bmb2d(d2bmb(P) \ ; \ BM \downarrow d2bmb(Q)) \quad \{\text{Lemma C.2.9}\} \]

\[ = \text{ok} \Rightarrow \begin{cases} \ldots \end{cases} \quad \{\text{Lemma C.2.8}\} \]

\[ = \text{ok} \Rightarrow \begin{cases} \ldots \end{cases} \quad \{\text{Lemma C.2.7}\} \]

\[ = \text{ok} \Rightarrow \begin{cases} \ldots \end{cases} \quad \{\text{Assumption: } \bot \notin ac'\} \]

Revision: 704f887 (2014-02-04 11:14:10 +0000) 124
\[ \begin{align*}
&= \text{ok} \Rightarrow \\
&\quad \left( (\neg P^f \Rightarrow P^t)[\{s : \text{State} \mid (\neg Q^f \Rightarrow Q^t)\}/ac'] \land ok' \right) \\
&\quad \lor \\
&\quad (P^f[\{s_1 : \text{State} \mid \text{true}\}/ac']) \\
&\quad \lor \\
&\quad (\neg P^f \Rightarrow P^t)[\{s : \text{State} \mid Q^f\}/ac']) \\
&\quad \{\text{Definition of } \; \mathcal{A}\} \\
&= \text{ok} \Rightarrow \\
&\quad \left( (((\neg P^f \Rightarrow P^t) \; ; \mathcal{A} (\neg Q^f \Rightarrow Q^t)) \land ok') \\
&\quad \lor \\
&\quad (P^f \; ; \mathcal{A} \text{ true}) \\
&\quad \lor \\
&\quad (P^f \; ; \mathcal{A} Q^f) \lor (P^t \; ; \mathcal{A} Q^f) \\
&\quad \{\text{Predicate calculus and Law E.3.1}\} \\
&= \text{ok} \Rightarrow \\
&\quad \left( ((P^f \; ; \mathcal{A} (\neg Q^f \Rightarrow Q^t)) \lor (P^t \; ; \mathcal{A} Q^f)) \land ok' \right) \\
&\quad \lor \\
&\quad (P^f \; ; \mathcal{A} \text{ true}) \\
&\quad \lor \\
&\quad (P^f \; ; \mathcal{A} Q^f) \lor (P^t \; ; \mathcal{A} Q^f) \\
&\quad \{\text{Theorem E.8.1 under assumption that } P \text{ is PBMH-healthy}\} \\
&= \text{ok} \Rightarrow \\
&\quad \left( ((P^f \; ; \mathcal{A} Q^f) \lor ((P^t \; ; \mathcal{A} (\neg Q^f \Rightarrow Q^t)) \land ok') \\
&\quad \lor \\
&\quad (P^f \; ; \mathcal{A} \text{ true}) \lor (P^t \; ; \mathcal{A} Q^f) \\
&\quad \{\text{Lemma E.8.3 under assumption that } P \text{ is PBMH-healthy}\} \\
&= \text{ok} \Rightarrow \\
&\quad \left( (P^t \; ; \mathcal{A} Q^f) \lor ((P^t \; ; \mathcal{A} (\neg Q^f \Rightarrow Q^t)) \land ok') \\
&\quad \lor \\
&\quad (P^f \; ; \mathcal{A} \text{ true}) \lor (P^t \; ; \mathcal{A} Q^f) \\
&\quad \{\text{Predicate calculus}\} \\
&= \text{ok} \Rightarrow \\
&\quad \left( (P^t \; ; \mathcal{A} Q^f) \lor ((P^t \; ; \mathcal{A} (\neg Q^f \Rightarrow Q^t)) \land ok') \\
&\quad \lor \\
&\quad (P^f \; ; \mathcal{A} \text{ true}) \\
&\quad \{\text{Predicate calculus}\} \\
\end{align*}\]
\( = \left( \left( ok \land \neg (P^t \land A \land Q^f) \land \neg (P^f \land A \land true) \right) \Rightarrow ((P^t \land A \land (\neg Q^f \Rightarrow Q^f)) \land ok^c) \right) \)  \\
\{Definition of design\}

\( = \left( \left( \neg (P^t \land A \land Q^f) \land \neg (P^f \land A \land true) \right) \right) \)  \\
\{Theorem 5.5.1\}

\( = (\neg P^f \vdash P^t) ; \mathcal{D}_\text{ac} (\neg Q^f \vdash Q^t) \)  \\
{Assumption: \( P \) and \( Q \) are \( A \)-healthy designs}

\( = P ; \mathcal{D}_\text{ac} Q \)

Furthermore, together with the closure of \( ; \mathcal{D}_\text{ac} \), this result enables us to ascertain the closure of \( ; \mathcal{B}_M \).

This concludes our discussion of the definition of sequential composition. In what follows, we concentrate our attention on important properties observed by the sequential composition operator.

**Skip**

Similarly to the original theory of designs, we identify the **Skip** of the theory. We denote it by \( \Pi_{\mathcal{D}_\text{ac}} \) and define it as follows.

**Definition 62**

\[ \Pi_{\mathcal{D}_\text{ac}} \equiv (\text{true} \vdash s \in ac^c) \]

This is a design whose precondition is \text{true}, thus it is always applicable, and upon terminating it establishes that the input state \( s \) is in all sets of angelic choices \( ac^c \). The only results that can be guaranteed by the angel are those that are available in all demonic choices of the value of \( ac^c \) that can be made. In this case, \( s \) is the only guarantee that we have, so the behaviour of \( \Pi_{\mathcal{D}_\text{ac}} \) is to maintain the current state. In the following laws we prove that \( \Pi_{\mathcal{D}_\text{ac}} \) is \( A \)-healthy and that it is the left-unit for sequential composition \( ( ; \mathcal{D}_\text{ac}) \).

**Law 5.5.2 (\( \Pi_{\mathcal{D}_\text{ac}} \)-A-healthy)**

\[ A(\Pi_{\mathcal{D}_\text{ac}}) = \Pi_{\mathcal{D}_\text{ac}} \]
Proof.

\[ A(\Pi_{\text{dac}}) = A(\text{true} \vdash s \in ac') \]

\[ = (\neg \text{PBMH}(\neg \text{true}) \vdash \text{PBMH}(s \in ac') \land ac' \neq \emptyset) \quad \{\text{Lemma [D.4.2]}\} \]

\[ = (\neg \text{false} \vdash \text{PBMH}(s \in ac') \land ac' \neq \emptyset) \quad \{\text{Lemma [D.4.3]}\} \]

\[ = (\neg \text{false} \vdash s \in ac' \land ac' \neq \emptyset) \quad \{\text{Property of sets and predicate calculus}\} \]

\[ = (\text{true} \vdash s \in ac') \quad \{\text{Definition of } I_{\text{I}} \text{Dac}\} \]

\[ = \Pi_{\text{dac}} \]

\[ \square \]

**Law 5.5.3 ( ; \text{dac-left-unit})** Provided \( P \) is a design.

\[ \Pi_{\text{dac}} ; \text{dac} \ P = P \]

Proof.

\[ \Pi_{\text{dac}} ; \text{dac} \ P \]

\[ = (\text{true} \vdash s \in ac') ; \text{dac} (\neg P^f \vdash P^t) \quad \{\text{Theorem [5.5.1]}\} \]

\[ = (\neg (\text{false} ; \text{A} \text{true}) \land \neg (s \in ac' ; \text{A} P^f) \vdash s \in ac' ; \text{A} (\neg P^f \Rightarrow P^t)) \quad \{\text{Predicate calculus}\} \]

\[ = (\neg (\text{false} ; \text{A} \text{true}) \land \neg (s \in ac' ; \text{A} P^f) \vdash s \in ac' ; \text{A} (\neg P^f \Rightarrow P^t)) \quad \{\text{Definition of } \text{A} \text{ and substitution}\} \]

\[ = (\neg (s \in ac' ; \text{A} P^f) \vdash s \in ac' ; \text{A} (\neg P^f \Rightarrow P^t)) \quad \{\text{Predicate calculus}\} \]

\[ = (\neg (s \in ac' ; \text{A} P^f) \vdash s \in ac' ; \text{A} (\neg P^f \Rightarrow P^t)) \quad \{\text{Law [E.7.2]}\} \]

\[ = (\neg P^f \vdash (\neg P^f \Rightarrow P^t)) \quad \{\text{Predicate calculus}\} \]

\[ = (\neg P^f \vdash P^t) \quad \{\text{Definition of design}\} \]

\[ = P \]

\[ \square \]

These laws establish that \( \Pi_{\text{dac}} \) is indeed a suitable definition for \textbf{Skip}.

In what follows we establish that an \textbf{H3}-design in our theory requires the precondition not to mention dashed variables, as expected [1]. We first
show the result of sequentially composing an $A$-healthy design $P$ with $\Pi_{\text{dac}}$ in Law 5.5.4. Finally Law 5.5.5 establishes that $P \; \Pi_{\text{dac}} \; \Pi_{\text{dac}} = P$ restricts the precondition to a condition.

**Law 5.5.4 ( ; dac-sequence-Skip)**  Provided $P$ is $A$-healthy.

$$P \; \Pi_{\text{dac}} = (\neg \exists ac' \bullet P^f \vdash P^t)$$

**Proof.**

$P \; \Pi_{\text{dac}} \Pi_{\text{dac}}$

{Definition of design and $\Pi_{\text{dac}}$}

$= (\neg P^f \vdash P^t) \; \Pi_{\text{dac}} (\text{true} \vdash s \in ac')$

{Theorem 5.5.1}

$= (\neg (P^f \; ; A \text{ true}) \land \neg (P^t \; ; A \text{ false}) \vdash P^t \; ; A \text{ (true } \Rightarrow s \in ac'))$

{Predicate calculus}

$= (\neg (P^f \; ; A \text{ true}) \land \neg (P^t \; ; A \text{ false}) \vdash P^t \; ; A \text{ s } \in ac')$

{Assumption: $P$ is $A$-healthy}

$= \left( \neg (P^f \; ; \text{true} \land \neg ((P^t \land ac' \neq \emptyset) \; ; A \text{ false})) \right)$

{Right-distributivity of $; A$ (Law E.4.1)}

$= \left( \neg (P^f \; ; \text{true} \land \neg ((P^t \land ac' \neq \emptyset) \land \text{false})) \right)$

{Predicate calculus}

$= \left( \neg (P^f \; ; \text{true} \land \neg ((P^t \land ac' \neq \emptyset) \land \text{false})) \right)$

{Assumption: $P$ is $A$-healthy}

$= \left( \neg (P^f \; ; \text{true} \land \neg ((P^t \land ac' \neq \emptyset) \land \text{false})) \right)$

{Predicate calculus}

$= \left( \neg (P^f \; ; \text{true} \land \neg ((P^t \land ac' \neq \emptyset) \land \text{false})) \right)$

{Assumption: $P$ is $A$-healthy}

$= (\neg P^f \; ; A \text{ true}) \vdash P^t \land ac' \neq \emptyset$

{Law E.7.3}

$= (\neg \exists ac' \bullet P^f \vdash P^t \land ac' \neq \emptyset)$

{Law E.5.2}

$= (\neg \exists ac' \bullet P^f \vdash P^t)$

{Assumption: $P$ is $A$-healthy}

$= (\neg \exists ac' \bullet P^f \vdash P^t)$
Law 5.5.5 (H3- ; $\mathcal{D}_{ac}$) Provided $P$ is A-healthy, it is H3-healthy if, and only if, its precondition does not mention $ac'$.

$$(P ; D_{\mathcal{II}D_{ac}}) = P \iff (\exists ac' \cdot \neg P^f = \neg P^f)$$

Proof.

$$(P ; D_{\mathcal{II}D_{ac}}) = P \quad \{\text{Assumption: } P \text{ is A-healthy}\}$$

$$\iff (P ; D_{\mathcal{II}D_{ac}}) = (\neg P^f \vdash P^t \land ac' \neq \emptyset) \quad \{\text{Law 5.5.4}\}$$

$$\iff (\neg \exists ac' \cdot P^f \vdash P^t \land ac' \neq \emptyset) = (\neg P^f \vdash P^t \land ac' \neq \emptyset) \quad \{\text{Equality of designs}\}$$

$$\iff [(\neg \exists ac' \cdot P^f) = \neg P^f] \quad \{\text{Predicate calculus}\}$$

$$\iff [(\exists ac' \cdot P^f) = P^f] \quad \{\text{Predicate calculus (Lemma C.3.1)}\}$$

$$\iff [(\exists ac' \cdot \neg P^f) = \neg P^f] \quad \{\text{Law 5.5.5}\}$$

These results show that we have a theory whose essential properties concerning sequential composition hold as in the original theory of designs [1].

**Sequential composition and the extreme points**

In this section we establish the results of sequentially composing a program with the extreme points of the lattice. As expected, we establish the same left-zero laws that hold in the original theory of designs [1].

The following Law 5.5.6 establishes that it is impossible to recover from an aborting program. Law 5.5.7 establishes that if a design is miraculous then sequentially composing it with another design does not change its behaviour.

**Law 5.5.6**

$${\bot}_{D_{ac}} ; D_{ac} P = {\bot}_{D_{ac}}$$

Proof.

$${\bot}_D ; D_{ac} P \quad \{\text{Definition of } {\bot}_D\}$$

$$= true ; D_{ac} P \quad \{\text{Definition of } ; D_{ac}\}$$

$$= \exists ok_0 \cdot true[ok_0/ok'] ; A P[ok_0/ok] \quad \{\text{Case-split on } ok_0 \text{ and property of substitution}\}$$
\[
\begin{align*}
&= (\text{true} \odot \mathcal{A} \ P[\text{true}/\text{ok}]) \lor (\text{true} \odot \mathcal{A} \ P[\text{false}/\text{ok}]) \quad \{\text{Definition of } \odot \mathcal{A}\} \\
&= \text{true} \lor \text{true} \quad \{\text{Propositional calculus and definition of } \bot_{\mathcal{D}_{\text{ac}}}\} \\
&= \bot_{\mathcal{D}_{\text{ac}}} \\
\end{align*}
\]

\text{Law 5.5.7}

\[
\top_{\mathcal{D}_{\text{ac}}} \odot \mathcal{D}_{\text{ac}} \ P = \top_{\mathcal{D}_{\text{ac}}}
\]

\text{Proof.}

\[
\begin{align*}
\top_{\mathcal{D}_{\text{ac}}} \odot \mathcal{D}_{\text{ac}} \ P & \quad \{\text{Definition of } \top_{\mathcal{D}_{\text{ac}}}\} \\
= (\neg \text{ok}) \odot \mathcal{D}_{\text{ac}} \ P & \quad \{\text{Definition of } \odot \mathcal{D}_{\text{ac}}\} \\
= \exists \text{ok}_0 \bullet (\neg \text{ok})[\text{ok}_0/\text{ok}'] \odot \mathcal{A} \ P[\text{ok}_0/\text{ok}] & \quad \{\text{Substitution and case-split on } \text{ok}_0\} \\
= (\neg \text{ok} \odot \mathcal{A} \ P[\text{true}/\text{ok}]) \lor (\neg \text{ok} \odot \mathcal{A} \ P[\text{false}/\text{ok}]) & \quad \{\text{Definition of } \odot \mathcal{A} \text{ and substitution}\} \\
= \neg \text{ok} & \quad \{\text{Definition of } \top_{\mathcal{D}_{\text{ac}}}\} \\
= \top_{\mathcal{D}_{\text{ac}}} & \quad \square
\end{align*}
\]

Both of these results are expected of a theory of designs [1]. This concludes our discussion of the main operators of the theory and their properties. In the following section we concentrate our attention on nondeterminism.

\section{5.6 Demonic and angelic nondeterminism}

In this section we explore the two types of nondeterminism operators supported by the theory: angelic and demonic choice. We first discuss demonic nondeterminism in Section 5.6.1 followed by angelic nondeterminism in Section 5.6.2. For each operator we establish its closure and the relationship with the corresponding operator in the theory of binary multirelations of Chapter 4. In addition, based on similar results established in the literature [13, 16, 20, 27], we state and prove certain properties of the operators.
5.6.1 Demonic choice

The intuition for the demonic choice in our theory is related to the possible ways of choosing a value for $ac'$. In general, this can be described using disjunction like in the original theory of designs [1].

**Definition 63**

$$P \cap_{\text{dac}} Q \equiv P \lor Q$$

This corresponds to the greatest lower bound of the lattice. We consider the following example, where $\oplus$ is the overriding operator [28].

**Example 17**

$$(x := 1) \cap_{\text{dac}} (x := 2) \quad \{\text{Definition of assignment}\}$$

$$= (\text{true} \vdash s \oplus (x \mapsto 1) \in ac') \cap_{\text{dac}} (\text{true} \vdash s \oplus (x \mapsto 2) \in ac') \quad \{\text{Definition of } \cap_{\text{dac}} \text{ and disjunction of designs}\}$$

$$= (\text{true} \vdash s \oplus (x \mapsto 1) \in ac' \lor s \oplus (x \mapsto 2) \in ac')$$

In this example we have at least two choices for the final value of $ac'$: one has a state where $x$ is 1 and the other has a state where $x$ is 2. The demon can choose any set $ac'$ satisfying either predicate. In this case, the angel is not guaranteed to be able to choose a particular final value for $x$, since there are no choices in the intersection of all possible choices of $ac'$.

**Closure properties**

The demonic choice operator is closed with respect to $A$, provided that both operands are also $A$-healthy. This result follows from the distributive property of $A$ with respect to disjunction, as established by the following Law [5.6.1].

**Law 5.6.1 (A-disjunction-distribute)**

$$A(P \lor Q) = A(P) \lor A(Q)$$

**Proof.**

$$A(P \lor Q) \quad \{\text{Definition of design}\}$$
\[ A((\neg P^f \vdash P^t) \lor (\neg Q^f \vdash Q^t)) \quad \{ \text{Disjunction of designs} \} \]
\[ = A(\neg P^f \land \neg Q^f \vdash P^t \lor Q^t) \quad \{ \text{Predicate calculus} \} \]
\[ = A(\neg (P^f \lor Q^f) \vdash P^t \lor Q^t) \quad \{ \text{Definition of } A \} \]
\[ = (\neg \text{PBMH}(P^f \lor Q^f) \vdash \text{PBMH}(P^t \lor Q^t) \land \text{ac}' \neq \emptyset) \quad \{ \text{Definition of PBMH} \} \]
\[ = \left( \begin{array}{c}
\neg \text{PBMH}(P^f) \lor \text{PBMH}(Q^f) \\
\text{PBMH}(P^t) \lor \text{PBMH}(Q^t) \land \text{ac}' \neq \emptyset
\end{array} \right) \quad \{ \text{Predicate calculus} \} \]
\[ = \left( \begin{array}{c}
\neg \text{PBMH}(P^f) \land \neg \text{PBMH}(Q^f) \\
\text{PBMH}(P^t) \land \text{ac}' \neq \emptyset \lor \text{PBMH}(Q^t) \land \text{ac}' \neq \emptyset
\end{array} \right) \quad \{ \text{Disjunction of designs} \} \]
\[ = A(\neg P^f \vdash P^t) \lor A(\neg Q^f \vdash Q^t) \quad \{ \text{Definition of } A \} \]
\[ = A(\neg P^f \vdash P^t) \lor A(\neg Q^f \vdash Q^t) \]
\[ \square \]

**Law 5.6.2**  Provided \( P \) and \( Q \) are \( A \)-healthy.
\[
A(P \sqcap_{\text{dac}} Q) = P \sqcap_{\text{dac}} Q
\]

**Proof.**
\[
A(P \sqcap_{\text{dac}} Q) \quad \{ \text{Definition of } \sqcap_{\text{dac}} \text{ and Law } 5.6.1 \}
\]
\[
= A(P) \lor A(Q) \quad \{ \text{Assumption: } P \text{ and } Q \text{ are } A \text{-healthy} \}
\]
\[
= P \sqcap_{\text{dac}} Q
\]
\[ \square \]

This concludes the proof for closure of \( \sqcap_{\text{dac}} \) with respect to \( A \).

**Relationship with binary multirelations**

The demonic choice operator \((\sqcap_{\text{dac}})\) corresponds exactly to the demonic choice operator \((\sqcap_{\text{BM}_\bot})\) of the binary multirelational model. This result is established by the following Theorem 5.6.1.
Theorem 5.6.1

\[ bmb2p(B_0 \cap_{BM} B_1) = bmb2p(B_0) \cap_{Dab} bmb2p(B_1) \]

Proof.

\[ bmb2p(B_0 \cap_{BM} B_1) \]
\[ = bmb2p(B_0 \cup B_1) \quad \{ \text{Definition of } \cap_{BM} \} \]
\[ = ok \Rightarrow \left( \left( (s, ac') \in (B_0 \cup B_1) \land \bot \notin ac' \land ok' \right) \lor \left( (s, ac' \cup \{\bot\}) \in (B_0 \cup B_1) \land \bot \notin ac' \right) \right) \quad \{ \text{Property of sets} \} \]
\[ = ok \Rightarrow \left( \left( (((s, ac') \in B_0) \lor (s, ac') \in B_1) \land \bot \notin ac' \land ok' \right) \lor \left( (((s, ac' \cup \{\bot\}) \in B_0) \lor (s, ac' \cup \{\bot\}) \in B_1) \land \bot \notin ac' \right) \right) \quad \{ \text{Propositional calculus} \} \]
\[ = ok \Rightarrow \left( \left( \left( (s, ac') \in B_0 \land \bot \notin ac' \right) \lor \left( (s, ac') \in B_1 \land \bot \notin ac' \right) \land ok' \right) \right) \quad \{ \text{Propositional calculus} \} \]
\[ \Rightarrow \quad \left( \left( \left( \left( (s, ac') \in B_0 \land \bot \notin ac' \right) \lor \left( (s, ac') \in B_1 \land \bot \notin ac' \right) \land ok' \right) \right) \land \neg \left( (s, ac' \cup \{\bot\}) \in B_0 \land \bot \notin ac' \right) \right) \quad \{ \text{Property of designs} \} \]
\[
\begin{align*}
&= \begin{pmatrix}
\neg ((s, ac' \cup \perp) \in B_0 \land \perp \notin ac') \\
\land \\
\neg ((s, ac' \cup \perp) \in B_1 \land \perp \notin ac')
\end{pmatrix} \\
&\vdash \\
\begin{pmatrix}
(s, ac') \in B_0 \land \perp \notin ac' \\
\lor \\
(s, ac') \in B_1 \land \perp \notin ac'
\end{pmatrix}
\{\text{Disjunction of designs and definition of } \land_{\text{ac}}\}
\end{align*}
\]

\[
= \begin{pmatrix}
\neg ((s, ac' \cup \perp) \in B_0 \land \perp \notin ac') \vdash (s, ac') \in B_0 \land \perp \notin ac'
\end{pmatrix}
\{\text{Definition of } \land_{\text{ac}}\}
\]

\[
= \begin{pmatrix}
\neg ((s, ac' \cup \perp) \in B_1 \land \perp \notin ac') \vdash (s, ac') \in B_1 \land \perp \notin ac'
\end{pmatrix}
\{\text{Definition of } bmb2p\}
\]

\[
= bmb2p(B_0) \land_{\text{ac}} bmb2p(B_1)
\]

This result confirms the correspondence of demonic choice in both models. In the following section we focus our attention on its properties.

**Properties**

In general, and since demonic choice is the greatest lower bound, if presented with the possibility to abort \((\perp_{\text{ac}})\), we expect the demon to choose the worst possible outcome as established by the following law.

**Law 5.6.3** \((\neg \land_{\text{ac}})\)

\[P \land_{\text{ac}} \perp_{\text{ac}} = \perp_{\text{ac}}\]

*Proof.*

\[
P \land_{\text{ac}} \perp_{\text{ac}} \quad \{\text{Definition of } \land_{\text{ac}} \text{ and } \perp_{\text{ac}}\}
\]

\[
= P \lor \text{true} \quad \{\text{Propositional calculus and definition of } \perp_{\text{ac}}\}
\]

\[
= \perp_{\text{ac}}
\]

\[\square\]

As observed in the original theory of designs \([\Pi]\), the sequential composition operator distributes through demonic choice, but only from the right as established by ?? and Law \[\square\].
**Law 5.6.4 (\(\sqcap\)-right-distributivity)**

\[
(P \sqcap_{\text{dac}} Q) \sqcap_{\text{dac}} R = (P \sqcap_{\text{dac}} R) \sqcap_{\text{dac}} (Q \sqcap_{\text{dac}} R)
\]

**Proof.**

\[
(P \sqcap_{\text{dac}} R) \sqcap_{\text{dac}} (Q \sqcap_{\text{dac}} R) = (\exists ok_0 \bullet (P[ok_0/ok'] \sqcap_{\mathcal{A}} R[ok_0/ok']) \lor (\exists ok_0 \bullet Q[ok_0/ok'] \sqcap_{\mathcal{A}} R[ok_0/ok]))
\]

\[
= \exists ok_0 \bullet ((P[ok_0/ok'] \lor Q[ok_0/ok']) \sqcap_{\mathcal{A}} R[ok_0/ok])
\]

\[
= (P \sqcap_{\text{dac}} Q) \sqcap_{\text{dac}} R
\]

These results conclude our discussion regarding the demonic choice operator and its properties. In the following section we focus our attention on the angelic choice operator and its respective properties.

### 5.6.2 Angelic choice

In the original theory of designs there is no angelic choice, and therefore the least upper bound of the lattice of designs, defined as conjunction, does not correspond to angelic choice. In other theories, such as in the predicate transformer model, angelic choice is defined exactly as the dual operator of demonic choice [13]. The same is applicable for the model of [14], where angelic choice is defined by conjunction, while demonic choice is disjunction. The definition adopted in our model is also conjunction of designs.

**Definition 64 (\(\sqcup\)\(_{\text{dac}}\))**

\[
P \sqcup_{\text{dac}} Q \equiv P \land Q
\]

This definition is justified by the correspondence with the angelic choice operator of the binary multirelational model of Chapter [13].

To provide the intuition for this definition we consider the following Example [18].
Example 18

\[(x \mapsto 1) \notin ac \vdash (x \mapsto 1) \in ac' \sqcup_{\text{Dac}} (\text{true} \vdash (x \mapsto 2) \in ac')\]

\[
\begin{pmatrix}
(x \mapsto 1) \notin ac' \lor \text{true} \\
\land \\
\text{true} \Rightarrow (x \mapsto 2) \in ac'
\end{pmatrix}
\]

\{
\text{Predicate calculus}
\}

\[
= (\text{true} \vdash (x \mapsto 1) \in ac' \land (x \mapsto 2) \in ac')
\]

It considers the angelic choice between a design that assigns 1 to the only program variable \(x\) but does not necessarily terminate, and a design that assigns 2 to \(x\) but terminates. The result is a program that terminates and, for every set of final states, there is the possibility for the angel to choose the assignment of the value 1 or 2 to \(x\).

**Closure properties**

Having defined angelic choice as the least upper bound operator, in the following Law \([5.6.5]\) we prove that it is closed under \(A\), provided that both operands are \(A\)-healthy.

**Law 5.6.5 (\(\sqcup_{\text{Dac}}\)-A-closed)** Provided \(P\) and \(Q\) are \(A\)-healthy.

\[
A(P \sqcup_{\text{Dac}} Q) = P \sqcup_{\text{Dac}} Q
\]

**Proof.**

\[
P \sqcup_{\text{Dac}} Q
\]

\{
\text{Definition of design}
\}

\[
= (\neg P^t \vdash P^t) \sqcup_{\text{Dac}} (\neg Q^f \vdash Q^f)
\]

\{
\text{Law [A.2.6]}
\}

\[
= (\neg P^f \lor \neg Q^f \vdash (P^f \land Q^f) \lor (P^t \land Q^f) \lor (P^t \land Q^f))
\]

\{
\text{Assumption: } P \text{ and } Q \text{ are } A\text{-healthy}
\}
\[
\begin{align*}
\neg \text{PBMH}(P_f) \lor \neg \text{PBMH}(Q_f) \\
\vdash & \left( (\text{PBMH}(P_f) \land \text{PBMH}(Q_f) \land ac' \neq \emptyset) \lor (\text{PBMH}(P_t) \land ac' \neq \emptyset \land \text{PBMH}(Q_f)) \lor (\text{PBMH}(P_t) \land ac' \neq \emptyset \land \text{PBMH}(Q_t)) \right) \\
\end{align*}
\]

\{Predicate calculus\}

\[
\begin{align*}
\neg (\text{PBMH}(P_f) \land \text{PBMH}(Q_f)) \\
\vdash & \left( (\text{PBMH}(P_f) \land \text{PBMH}(Q_f)) \lor (\text{PBMH}(P_t) \land \text{PBMH}(Q_f)) \lor (\text{PBMH}(P_t) \land \text{PBMH}(Q_t)) \land ac' \neq \emptyset \right) \\
\end{align*}
\]

\{Law [D.2.2]\}

\[
\begin{align*}
\neg \text{PBMH}(\text{PBMH}(P_f) \land \text{PBMH}(Q_f)) \\
\vdash & \left( \text{PBMH}(\text{PBMH}(P_f) \land \text{PBMH}(Q_f)) \lor \text{PBMH}(\text{PBMH}(P_t) \land \text{PBMH}(Q_f)) \lor \text{PBMH}(\text{PBMH}(P_t) \land \text{PBMH}(Q_t)) \right) \land ac' \neq \emptyset \\
\end{align*}
\]

\{Law [D.2.1]\}

\[
\begin{align*}
\neg \text{PBMH}(\text{PBMH}(P_f) \land \text{PBMH}(Q_f)) \\
\vdash & \left( \text{PBMH}(\text{PBMH}(P_f) \land \text{PBMH}(Q_f)) \lor \text{PBMH}(\text{PBMH}(P_t) \land \text{PBMH}(Q_f)) \lor \text{PBMH}(\text{PBMH}(P_t) \land \text{PBMH}(Q_t)) \right) \land ac' \neq \emptyset \\
\end{align*}
\]

\{Definition of A and predicate calculus\}
\[ \neg \text{PBMH}(P_f) \lor \neg \text{PBMH}(Q_f) \]

\[
\vdash \left( \begin{array}{c}
\text{PBMH}(P_f) \land \text{PBMH}(Q_f) \land ac' \neq \emptyset \\
\lor \\
\text{PBMH}(P_t) \land \text{PBMH}(Q_t) \land ac' \neq \emptyset \\
\lor \\
\text{PBMH}(P_t) \land \text{PBMH}(Q_t) \land ac' \neq \emptyset
\end{array} \right)
\]

\[
\{ \text{Assumption: } P \text{ and } Q \text{ are A-healthy}\}
\]

\[
= \text{A}(\neg P_f \lor \neg Q_f \vdash (P_f \land Q_f) \lor (P_t \land Q_f) \lor (P_t \land Q_t)) \quad \{ \text{Law A.2.6} \}
\]

\[
= \text{A}((\neg P_f \vdash P_t) \uplus_d (\neg Q_f \vdash Q_t)) \quad \{ \text{Definition of design} \}
\]

\[
= \text{A}(P \uplus_d Q) \quad \Box
\]

This proof relies on properties of \text{PBMH} and on Law A.2.6 that provides a different result for the least upper bound of designs. Having established closure, in the following section we establish the correspondence with the binary multirelational model of Chapter 4.

**Relationship with binary multirelations**

In the following Theorem 5.6.2 we establish the correspondence of angelic choice in both models. This law requires the operands to be BMH1-healthy. This is satisfied by every binary multirelation that is BMH0-BMH2.

**Theorem 5.6.2** Provided \( B_0 \) and \( B_1 \) are BMH1-healthy.

\[
bmb2p(B_0 \uplus_{BM} B_1) = bmb2p(B_0) \uplus_d bmb2p(B_1)
\]

**Proof.**

\[
bmb2p(B_0) \uplus_d bmb2p(B_1) \quad \{ \text{Definition of } bmb2p \text{ and } \uplus_d \}
\]

\[
= \left( \begin{array}{c}
(s, ac' \cup \{\bot\}) \notin B_0 \lor \bot \in ac' \vdash (s, ac') \in B_0 \land \bot \notin ac'
\end{array} \right)
\]

\[
\vdash \left( \begin{array}{c}
(s, ac' \cup \{\bot\}) \notin B_1 \lor \bot \in ac' \vdash (s, ac') \in B_1 \land \bot \notin ac'
\end{array} \right)
\]

\{Definition of \( \uplus \) for designs\}

Revision: 704f887 (2014-02-04 11:14:10 +0000) 138
\[
\begin{align*}
&= \left( (s, ac' \cup \{\bot\}) \notin B_0 \lor \bot \in ac' \lor (s, ac' \cup \{\bot\}) \notin B_1 \lor \bot \in ac' \right) \\
&\quad \quad \left( (s, ac' \cup \{\bot\}) \notin B_0 \lor \bot \in ac' \implies ((s, ac') \in B_0 \land \bot \notin ac') \right) \\
&\quad \quad \left( (s, ac' \cup \{\bot\}) \notin B_1 \lor \bot \in ac' \implies ((s, ac') \in B_1 \land \bot \notin ac') \right) \\
&\quad \quad \{\text{Propositional calculus}\}
\end{align*}
\]

\[
\begin{align*}
&= \left( (s, ac' \cup \{\bot\}) \notin B_0 \lor \bot \in ac' \lor (s, ac' \cup \{\bot\}) \notin B_1 \right) \\
&\quad \quad \left( ((s, ac' \cup \{\bot\}) \in B_0 \lor (s, ac') \in B_0) \land \bot \notin ac' \right) \\
&\quad \quad \left( ((s, ac' \cup \{\bot\}) \in B_1 \lor (s, ac') \in B_1) \land \bot \notin ac' \right) \\
&\quad \quad \{\text{Assumption: } B_0 \text{ and } B_1 \text{ are BMH1-healthy}\}
\end{align*}
\]

\[
\begin{align*}
&= \left( (s, ac' \cup \{\bot\}) \notin B_0 \lor \bot \in ac' \lor (s, ac' \cup \{\bot\}) \notin B_1 \right) \\
&\quad \quad \left( (s, ac') \in B_0 \land (s, ac') \in B_1 \land \bot \notin ac' \right) \\
&\quad \quad \{\text{Propositional calculus: absorption law}\}
\end{align*}
\]

\[
\begin{align*}
&= \left( (s, ac' \cup \{\bot\}) \notin B_0 \lor \bot \in ac' \lor (s, ac' \cup \{\bot\}) \notin B_1 \right) \\
&\quad \quad \left( \neg ((s, ac' \cup \{\bot\}) \in B_0 \land (s, ac' \cup \{\bot\}) \notin B_1) \lor \bot \in ac' \right) \\
&\quad \quad \{\text{Propositional calculus}\}
\end{align*}
\]

\[
\begin{align*}
&= \left( (s, ac' \cup \{\bot\}) \notin (B_0 \cap B_1) \lor \bot \in ac' \right) \\
&\quad \quad \left( (s, ac') \in (B_0 \cap B_1) \land \bot \notin ac' \right) \\
&\quad \quad \{\text{Definition of } bmb2p\}
\end{align*}
\]

\[
\begin{align*}
&= bmb2p(B_0 \cap B_1) \\
&= bmb2p(B_0 \cup_{BM\bot} B_1) \\
&\quad \quad \{\text{Definition of } \cup_{BM\bot}\}
\end{align*}
\]

\[\square\]

Having established the correspondence of the angelic choice operator in both models, in the following section we focus on its properties.
Properties

In general, and since angelic choice is the least upper bound, the angelic choice of a design $P$ and the top of the lattice ($\top_{\text{Dac}}$) is also $\top_{\text{Dac}}$.

**Law 5.6.6**  Provided $P$ is a design.

$$P \sqcup_{\text{Dac}} \top_{\text{Dac}} = \top_{\text{Dac}}$$

**Proof.**

\[
\begin{align*}
P \sqcup_{\text{Dac}} \top_{\text{Dac}} & \quad \{\text{Definition of } \sqcup_{\text{Dac}} \text{ and } \top_{\text{Dac}}\} \\
= P \land \neg \text{ok} & \quad \{\text{Definition of design}\} \\
= (\neg P^f \vdash P^t) \land \neg \text{ok} & \quad \{\text{Definition of design}\} \\
= ((\text{ok} \land \neg P^f) \Rightarrow (P^t \land \text{ok}')) \land \neg \text{ok} & \quad \{\text{Predicate calculus}\} \\
= (\neg \text{ok} \lor P^f \lor (P^t \land \text{ok}')) \land \neg \text{ok} & \quad \{\text{Predicate calculus: absorption law}\} \\
= \neg \text{ok} & \quad \{\text{Definition of } \top_{\text{Dac}}\} \\
= \top_{\text{Dac}} & \\
\end{align*}
\]

In this model, sequential composition does not necessarily distribute from the right nor from the left. In order to explain the intuition behind this we present the following Counter-example 2 for distribution from the left.

**Counter-example 2**

\[
\begin{align*}
\left(\begin{array}{c}
(\text{true} \vdash s \oplus (x \mapsto 1) \in a'c') \\
\sqcap_{\text{Dac}}
\end{array}\right) \sqcup_{\text{Dac}} \left(\begin{array}{c}
(s.x = 1 \vdash \text{false}) \\
\sqcup
\end{array}\right) \\
\end{align*}
\]

\[
\begin{align*}
\left(\begin{array}{c}
(\text{true} \vdash s \oplus (x \mapsto -1) \in a'c') \\
\sqcap_{\text{Dac}}
\end{array}\right) \sqcup_{\text{Dac}} \left(\begin{array}{c}
(s.x = -1 \vdash \text{false}) \\
\sqcup
\end{array}\right) \\
\end{align*}
\]

\[
\{\text{Assumption: } \sqcup_{\text{Dac}} \text{ distributes over } \sqcap_{\text{Dac}}\}
\]
\[
\begin{align*}
\text{Definition of } & \sqcap \\
\begin{cases}
\text{true } \vdash s \oplus (x \mapsto 1) \in ac' \\
\not\sqsubseteq_{\text{Dac}} (\text{true } \vdash s \oplus (x \mapsto -1) \in ac') \\
\not\sqsubseteq_{\text{Dac}} (\text{true } \vdash s \oplus (x \mapsto 1) \in ac') \\
\not\sqsubseteq_{\text{Dac}} (\text{true } \vdash s \oplus (x \mapsto -1) \in ac')
\end{cases} & ; \text{Dac (} s.x = 1 \vdash \text{false) }
\end{align*}
\]

\[
\begin{align*}
\begin{cases}
\text{true } \vdash s \oplus (x \mapsto 1) \in ac' \\
\not\sqsubseteq_{\text{Dac}} (\text{true } \vdash s \oplus (x \mapsto -1) \in ac')
\end{cases} & ; \text{Dac (} s.x = -1 \vdash \text{false) } \\
\{ \text{Definition of } \sqcap \}
\end{align*}
\]

\[
\begin{align*}
\begin{cases}
\text{true } \vdash s \oplus (x \mapsto 1) \in ac' \lor s \oplus (x \mapsto -1) \in ac' \\
\not\sqsubseteq_{\text{Dac}} (\text{true } \vdash s \oplus (x \mapsto 1) \in ac') \\
\not\sqsubseteq_{\text{Dac}} (\text{true } \vdash s \oplus (x \mapsto -1) \in ac')
\end{cases} & ; \text{Dac (} s.x = 1 \vdash \text{false) } \\
\{ \text{Theorem } \text{5.5.1} \}
\end{align*}
\]

\[
\begin{align*}
\begin{cases}
\text{true } ; \text{A true } \land \\
\not\vdash ((s \oplus (x \mapsto 1) \in ac' \lor s \oplus (x \mapsto -1) \in ac') ; \text{A } s.x \neq 1) \\
\vdash (s \oplus (x \mapsto 1) \in ac' \lor s \oplus (x \mapsto -1) \in ac') ; \text{A } (s.x = 1 \Rightarrow \text{false})
\end{cases} & ; \text{A } (s.x = 1 \Rightarrow \text{false) } \\
\{ \text{Predicate calculus} \}
\end{align*}
\]

\[
\begin{align*}
\begin{cases}
\text{true } ; \text{A true } \land \\
\not\vdash ((s \oplus (x \mapsto 1) \in ac' \lor s \oplus (x \mapsto -1) \in ac') ; \text{A } s.x \neq -1) \\
\vdash (s \oplus (x \mapsto 1) \in ac' \lor s \oplus (x \mapsto -1) \in ac') ; \text{A } (s.x = -1 \Rightarrow \text{false})
\end{cases} & ; \text{A } (s.x = -1 \Rightarrow \text{false) } \\
\{ \text{Property of ; A and propositional calculus} \}
\end{align*}
\]

Revision: 704f887 (2014-02-04 11:14:10 +0000) 141
This is a sequential composition. In the first program the precondition always holds and the program presents a choice to the demon. In this case, the demon can choose the set of final states, $ac'$, by guaranteeing that either $x$ is set to 1 or $-1$ in the final set of states $ac'$. The second program presents an
angelic choice, but the precondition makes a restriction on the value of \( x \) in the initial state \( s \): in either case, if the precondition is satisfied the program is \( \top_{\text{Dac}} \), otherwise if no precondition can be satisfied, the program behaves as \( \bot_{\text{Dac}} \).

It is expected that the angel will avoid \( \bot_{\text{Dac}} \) if that is possible. In this case, it is expected, since the angel can avoid aborting irrespective of the choice the demon makes before the angel. However, if we assume that the sequential composition operator \( ;_{\text{Dac}} \) left-distributes over angelic choice we get a different result as shown above.

In addition, sequential composition does not distribute from the right. We illustrate this problem in Counter-example 3. It is the sequential composition of two designs. The first design is the angelic choice between the program that assigns 2 to \( x \), but may not terminate, and the program that always terminates but whose final set of states \( ac' \) is unrestricted, except that it cannot be the empty set. The second design is miraculous for \( s.x = 2 \) and for every other value of \( s.x \) it aborts.

**Counter-example 3**

\[
\begin{align*}
\left( \begin{array}{c}
(x \mapsto 2) \notin ac' \vdash (x \mapsto 2) \in ac' \\
\uplus_{\text{Dac}} \\
(\text{true} \vdash ac' \neq \emptyset)
\end{array} \right) \quad ;_{\text{Dac}} \quad \begin{array}{c}
\vdash s.x = 2 \\
\vdash \neg s.x \neq 2 \land ac' \neq \emptyset
\end{array}
\end{align*}
\]

\[= \left( \begin{array}{c}
\vdash (x \mapsto 2) \notin ac' \lor \text{true} \\
\vdash (x \mapsto 2) \notin ac' \Rightarrow (x \mapsto 2) \in ac' \\
\land \text{true} \Rightarrow ac' \neq \emptyset
\end{array} \right) \quad ;_{\text{Dac}} \quad \begin{array}{c}
\vdash s.x = 2 \\
\vdash s.x \neq 2 \land ac' \neq \emptyset
\end{array}
\]

\[\quad \text{Definition of} \ \uplus_{\text{Dac}}\]

\[= (\text{true} \vdash (x \mapsto 2) \in ac' \land ac' \neq \emptyset) \quad ;_{\text{Dac}} \quad (s.x = 2 \vdash s.x \neq 2 \land ac' \neq \emptyset)
\]

\[\quad \text{Property of sets and predicate calculus}
\]

\[= (\text{true} \vdash (x \mapsto 2) \in ac') \quad ;_{\text{Dac}} \quad (s.x = 2 \vdash s.x \neq 2 \land ac' \neq \emptyset)
\]

\[\quad \text{Theorem 5.5.1}
\]
\[
\begin{align*}
\neg (\text{false } & ; A \ \text{true}) \land \neg ((x \mapsto 2) \in ac' ; A s.x \neq 2) \\
\vdash (x \mapsto 2) \in ac' ; A (s.x = 2 \Rightarrow (s.x \neq 2 \land ac' \neq \emptyset)) \\
\end{align*}
\]

{\text{Predicate calculus}}

\[
\begin{align*}
\neg (\text{false } & ; A \ \text{true}) \land \neg ((x \mapsto 2) \in ac' ; A s.x \neq 2) \\
\vdash (x \mapsto 2) \in ac' ; A s.x \neq 2 \\
\end{align*}
\]

{Definition of \; A and substitution}

\[
\begin{align*}
\neg \text{false} & \land \neg ((x \mapsto 2) \in \{z \mid z.x \neq 2\}) \\
\vdash (x \mapsto 2) \in \{z \mid z.x \neq 2\} \\
\end{align*}
\]

{Property of sets}

\[
\begin{align*}
\neg \text{false} & \land \neg ((x \mapsto 2).x \neq 2) \\
\vdash (x \mapsto 2).x \neq 2 \\
\end{align*}
\]

{Predicate calculus}

\[
\begin{align*}
\neg (2 \neq 2) & \vdash 2 \neq 2 \\
\end{align*}
\]

{Predicate calculus}

\[
\begin{align*}
\text{true} & \vdash \text{false} \\
\end{align*}
\]

{Predicate calculus and definition of \; \top_{Dac}}

\[
\begin{align*}
\top_{Dac} & \neq \\
\end{align*}
\]

{Predicate calculus}

\[
\begin{align*}
((x \mapsto 2) \notin ac' \land (x \mapsto 2) \in ac') ; Dac (s.x = 2 \vdash s.x \neq 2 \land ac' \neq \emptyset) \\
\upharpoonright_{Dac} (true \vdash ac' \neq \emptyset) ; Dac (s.x = 2 \vdash s.x \neq 2 \land ac' \neq \emptyset) \\
\end{align*}
\]

{Theorem \[5.5.1\]}

\[
\begin{align*}
\neg ((x \mapsto 2) \in ac' ; A \text{true}) & \land \neg ((x \mapsto 2) \in ac' ; A s.x \neq 2) \\
\vdash (x \mapsto 2) \in ac' ; A (s.x = 2 \Rightarrow (s.x \neq 2 \land ac' \neq \emptyset)) \\
\end{align*}
\]

{Predicate calculus}

\[
\begin{align*}
\neg (\text{false } & ; A \ \text{true}) \land \neg (ac' \neq \emptyset ; A s.x \neq 2) \\
\vdash ac' \neq \emptyset ; A (s.x = 2 \Rightarrow (s.x \neq 2 \land ac' \neq \emptyset)) \\
\end{align*}
\]

{Predicate calculus}
In the first case, the angelic choice is resolved first and the result is the program that always terminates and whose set of final states $ac'$ has a state where $x$ is assigned the value 2. Sequentially composing this with the second design results in a miracle ($\top_{\mathcal{D}_{ac}}$) as the only state available for angelic choice is where $x$ has the value 2. However, this is precisely the case in which the design behaves miraculously.

In the second case, we assume that sequential composition distributes through angelic choice. In the resulting angelic choice there are two sequential compositions. In the first one, the result is $\bot_{\mathcal{D}_{ac}}$ as the first design may not...
terminate. While in the second, termination is guaranteed but any final set of states \((ac' \neq \emptyset)\) may fail to satisfy the precondition \(s.x = 2\), in which case the design aborts.

Finally, the demonic and angelic choice operators distribute over one another.

**Law 5.6.7 (demonic-angelic-distributivity)**

\[
P \cap_{\text{dac}} (Q \cup_{\text{dac}} R) = (P \cap_{\text{dac}} Q) \cup_{\text{dac}} (P \cap_{\text{dac}} R)
\]

*Proof.* Follows from the distributive properties of conjunction and disjunction. Equivalently, this follows from the results established in the binary multirelational model of Chapter 4 and the respective isomorphism. 

This result has also been established in other models, such as the predicate transformer model [13]. Since the angelic choice operator is the least upper bound of the lattice, this result follows directly from the properties of the lattice.

### 5.7 Relationship of H3 designs with angelic nondeterminism

In this section we explore the relationship between the theory that we propose and that of [14]. An isomorphism is established for a subset of the theory of designs with angelic nondeterminism that are \(A\) and \(H3\)-healthy.

We begin Section 5.7.1 by characterising the correspondence between the alphabets of the two theories. In Section 5.7.2 and Section 5.7.3 the linking functions between the theories are defined: \(d2pbmh\) that maps from designs into predicates, and \(pbmh2d\) that maps in the inverse direction. We prove that both functions are closed within the respective theories. Finally in Section 5.7.4 the isomorphism is established.

#### 5.7.1 Alphabets

As mentioned previously in Section 5.1 the alphabet of the theory we propose differs slightly from that of [14], in that \(ac'\) is a set of final states, but we
consider undashed variables in the record components instead. In the following Law 5.7.1 we establish that the functions we presented earlier, $\text{acdash}2\text{ac}$ and $\text{ac}2\text{acdash}$, are actually the inverse of each other.

**Law 5.7.1** ($\text{acdash}2\text{ac} \circ \text{ac}2\text{acdash}$)  
$$\text{acdash}2\text{ac} \circ \text{ac}2\text{acdash}(ss) = ss$$

**Proof.**

$$\text{acdash}2\text{ac} \circ \text{ac}2\text{acdash}(ss) = \begin{cases} s_0 : S_{\text{ina}}, s_1 : S_{\text{outa}} \\ s_1 \in \text{ac}2\text{acdash}(ss) \\ \land (\forall x : \alpha \cdot s_0.x = s_1.(x')) \bullet s_0 \end{cases} \begin{cases} s_0 : S_{\text{ina}}, s_1 : S_{\text{outa}} \\ s_1 \in \begin{cases} z_0 : S_{\text{ina}}, z_1 : S_{\text{outa}} \\ \land (\forall x : \alpha \cdot z_0.x = z_1.(x')) \bullet z_1 \end{cases} \end{cases} \begin{cases} s_0 : S_{\text{ina}}, s_1 : S_{\text{outa}} \\ \exists z_0 : S_{\text{ina}} \cdot z_0 \in ss \land (\forall x : \alpha \cdot z_0.x = s_1.(x')) \bullet s_0 \\ \land (\forall x : \alpha \cdot s_0.x = s_1.(x')) \bullet s_0 \end{cases}$$

$$= \{ s_0 : S_{\text{ina}} \mid \exists z_0 : S_{\text{ina}} \cdot z_0 \in ss \land z_0 = s_0 \cdot s_0 \} \begin{cases} \text{One-point rule} \end{cases} = \{ s_0 : S_{\text{ina}} \mid s_0 \in ss \bullet s_0 \} \begin{cases} \text{Property of sets} \end{cases} = ss$$

This means that we can recover the $ac'$ of either theory as needed. Some of the proofs in this section use auxiliary results about these functions that are established in Appendix G.

We observe that we also need to address the fact that we have a single initial state $s$ that encapsulates the values of the initial program variables as record components. This notion is handled directly by the linking functions.
5.7.2 From designs to PBMH predicates (d2pbmh)

The first linking function of interest is d2pbmh that maps from designs that are A and H3-healthy into the theory of [14]. Its definition is presented below.

Definition 65

\[ d_{2pbmh} : A \rightarrow \text{PBMH} \]

\[ d_{2pbmh}(P) \]

\[ \equiv \]

\[ \exists ac_0 \bullet (\neg P^f \Rightarrow P^t)[ac_0/ac'][\text{in}\alpha/s] \land ac2acdash(ac_0) \subseteq ac' \]

For a design P, via the substitution in Pf and Pt, we consider both its pre and postconditions directly. This is sufficient since we require ok to be true and hide ok' (Law A.2.3). The substitution of in\alpha for s corresponds to the substitution of every occurrence of a record component s.x for x, where x is an input program variable. Finally, we substitute ac' in P with the temporary variable ac_0. This allows us to relate the set of final states ac_0 with ac' by applying ac2acdash that replaces every undashed variable in all sets of states in ac_0 into dashed ones. Although the definition considers a superset of ac2acdash(ac_0) rather than equality this is not an issue, since for every P that is A-healthy the sets of final states are always upward closed.

In the following Theorem 5.7.1 we prove that d2pbmh yields predicates that are PBMH-healthy.

Theorem 5.7.1 Provided P is A and H3-healthy.

\[ \text{PBMH}(d_{2pbmh}(P)) = d_{2pbmh}(P) \]

Proof.

\[
\text{PBMH}(d_{2pbmh}(P)) \quad \{\text{Definition of } d_{2pbmh}\}
\]

\[ = \text{PBMH}(\exists ac_0 \bullet (\neg P^f \Rightarrow P^t)[ac_0/ac'][\text{in}\alpha/s] \land ac2acdash(ac_0) \subseteq ac') \quad \{\text{Definition of PBMH}\}
\]

\[ = (\exists ac_0 \bullet (\neg P^f \Rightarrow P^t)[ac_0/ac'][\text{in}\alpha/s] \land ac2acdash(ac_0) \subseteq ac') \land ac \subseteq ac' \quad \{\text{Definition of sequential composition}\}
\]

\[ = \exists ac_1, ac_0 \bullet (\neg P^f \Rightarrow P^t)[ac_0/ac'][\text{in}\alpha/s] \land ac2acdash(ac_0) \subseteq ac_1 \land ac_1 \subseteq ac' \quad \{\text{Transitivity of subset inclusion}\}
\]
\[
\exists a_0 \bullet (\neg P^f \Rightarrow P^t)[ac_0/ac'][in\alpha/s] \land ac_2acdash(ac_0) \subseteq ac' \\
\{\text{Definition of } d2pbmh\}
\]

\[= d2pbmh(P)\]

The upward closure of \(d2pbmh\) follows directly from the definition of \(d2pbmh\). The proviso of Theorem \ref{thm:5.7.1} ensures that the function is only applied to designs that are \(A\) and \(H3\)-healthy.

### 5.7.3 From PBMH predicates to designs \((pbmh2d)\)

In this section we define the second linking function \(pbmh2d\) that maps from predicates in the theory of \cite{14} into designs that are \(A\) and \(H3\)-healthy.

**Definition 66**

\[
pbmh2d : \text{PBMH} \rightarrow A
\]

\[
pbmh2d(P) \equiv \left( \neg P[\emptyset/ac'][s/in\alpha] \vdash \exists a_0 \bullet P[ac_0/ac'][s/in\alpha] \land acdash2ac(ac_0) \subseteq ac' \right)
\]

The definition yields a design whose precondition guarantees successful termination, the postcondition follows the same idea explored in the definition of \(d2pbmh\). Every input program variable \(x\) in \(in\alpha\) is substituted with \(s.x\), where \(s\) is the initial state, and \(ac_0\) is related to \(ac'\) in our theory by application of \(acdashedac\). In the model of \cite{14}, the possibility of non termination occurs when \(ac'\) is the empty set. Therefore the negation of this predicate can be taken as a precondition.

In the following Theorem \ref{thm:5.7.2} we prove that \(pbmh2d\) yields designs that are \(A\) and \(H3\)-healthy.

**Theorem 5.7.2** Provided \(P\) is satisfies PBMH.

\[
A \circ H3(pbmh2d(P)) = pbmhd2d(P)
\]

**Proof.**

\[
A \circ H3(pbmh2d(P)) \quad \{\text{Definition of } pbmh2d\}
\]
Similarly to the definition of $d2pbmh$, the proviso of Theorem 5.7.2 ensures that the function is only applied to predicates that are $\text{PBMH}$-healthy.
5.7.4 Isomorphism: $d2pbmh$ and $pbmh2d$

In this section we establish that the linking functions $d2pbmh$ and $pbmh2d$ are bijections. This result is established by Theorems 5.7.3 and 5.7.4.

Theorem 5.7.3 Provided $P$ is $A \circ H3$-healthy.

$$pbmh2d \circ d2pbmh(P) = P$$

Proof.

$$pbmh2d \circ d2pbmh(P) = pbmh2d(\exists ac_0 \bullet (\neg P^f \Rightarrow P^t)[ac_0/ac'][in\alpha/s] \land ac2acdash(ac_0) \subseteq ac') \quad \{\text{Definition of } d2pbmh\}$$

$$= \left( \neg \left( \exists ac_0 \bullet (\neg P^f \Rightarrow P^t)[ac_0/ac'][in\alpha/s] \land ac2acdash(ac_0) \subseteq ac' \right) \quad \{\text{Definition of } pbmh2d\} \right) \left[\emptyset/ac'][s/in\alpha]\right]$$

$$= \left( \exists ac_0 \bullet \left( \exists ac_0 \bullet (\neg P^f \Rightarrow P^t)[ac_0/ac'][in\alpha/s] \land ac2acdash(ac_0) \subseteq ac' \right) \quad \{\text{Variable renaming}\} \right) [ac_0/ac'][s/in\alpha]$$

$$= \left( \exists ac_0 \bullet \left( \exists ac_1 \bullet (\neg P^f \Rightarrow P^t)[ac_1/ac'][in\alpha/s] \land ac2acdash(ac_1) \subseteq ac' \right) \quad \{\text{Substitution}\} \right) [ac_0/ac'][s/in\alpha]$$

$$= \left( \exists ac_0, ac_1 \bullet (\neg P^f \Rightarrow P^t)[ac_1/ac'][ac2acdash(ac_1) \subseteq ac_0 \land acdash2ac(ac_1) \subseteq ac'] \quad \{\text{Property of } ac2acdash \text{ and } acdash2ac\} \right)$$
\[
\begin{align*}
\neg \left( \exists a_{c_0} \cdot (\neg P^f \Rightarrow P^t)[a_{c_0}/a_c] \land \neg a_{c_2} a_{c_0} \circ a_2 a_{c_2}(a_{c_0}) \subseteq a_{c_2} a_{c_0}(0) \right) \\
\vdash \exists a_{c_0}, a_{c_1} \cdot (\neg P^f \Rightarrow P^t)[a_{c_1}/a_c] \\
\land a_{c_2} a_{c_0} \circ a_2 a_{c_2}(a_{c_1}) \subseteq a_{c_2} a_{c_0}(a_{c_0}) \\
\land a_{c_2} a_{c_0}(a_{c_1}) \subseteq a_{c_1}
\end{align*}
\]

\{Transitivity of subset inclusion and Law G.1.3\}

\[
\begin{align*}
\neg \left( \exists a_{c_0} \cdot (\neg P^f \Rightarrow P^t)[a_{c_0}/a_c] \land a_{c_0} \subseteq 0 \right) \\
\vdash \exists a_{c_1} \cdot (\neg P^f \Rightarrow P^t)[a_{c_1}/a_c] \land a_{c_1} \subseteq a_c
\end{align*}
\]

\{Case-analysis on \(a_{c_0}\) and definition of sequential composition\}

\[
\begin{align*}
\neg (\neg P^f \Rightarrow P^t)[a_{c_0}/a_c][0/a_{c_0}] \\
\vdash (\neg P^f \Rightarrow P^t) \land a_{c} \subseteq a_c
\end{align*}
\]

\{Substitution and definition of \(PBMH\)\}

\[
\begin{align*}
\neg (\neg P^f \Rightarrow P^t)[0/a_c] \vdash PBMH(\neg P^f \Rightarrow P^t) \\
\{Assumption: P is A \circ H3-healthy\}
\end{align*}
\]

\[
\begin{align*}
\neg (\neg P^f \Rightarrow (PBMH(P^t) \land ac' \neq 0))[0/a_c] \\
\vdash PBMH(\neg P^f \Rightarrow (PBMH(P^t) \land ac' \neq 0))
\end{align*}
\]

\{Substitution under assumption that \(ac'\) is not free in \(P^f\)\}

\[
\begin{align*}
\neg (\neg P^f \Rightarrow (PBMH(P^t) \land \emptyset \neq 0)) \\
\vdash PBMH(\neg P^f \Rightarrow (PBMH(P^t) \land ac' \neq 0))
\end{align*}
\]

\{Property of sets and predicate calculus\}

\[
\begin{align*}
(\neg P^f \vdash PBMH(P^f \lor (PBMH(P^t) \land ac' \neq 0)))
\{Law D.2.1\}
\end{align*}
\]

\[
(\neg P^f \vdash PBMH(P^f) \lor PBMH(PBMH(P^t) \land ac' \neq 0))
\{Lemma D.4.5 and Law D.2.2\}
\]

\[
(\neg P^f \vdash PBMH(P^f) \lor PBMH(PBMH(P^t) \land ac' \neq 0))
\{Law D.1.1 and Lemma D.4.6\}
\]

\[
(\neg P^f \vdash P^f \lor PBMH(P^t) \land ac' \neq 0)
\{Property of sets and predicate calculus\}
\]

\[
(\neg P^f \land (\neg P^f \lor \neg ok') \vdash PBMH(P^t) \land ac' \neq 0)
\{Predicate calculus: absorption law\}
\[\neg P^t \vdash \text{PBMH}(P^t) \land ac' \neq \emptyset\]  \hspace{1cm} \{\text{Definition of } A \circ \text{H3}\}

\[A \circ \text{H3}(P)\]  \hspace{1cm} \{\text{Assumption: } P \text{ is } A \circ \text{H3}-healthy\}

\[= P\]

\[\square\]

**Theorem 5.7.4**  Provided \(P\) is \text{PBMH}-healthy.

\[d2pbmh \circ \text{pbmh2d}(P) = P\]

**Proof.**

\[d2pbmh \circ \text{pbmh2d}(P)\]  \hspace{1cm} \{\text{Definition of } \text{pbmh2d}\}

\[= d2pbmh \left( \neg P[\emptyset/ac'][s/in\alpha] \right)\]  \hspace{1cm} \{\text{Definition of } d2pbmh\}

\[= \left( \exists ac_0 \cdot P[ac_0/ac'][s/in\alpha] \land acdash 2ac(ac_0) \subseteq ac' \right)\]  \hspace{1cm} \{\text{Variable renaming}\}

\[\left( \exists ac_0 \cdot P[ac_1/ac'][s/in\alpha] \land acdash 2ac(ac_1) \subseteq ac' \right)\]  \hspace{1cm} \{\text{Predicate calculus and substitution}\}

\[\left( \exists ac_0, ac_1 \cdot P[ac_1/ac'][s/in\alpha][ac_0/ac'][in\alpha/s] \land acdash 2ac(ac_0) \subseteq ac' \right)\]  \hspace{1cm} \{\text{Substitution}\}

Revision: 704f887 (2014-02-04 11:14:10 +0000) 153
While this is an expected result, it is reassuring that the subset of our theory that is \textbf{H3}-healthy is in exact correspondence with the \textbf{UTP} theory of [14].

We observe that the subset of the binary multirelational model of Chapter 4 that is \textbf{BMH3}-healthy is isomorphic to the original theory of binary multirelations. Since binary multirelations are also isomorphic to the \textbf{UTP} theory of [14], the result presented in this section is also in agreement. This result completes the relationships depicted in Figure 1.1.

5.8 Final considerations

In this chapter we have presented a new \textbf{UTP} theory of designs that is capable of modelling angelic and demonic nondeterminism. The novel contribution lies in the use of the variables \(ok\) and \(ok'\), as in every theory of designs, and the capability to express non-\textbf{H3}-designs as well as both demonic and angelic choice. While all known existing models for program correctness restrict
their attention to necessarily terminating programs, we relax this constraint in order to pave the way for the development of a theory of reactive designs with angelic nondeterminism.

The healthiness conditions of the theory have been presented and their properties proved, including idempotency and monotonicity. Through the co-development of the binary multirelational model in Chapter 4 and its subsequent isomorphism, we have been able to justify and explore the definition of the operators and the refinement ordering. It is reassuring to know that the refinement order as given by universal reverse implication corresponds to subset inclusion in the binary multirelational model.

Perhaps the most challenging aspect of the theory is that it is non-homogeneous. As a consequence sequential composition cannot be defined as relational composition. While the definition for sequential composition is not immediately obvious, it is more intuitive when considered in the equivalent binary multirelational model.

Finally, we have also linked a subset of this model with the UTP theory of $\mathcal{UTP}$. This is a complementary result to the link between the binary multirelational model of $BM_\perp$ relations and that of the original theory of binary multirelations. This gives us further assurance as to the capability to express the existing theories as a subset of our own correctly.
Chapter 6

Conclusions

In this chapter we present a summary of our findings in Section 6.1. This is followed by the discussion of future work in Section 6.2.

6.1 Summary

The concept of angelic nondeterminism is useful in the context of formal specifications. It has traditionally been studied in the context of the refinement calculus [11–13]. However, as far as we know, it has not been characterised in a relational setting capable of modelling reactive programs.

In this work we have presented a new UTP theory of designs with angelic nondeterminism that can cope with non-$H3$ designs, a first step in the definition of a theory of reactive designs with angelic nondeterminism. The healthiness conditions and the main operators have been defined and their properties proved.

In order to motivate our predicative model, we developed an equivalent extended binary multirelational model. This provides an insight into the definition of some of the operators, such as sequential composition. Its definition in the binary multirelational model is based on our understanding of the original theory of designs [1] and the theory of binary multirelations [15].

Unfortunately, in the model we propose sequential composition cannot be defined as relational composition as we use non-homogenous relations. Instead, we provide an alternative definition that is partially based on substitution as proposed by Cavalanti et al. [14]. We extend that notion for non-$H3$ designs and justify its definition with the isomorphism between the
models. It is pleasing that our definition resembles that of the theory of designs.

For both of the models that we have developed, we have studied the relationship between their subsets of interest and the existing theories. The fact that we have been able to prove that they are equivalent is reassuring. These results consolidate our understanding of the models.

6.2 Future work

As already mentioned, the theory proposed in this work is the first step towards the definition of a theory of reactive designs with angelic nondeterminism. Although the results we have obtained are consistent with a theory of designs, it remains to be seen what are the implications with respect to a theory of reactive programs.

In addition, since there is a collection of different binary multirelational models as pointed out by Rewitzky [15], it would be interesting to explore whether other isomorphisms can be established. In fact, exploring the relationship between our model and any other existing theories would further help validate the model and consolidate our understanding of it.

Since it is our goal to provide a mathematically rigorous theory for software engineering, it is only recommended that, in the future, further validation of all applicable theorems and lemmas is carried out by mechanising the theory with the help of a theorem prover.

Finally, due to the foundational importance of our contribution, it would be desirable if this model could be exploited in practice, perhaps even in the context of unforeseen domains.
Acronyms

**CSP**  Communicating Sequential Processes
**ZRC**  Z Refinement Calculus
**VDM**  Vienna Development Method
**ASM**  Abstract State Machine
**FSM**  Finite State Machines
**CCS**  Calculus of Concurrent Systems
**JCSP**  Java Communicating Sequential Processes
**FDR**  Failures-Divergence Refinement
**UTP**  Unifying Theories of Programming
**BNF**  Backus-Naur Normal Form
Bibliography


Appendix A

Theory of designs

A.1 Healthiness conditions

H2A

Definition 67

\[ \text{H2A}(P) \equiv \neg P^f \Rightarrow (P^t \land ok') \]

Law A.1.1 (H2A ⇔ H2) *The definition of H2A implies that the fixpoints are the same as those of H2.*

Proof for implication. The following proof is based on [29].

\[
P = \exists ok_0 \cdot P \land ok' = ok_0 \quad \{ \text{Introduce fresh variable and substitution} \}
\]

\[
= (\neg ok' \land P^f) \lor (ok' \land P^t) \quad \{ \text{Case-split on } ok_0 \}
\]

\[
= ((\neg ok' \land P^f) \lor ok') \land P^t \quad \{ \text{Assumption: P is H2-healthy} \}
\]

\[
= ((P^f \lor ok') \land P^t) \quad \{ \text{Propositional calculus} \}
\]

\[
= (P^f \land P^t) \lor (ok' \land P^t) \quad \{ \text{Assumption: P is H2-healthy} \}
\]

\[
= P^f \lor (ok' \land P^t) \quad \{ \text{Propositional calculus} \}
\]

\[
= \neg P^f \Rightarrow (P^t \land ok')
\]

\[ \Box \]
Proof for reverse implication.

\[
[(\text{H2A}(P))^f \Rightarrow (\text{H2A}(P))^t]
\]

\begin{align*}
&= [\neg P^f \Rightarrow (P^t \land ok')^f \Rightarrow (\neg P^f \Rightarrow (P^t \land ok'))^t] \quad \{\text{Definition of H2A}\} \\
&= [(P^f \Rightarrow (\neg P^f \Rightarrow P^t)^t)] \quad \{\text{Substitution}\} \\
&= [\neg P^f \lor P^f \lor P^t] \quad \{\text{Propositional calculus}\} \\
&= true
\end{align*}

\[\square\]

A.2 Lemmas

Law A.2.1 (design-true-ok') Provided ok \land P and ok' is not free in P.

\[(P \vdash Q)^t = Q\]

Proof. As stated and proved in [30] (Lemma 4.2).

\[\square\]

Law A.2.2 (design-false-ok') Provided ok' is not free in P.

\[ok \land \neg (P \vdash Q)^f = ok \land P\]

Proof. As stated and proved in [30] (Lemma 4.3).

\[\square\]

Law A.2.3 (design-exists-ok')

\[\exists ok' \bullet (P \vdash Q) = (ok \land P) \Rightarrow Q\]

Proof.

\[
\begin{align*}
\exists ok' \bullet (P \vdash Q) &\quad \{\text{Definition of design}\} \\
= \exists ok' \bullet (ok \land P) \Rightarrow (Q \land ok') &\quad \{\text{Case-split on ok'}\} \\
= ((ok \land P) \Rightarrow Q) \lor \neg (ok \land P) &\quad \{\text{Propositional calculus}\} \\
= (ok \land P) \Rightarrow Q
\end{align*}
\]

\[\square\]
Law A.2.4 (design-⊔)

\[ (\neg P^f \vdash P^t) \sqcup (\neg Q^f \vdash Q^t) \]

\[ = (\neg P^f \lor \neg Q^f \vdash (\neg P^f \Rightarrow P^t) \land (\neg Q^f \Rightarrow Q^t)) \]

Proof.

\[ (\neg P^f \vdash P^t) \sqcup (\neg Q^f \vdash Q^t) \]

\[ = ((\neg P^f \vdash P^t) \lor (\neg Q^f \vdash Q^t)) \]

\[ = ((\neg P^f \vdash P^t) \land (\neg Q^f \vdash Q^t)) \]

\[ = (\neg P^f \lor \neg Q^f \vdash (\neg P^f \Rightarrow P^t) \land (\neg Q^f \Rightarrow Q^t)) \]

Law A.2.5 (design-exists-ok'-⊔) Provided P and Q are designs.

\[ \exists ok' \bullet (P \land Q) = (\exists ok' \bullet P) \land (\exists ok' \bullet Q) \]

Proof.

\[ (\exists ok' \bullet P) \land (\exists ok' \bullet Q) \]

\[ = (\exists ok' \bullet (\neg P^f \vdash P^t)) \land (\exists ok' \bullet (\neg Q^f \vdash Q^t)) \]

Revision: 704f887 (2014-02-04 11:14:10 +0000) 165
\[
\begin{align*}
& ((\text{ok} \land \neg P^f \Rightarrow P^t) \land ((\text{ok} \land \neg Q^f \Rightarrow Q^t)) \quad \{\text{Propositional calculus}\} \\
& = (\text{ok} \Rightarrow (P^t \lor P^i)) \land (\text{ok} \Rightarrow (Q^t \lor Q^i)) \quad \{\text{Propositional calculus}\} \\
& = \text{ok} \Rightarrow ((P^t \lor P^i) \land (Q^t \lor Q^i)) \quad \{\text{Propositional calculus: absorption law}\} \\
& = \text{ok} \Rightarrow (((P^f \land Q^f) \lor P^t) \land ((P^f \land Q^f) \lor Q^t) \lor Q^i) \quad \{\text{Propositional calculus}\} \\
& = \text{ok} \Rightarrow (((P^f \land Q^f) \lor (P^i \lor Q^i)) \land ((P^f \land Q^f) \lor Q^t) \lor Q^i) \quad \{\text{Law A.2.3}\} \\
& = \exists \text{ok}' \bullet (\neg (P^f \land Q^f) \vdash (\neg P^f \Rightarrow P^t) \land (\neg Q^f \Rightarrow Q^i)) \quad \{\text{Conjunction of designs}\} \\
& = \exists \text{ok}' \bullet (\neg P^f \vdash P^t) \land (\neg Q^f \vdash Q^t) \quad \{\text{Assumption: } P \text{ and } Q \text{ are designs}\} \\
& = \exists \text{ok}' \bullet (P \land Q) \\
\end{align*}
\]

\textbf{Law A.2.6}

\[
(\neg P^f \vdash P^t) \sqcup (\neg Q^f \vdash Q^t) = \\
(\neg P^f \lor \neg Q^f \vdash (P^f \land Q^f) \lor (P^i \land Q^f) \lor (P^t \land Q^i))
\]

\textit{Proof.}

\[
\begin{align*}
& (\neg P^f \vdash P^t) \sqcup (\neg Q^f \vdash Q^t) \quad \{\text{Conjunction of designs}\} \\
& = (\neg P^f \lor \neg Q^f \vdash (P^f \Rightarrow P^t) \land (\neg Q^f \Rightarrow Q^t)) \quad \{\text{Propositional calculus}\} \\
& = (\neg P^f \lor \neg Q^f \vdash (P^f \lor P^i) \land (Q^f \lor Q^i)) \quad \{\text{Predicate calculus}\} \\
& = (\neg (P^f \land Q^f) \vdash (P^f \land Q^f) \lor (P^f \land Q^i) \lor (P^i \land Q^f) \lor (P^t \land Q^t) \lor (P^f \land Q^f)) \quad \{\text{Definition of design}\} \\
& = \left( (\text{ok} \land \neg (P^f \land Q^f)) \Rightarrow \right) \quad \{\text{Predicate calculus}\}
\]

\[
\begin{align*}
& (((P^f \land Q^f) \lor (P^i \land Q^f) \lor (P^t \land Q^i) \lor (P^f \land Q^f)) \land \text{ok}')
\end{align*}
\]
\[
\begin{align*}
&= \left( (ok \land \neg (P_f \land Q_f) \land (\neg (P_f \land Q_f) \lor \neg ok')) \Rightarrow \\
&\quad \left( (((P_f \land Q_f^t) \lor (P_t \land Q_f^t) \lor (P_t \land Q_f^t)) \land ok') \right) \right) \\
&\quad \{\text{Predicate calculus: absorption law}\}
\end{align*}
\]

\[
= (ok \land \neg (P_f \land Q_f^t)) \Rightarrow (((P_f \land Q_f^t) \lor (P_t \land Q_f^t) \lor (P_t \land Q_f^t)) \land ok') \\
\quad \{\text{Definition of design}\}
\]

\[
= (\neg (P_f \land Q_f^t) \vdash (P_f \land Q_f^t) \lor (P_t \land Q_f^t) \lor (P_t \land Q_f^t)) \\
\quad \{\text{Predicate calculus}\}
\]

\[
= (\neg P_f \lor \neg Q_f^t \vdash (P_f \land Q_f^t) \lor (P_t \land Q_f^t) \lor (P_t \land Q_f^t))
\]

\[\square\]
Appendix B

Binary multirelational model

B.1 Healthiness conditions

\( \text{bmh}_0 \)

Lemma B.1.1 (\( \text{bmh}_0 \)-idempotent)

\( \text{bmh}_0 \circ \text{bmh}_0(B) = \text{bmh}_0(B) \)

Proof.

\[
\text{bmh}_0 \circ \text{bmh}_0(B) = \left\{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \mid \exists ss_0 \bullet (s, ss_0) \in \text{bmh}_0(B) \land ss_0 \subseteq ss \land (\bot \in ss_0 \iff \bot \in ss) \right\}
\]

\[
= \left\{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \mid \exists ss_0 \bullet (s, ss_0) \in \text{bmh}_0(B) \land ss_0 \subseteq ss \land (\bot \in ss_0 \iff \bot \in ss) \right\}
\]

\[
= \left\{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \mid \exists ss_0 \bullet (s, ss_0) \in \{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \mid \exists ss_0 \bullet (s, ss_0) \in B \land ss_0 \subseteq ss \land (\bot \in ss_0 \iff \bot \in ss) \} \right\}
\]

\[
= \left\{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \mid \exists ss_0 \bullet (s, ss_0) \in \{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \mid \exists ss_1 \bullet (s, ss_1) \in B \land ss_1 \subseteq ss \land (\bot \in ss_1 \iff \bot \in ss) \} \right\}
\]

\[
= \left\{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \mid \exists ss_0 \bullet (s, ss_0) \in \{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \mid \exists ss_1 \bullet (s, ss_1) \in B \land ss_1 \subseteq ss \land (\bot \in ss_1 \iff \bot \in ss) \} \right\}
\]
\[
\begin{align*}
= & \left\{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
& \quad \exists ss_0, ss_1 \cdot (s, ss_1) \in B \\
& \quad \land ss_1 \subseteq ss_0 \land (\bot \in ss_1 \iff \bot \in ss_0) \\
& \quad \land ss_0 \subseteq ss \land (\bot \in ss_0 \iff \bot \in ss) \right\} \\
& \quad \text{\{Predicate calculus and transitivity of subset inclusion\}} \\
= & \left\{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
& \quad \exists ss_1 \cdot (s, ss_1) \in B \land ss_1 \subseteq ss \land (\bot \in ss_1 \iff \bot \in ss) \right\} \\
& \quad \text{\{Definition of bmh}_0 \text{\}} \\
= & \text{bmh}_0(B)
\end{align*}
\]

\textbf{bmh}_1

Lemma B.1.2 (bmh}_1\text{-idempotent)

\[\text{bmh}_1 \circ \text{bmh}_1(B) = \text{bmh}_1(B)\]

Proof.

\[
\begin{align*}
\text{bmh}_1 \circ \text{bmh}_1(B) \\
= & \left\{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \mid (s, ss \cup \{\bot\}) \in \text{bmh}_1(B) \lor (s, ss) \in \text{bmh}_1(B) \right\} \\
& \quad \text{\{Definition of bmh}_1 \text{\}} \\
= & \left\{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
& \quad (s, ss \cup \{\bot\}) \in \left\{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
& \quad \mid (s, ss \cup \{\bot\}) \in B \lor (s, ss) \in B \right\} \\
& \quad \lor \\
& \quad (s, ss) \in \{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \mid (s, ss \cup \{\bot\}) \in B \lor (s, ss) \in B \} \right\} \\
& \quad \text{\{Property of sets\}} \\
= & \left\{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
& \quad (s, ss \cup \{\bot\} \cup \{\bot\}) \in B \lor (s, ss \cup \{\bot\}) \in B \\
& \quad \lor \\
& \quad (s, ss \cup \{\bot\}) \in B \lor (s, ss) \in B \\
& \quad \text{\{Property of sets and predicate calculus\}} \\
= & \left\{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \mid (s, ss \cup \{\bot\}) \in B \lor (s, ss) \in B \right\} \\
& \quad \text{\{Definition of bmh}_1 \text{\}} \\
= & \text{bmh}_1(B)
\end{align*}
\]
Lemma B.1.3 (\(\text{bmh}_2\)-idempotent)

\[ \text{bmh}_2 \circ \text{bmh}_2(B) = \text{bmh}_2(B) \]

Proof.

\[
\text{bmh}_2 \circ \text{bmh}_2(B) = \left\{ \begin{array}{l}
s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
(s, ss) \in \text{bmh}_2(B) \\
\land \\
((s, \{\bot\}) \in \text{bmh}_2(B) \iff (s, \emptyset) \in \text{bmh}_2(B))
\end{array} \right\}
\]

\[
= \left\{ \begin{array}{l}
s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
(s, ss) \in \left\{ \begin{array}{l}
s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
(s, ss) \in B \land ((s, \{\bot\}) \in B \iff (s, \emptyset) \in B)
\end{array} \right\} \\
\land \\
\iff
\left( \begin{array}{l}
(s, \{\bot\}) \in \left\{ \begin{array}{l}
s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
(s, ss) \in B \land ((s, \{\bot\}) \in B \iff (s, \emptyset) \in B)
\end{array} \right\} \\
\land \\
((s, \emptyset) \in B \land ((s, \{\bot\}) \in B \iff (s, \emptyset) \in B))
\end{array} \right) \\
\right\}
\]

\[
= \left\{ \begin{array}{l}
s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
(s, ss) \in B \land ((s, \{\bot\}) \in B \iff (s, \emptyset) \in B) \\
\land \\
((s, \emptyset) \in B \land ((s, \{\bot\}) \in B \iff (s, \emptyset) \in B))
\end{array} \right\}
\]

\[
= \left\{ \begin{array}{l}
s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
(s, ss) \in B \land ((s, \{\bot\}) \in B \iff (s, \emptyset) \in B) \\
\land \\
((s, \{\bot\}) \in B \land (s, \emptyset) \in B) \iff ((s, \emptyset) \in B \land (s, \{\bot\}) \in B))
\end{array} \right\}
\]

\[
= \left\{ \begin{array}{l}
s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
(s, ss) \in B \land ((s, \{\bot\}) \in B \iff (s, \emptyset) \in B) \\
\land \\
((s, \emptyset) \in B \land (s, \emptyset) \in B)
\end{array} \right\}
\]

\[
= \left\{ \begin{array}{l}
s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
(s, ss) \in B \land ((s, \{\bot\}) \in B \iff (s, \emptyset) \in B)
\end{array} \right\}
\]

\[
= \left\{ \begin{array}{l}
s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
(s, ss) \in B \land ((s, \{\bot\}) \in B \iff (s, \emptyset) \in B)
\end{array} \right\}
\]

\[
= \left\{ \begin{array}{l}
s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
(s, ss) \in B \land ((s, \{\bot\}) \in B \iff (s, \emptyset) \in B)
\end{array} \right\}
\]
\[ = \text{bmh}_2(B) \]

\[ \text{Lemma B.1.4 (bmh}_3\text{-idempotent)} \]

\[ \text{bmh}_3 \circ \text{bmh}_3(B) = B \]

Proof.

\[
\text{bmh}_3 \circ \text{bmh}_3(B) = \begin{cases} 
  s : \text{State}, ss : \mathbb{P} \text{State}_\perp 
  \mid (s, \emptyset) \in \text{bmh}_3(B) \lor \perp \notin ss) \land (s, ss) \in \text{bmh}_3(B) 
\end{cases} 
\]

\[
= \begin{cases} 
  s : \text{State}, ss : \mathbb{P} \text{State}_\perp 
  \mid (s, \emptyset) \in \{ s : \text{State}, ss : \mathbb{P} \text{State}_\perp \mid ((s, \emptyset) \in B \lor \perp \notin ss) \land (s, ss) \in B \} \lor \perp \notin ss) 
  \land (s, ss) \in \{ s : \text{State}, ss : \mathbb{P} \text{State}_\perp \mid ((s, \emptyset) \in B \lor \perp \notin ss) \land (s, ss) \in B \} 
\end{cases} 
\]

\[
= \begin{cases} 
  s : \text{State}, ss : \mathbb{P} \text{State}_\perp 
  \mid ((s, \emptyset) \in B \lor \perp \notin ss) \land (s, ss) \in B 
\end{cases} 
\]

\[
= \text{bmh}_3(B) 
\]

\[ \square \]
**Lemma B.1.5**

\[ \text{bmh}_0 \circ \text{bmh}_1(B) = \]
\[ \left\{ \begin{array}{l}
  s : \text{State}, ss : \mathbb{P} \text{State}_\perp \\
  \exists ss_0 \bullet ((s, ss_0) \in B \lor (s, ss_0 \cup \{ \perp \}) \in B) \land ss_0 \subseteq ss \land (\perp \in ss_0 \Leftrightarrow \perp \in ss)
\end{array} \right\} \]

**Proof.**

\[ \text{bmh}_0 \circ \text{bmh}_1(B) = \]
\[ \left\{ \begin{array}{l}
  s : \text{State}, ss : \mathbb{P} \text{State}_\perp \\
  \exists ss_0 \bullet ((s, ss_0) \in \text{bmh}_1(B) \land ss_0 \subseteq ss \land (\perp \in ss_0 \Leftrightarrow \perp \in ss))
\end{array} \right\} \]

\[ \left\{ \begin{array}{l}
  s : \text{State}, ss : \mathbb{P} \text{State}_\perp \\
  \exists ss_0 \bullet ((s, ss_0) \in \{ s : \text{State}, ss : \mathbb{P} \text{State}_\perp \mid (s, ss \cup \{ \perp \}) \in B \lor (s, ss) \in B \} \land ss_0 \subseteq ss \land (\perp \in ss_0 \Leftrightarrow \perp \in ss)
\end{array} \right\} \]

\[ \left\{ \begin{array}{l}
  s : \text{State}, ss : \mathbb{P} \text{State}_\perp \\
  \exists ss_0 \bullet ((s, ss_0 \cup \{ \perp \}) \in B \lor (s, ss_0) \in B) \land ss_0 \subseteq ss \land (\perp \in ss_0 \Leftrightarrow \perp \in ss)
\end{array} \right\} \]

\[ \left\{ \begin{array}{l}
  s : \text{State}, ss : \mathbb{P} \text{State}_\perp \\
  \exists ss_0 \bullet ((s, ss_0) \in B \land ss_0 \subseteq ss \land (\perp \in ss_0 \Leftrightarrow \perp \in ss)) \lor \\
  \exists ss_0 \bullet ((s, ss_0) \in B \land ss_0 \subseteq ss \land (\perp \in ss_0 \Leftrightarrow \perp \in ss))
\end{array} \right\} \]

\[ \left\{ \begin{array}{l}
  s : \text{State}, ss : \mathbb{P} \text{State}_\perp \\
  \exists ss_0 \bullet ((s, ss_0) \in B \lor (s, ss_0 \cup \{ \perp \}) \in B) \land ss_0 \subseteq ss \land (\perp \in ss_0 \Leftrightarrow \perp \in ss)
\end{array} \right\} \]

\[ \square \]

**Properties**

**Lemma B.1.6 (bmh\(_0\) \circ bmh\(_1\)-commutative)**

\[ \text{bmh}_0 \circ \text{bmh}_1(B) = \text{bmh}_1 \circ \text{bmh}_0(B) \]
Proof.

\begin{align*}
\text{bmh}_1 \circ \text{bmh}_0(B) & \quad \text{Definition of \text{bmh}_1} \\
= \{ \quad & s : \text{State}, ss : \mathbb{P} \text{State}_\bot \mid (s, ss \cup \{\bot\}) \in \text{bmh}_0(B) \lor (s, ss) \in \text{bmh}_0(B) \} \quad \text{Definition of \text{bmh}_0} \\
= \{ \quad & s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
& (s, ss \cup \{\bot\}) \in \left\{ \begin{array}{l}
\forall \exists ss_0 \cdot (s, ss_0) \in B \land ss_0 \subseteq ss \land (\bot \in ss_0 \iff \bot \in ss) \\
\end{array} \right\} \\
& \lor \\
& (s, ss) \in \left\{ \begin{array}{l}
\exists ss_0 \cdot (s, ss_0) \in B \land ss_0 \subseteq ss \land (\bot \in ss_0 \iff \bot \in ss) \\
\end{array} \right\} \\
& \quad \text{Property of sets} \\
= \{ \quad & s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
& \exists ss_0 \cdot ((s, ss_0) \in B \land ss_0 \subseteq (ss \cup \{\bot\}) \land (\bot \in ss_0 \iff \bot \in (ss \cup \{\bot\}))) \\
& \lor \\
& \exists ss_0 \cdot ((s, ss_0) \in B \land ss_0 \subseteq ss \land (\bot \in ss_0 \iff \bot \in ss)) \\
& \quad \text{Property of sets and predicate calculus} \\
= \{ \quad & s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
& \exists ss_0 \cdot ((s, ss_0) \in B \land ss_0 \subseteq (ss \cup \{\bot\}) \land \bot \in ss_0) \\
& \lor \\
& \exists ss_0 \cdot ((s, ss_0) \in B \land ss_0 \subseteq ss \land (\bot \in ss_0 \iff \bot \in ss)) \\
& \quad \text{Lemma [B.3.1]} \\
= \{ \quad & s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
& \exists ss_0 \cdot ((s, ss_0 \cup \{\bot\}) \in B \land ss_0 \subseteq ss \land (\bot \in ss_0 \iff \bot \in ss)) \\
& \lor \\
& \exists ss_0 \cdot ((s, ss_0) \in B \land ss_0 \subseteq ss \land (\bot \in ss_0 \iff \bot \in ss)) \\
& \quad \text{Predicate calculus} \\
= \{ \quad & s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
& \exists ss_0 \cdot ((s, ss_0 \cup \{\bot\}) \in B \lor (s, ss_0) \in B) \\
& \land ss_0 \subseteq ss \land (\bot \in ss_0 \iff \bot \in ss) \\
& \quad \text{Lemma [B.1.5]} \\
= \text{bmh}_0 \circ \text{bmh}_1(B) \\
\end{align*}
**bmh$_1$ and bmh$_2$**

**Lemma B.1.7**

\[ \text{bmh}_1 \circ \text{bmh}_2(B) = \left\{ s : \text{State}, ss : \mathcal{P} \text{State}\downarrow \mid ((s, \{\bot\}) \in B \iff (s, \emptyset) \in B) \land ((s, ss \cup \{\bot\}) \in B \lor (s, ss) \in B) \right\} \]

*Proof.*

\[ \text{bmh}_1 \circ \text{bmh}_2(B) \quad \{\text{Definition of \text{bmh}_1}\} \]

\[ = \left\{ s : \text{State}, ss : \mathcal{P} \text{State}\downarrow \mid (s, ss \cup \{\bot\}) \in \text{bmh}_2(B) \lor (s, ss) \in \text{bmh}_2(B) \right\} \quad \{\text{Definition of \text{bmh}_2}\} \]

\[ = \left\{ \begin{array}{l}
  s : \text{State}, ss : \mathcal{P} \text{State}\downarrow \\
  (s, ss \cup \{\bot\}) \in \left\{ s : \text{State}, ss : \mathcal{P} \text{State}\downarrow \\
  \quad \mid (s, ss) \in B \land ((s, \{\bot\}) \in B \iff (s, \emptyset) \in B) \right\} \\
  \lor \\
  (s, ss) \in \left\{ s : \text{State}, ss : \mathcal{P} \text{State}\downarrow \\
  \quad \mid (s, ss) \in B \land ((s, \{\bot\}) \in B \iff (s, \emptyset) \in B) \right\} \end{array} \right\} \quad \{\text{Property of sets}\} \]

\[ = \left\{ \begin{array}{l}
  s : \text{State}, ss : \mathcal{P} \text{State}\downarrow \\
  ((s, ss \cup \{\bot\}) \in B \land ((s, \{\bot\}) \in B \iff (s, \emptyset) \in B)) \\
  \lor \\
  ((s, ss) \in B \land ((s, \{\bot\}) \in B \iff (s, \emptyset) \in B)) \end{array} \right\} \quad \{\text{Predicate calculus}\} \]

\[ = \left\{ s : \text{State}, ss : \mathcal{P} \text{State}\downarrow \\
  \quad \mid ((s, \{\bot\}) \in B \iff (s, \emptyset) \in B) \land ((s, ss \cup \{\bot\}) \in B \lor (s, ss) \in B) \right\} \]

\[ \square \]

**Lemma B.1.8**

\[ \text{bmh}_2 \circ \text{bmh}_1(B) = \left\{ s : \text{State}, ss : \mathcal{P} \text{State}\downarrow \mid ((s, ss \cup \{\bot\}) \in B \lor (s, ss) \in B) \land ((s, \emptyset) \in B \Rightarrow (s, \{\bot\}) \in B) \right\} \]

*Proof.*

\[ \text{bmh}_2 \circ \text{bmh}_1(B) \quad \{\text{Definition of \text{bmh}_2}\} \]
\[
\begin{align*}
\{ s : \text{State}, \text{ss} : \mathbb{P} \text{State}_\bot \\
\mid (s, \text{ss}) &\in \text{bmh}_1(B) \land ((s, \{\bot\}) \in \text{bmh}_1(B) \iff (s, \emptyset) \in \text{bmh}_1(B)) \} \\
\text{Definition of bmh}_1
\end{align*}
\]

= \( \{ s : \text{State}, \text{ss} : \mathbb{P} \text{State}_\bot \\
\mid (s, \text{ss}) \in \{ s : \text{State}, \text{ss} : \mathbb{P} \text{State}_\bot \mid (s, \text{ss} \cup \{\bot\}) \in B \lor (s, \text{ss}) \in B \} \\
\land \\
(s, \{\bot\}) \in \{ s : \text{State}, \text{ss} : \mathbb{P} \text{State}_\bot \mid (s, \text{ss} \cup \{\bot\}) \in B \lor (s, \text{ss}) \in B \} \\
\iff \\
(s, \emptyset) \in \{ s : \text{State}, \text{ss} : \mathbb{P} \text{State}_\bot \mid (s, \text{ss} \cup \{\bot\}) \in B \lor (s, \text{ss}) \in B \} \} \}
\]

\{Property of sets\}

\[
\begin{align*}
\{ s : \text{State}, \text{ss} : \mathbb{P} \text{State}_\bot \\
\mid ((s, \text{ss} \cup \{\bot\}) \in B \lor (s, \text{ss}) \in B) \\
\land \\
((s, \{\bot\} \cup \{\bot\}) \in B \lor (s, \{\bot\}) \in B) \\
\iff \\
((s, \emptyset \cup \{\bot\}) \in B \lor (s, \emptyset) \in B) \} \\
\text{Property of sets and predicate calculus}
\end{align*}
\]

\[
\begin{align*}
\{ s : \text{State}, \text{ss} : \mathbb{P} \text{State}_\bot \\
\mid ((s, \text{ss} \cup \{\bot\}) \in B \lor (s, \text{ss}) \in B) \\
\land \\
((s, \{\bot\}) \in B \\
\iff \\
((s, \{\bot\}) \in B \lor (s, \emptyset) \in B) \} \\
\text{Predicate calculus}
\end{align*}
\]

\[
\begin{align*}
\{ s : \text{State}, \text{ss} : \mathbb{P} \text{State}_\bot \\
\mid ((s, \text{ss} \cup \{\bot\}) \in B \lor (s, \text{ss}) \in B) \\
\land \\
((s, \{\bot\}) \in B \lor (s, \emptyset) \in B) \} \\
\text{Lemma B.1.8}
\end{align*}
\]

It can be concluded from Lemma B.1.8 and Lemma B.1.7 that the functional application of \( \text{bmh}_1 \circ \text{bmh}_2 \) is stronger than that of \( \text{bmh}_2 \circ \text{bmh}_1 \). The order in which these two healthiness conditions are functionally composed is important, since they are not necessarily commutative. The following counter-example illustrates the issue for a relation that is not BMH2-healthy.

**Counter-example 4**

\[
\text{bmh}_2 \circ \text{bmh}_1(\{ s : \text{State}, \text{ss} : \mathbb{P} \text{State}_\bot \mid \text{ss} = \{\bot\} \}) \]

{Lemma B.1.8}

Revision: 704f887 (2014-02-04 11:14:10 +0000) 175
\[= \{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \mid ss = \{ \bot \} \lor ss = \emptyset \}\]

\[\text{bmh}_1 \circ \text{bmh}_2(\{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \mid ss = \{ \bot \} \})\quad \text{Lemma B.1.7}\]

\[= \emptyset\]

\textbf{bmh}_2 \text{ and } \textbf{bmh}_3

\textbf{Lemma B.1.9}

\[\text{bmh}_2 \circ \text{bmh}_3(B)\]

\[= \left\{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \right\}
\left\{ ((s, \emptyset) \in B \lor \bot \notin ss) \land (s, ss) \in B \land ((s, \emptyset) \in B \Rightarrow (s, \{ \bot \}) \in B) \right\}\]

\textit{Proof.}

\[\text{bmh}_2 \circ \text{bmh}_3(B)\quad \text{Definition of } \text{bmh}_2\]

\[= \left\{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \right\}
\left\{ ((s, \emptyset) \in B \lor \bot \notin ss) \land (s, ss) \in B \right\}
= \left\{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \right\}
\left\{ ((s, \emptyset) \in B \lor \bot \notin ss) \land (s, ss) \in B \right\}
\left\{ ((s, \emptyset) \in B \lor \bot \notin ss) \land (s, ss) \in B \right\}
\left\{ \text{Property of sets} \right\}\]

\[\text{bmh}_2 \circ \text{bmh}_3(B)\quad \text{Definition of } \text{bmh}_3\]

\[= \left\{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \right\}
\left\{ ((s, \emptyset) \in B \lor \bot \notin ss) \land (s, ss) \in B \right\}
\left\{ ((s, \emptyset) \in B \lor \bot \notin ss) \land (s, ss) \in B \right\}
\left\{ \text{Property of sets} \right\}\]

\[\text{bmh}_2 \circ \text{bmh}_3(B)\quad \text{Property of sets and predicate calculus}\]
\[
\begin{aligned}
&\{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
&\quad | ((s, \emptyset) \in B \lor \bot \notin ss) \land (s, ss) \in B) \\
&\quad \land \left( \left( ((s, \emptyset) \in B \land (s, \{ \bot \}) \in B) \right) \leftrightarrow \\
&\quad \left( (s, \emptyset) \in B \right) \right) \} \{ \text{Predicate calculus} \}
\end{aligned}
\]

\[
\begin{aligned}
&\{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
&\quad | ((s, \emptyset) \in B \lor \bot \notin ss) \land (s, ss) \in B \land ((s, \emptyset) \in B \Rightarrow (s, \{ \bot \}) \in B) \}
\end{aligned}
\]

\begin{proof}
\[
\begin{aligned}
&\{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
&\quad | ((s, \emptyset) \in \text{bmh}_2(B) \lor \bot \notin ss) \land (s, ss) \in \text{bmh}_2(B) \} \{ \text{Definition of bmh}_3 \}
\end{aligned}
\]

\[
\begin{aligned}
&\{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
&\quad | (s, \emptyset) \in \text{bmh}_2(B) \land (s, ss) \in \text{bmh}_2(B) \} \{ \text{Definition of bmh}_2(B) \}
\end{aligned}
\]

\[
\begin{aligned}
&\{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
&\quad | (s, \emptyset) \in \text{bmh}_2(B) \land (s, ss) \in \text{bmh}_2(B) \} \{ \text{Property of sets} \}
\end{aligned}
\]

\[
\begin{aligned}
&\{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
&\quad | (((s, \emptyset) \in B \land (s, \{ \bot \}) \in B \land (s, \{ \bot \}) \in B) \lor \bot \notin ss) \\
&\quad \land (s, ss) \in B \land ((s, \{ \bot \}) \in B \Rightarrow (s, \emptyset) \in B) \} \{ \text{Predicate calculus} \}
\end{aligned}
\]

\[
\begin{aligned}
&\text{Lemma B.1.10} \\
&\text{bmh}_3 \circ \text{bmh}_2(B) \\
&= \{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
&\quad | ((s, \emptyset) \in \text{bmh}_2(B) \land (s, ss) \in \text{bmh}_2(B) \} \{ \text{Definition of bmh}_3 \}
\end{aligned}
\]

The functions $bmh_2$ and $bmh_3$ are not in general commutative. The following counter-example illustrates the issue for a relation that is not BMH2-health.

**Counter-example 5**

$$bmh_2 \circ bmh_3(\{s : State, ss : \mathbb{P} State_\bot \mid ss = \{\bot\} \lor ss = \{s\}\})$$

$$= \{s : State, ss : \mathbb{P} State_\bot \mid ((s, \emptyset) \in B \lor \bot \notin ss) \land ((s, \bot) \in B \lor s \in ss) \land (s, ss \cup \{\bot\}) \in B \lor (s, \emptyset) \in B\}$$

{Lemma B.1.9}

$$bmh_3 \circ bmh_2(\{s : State, ss : \mathbb{P} State_\bot \mid ss = \{\bot\} \lor ss = \{s\}\})$$

$$= \emptyset$$

{Lemma B.1.10}

**bmh_1 and bmh_3**

**Lemma B.1.11**

$$bmh_3 \circ bmh_1(B)$$

$$= \{s : State, ss : \mathbb{P} State_\bot \mid ((s, \bot) \in B \lor (s, \emptyset) \in B \lor \bot \notin ss) \land ((s, ss \cup \{\bot\}) \in B \lor (s, ss) \in B)\}$$
Proof.

\[ \text{bmh}_3 \circ \text{bmh}_1(B) \]
\[ = \{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \mid ((s, \emptyset) \in \text{bmh}_1(B) \lor \bot \notin ss) \land (s, ss) \in \text{bmh}_1(B) \} \]
\[ \text{Definition of bmh}_3 \]
\[ = \left\{ \begin{array}{l}
   s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
   \left( (s, \emptyset) \in \{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \mid (s, ss \cup \{ \bot \}) \in B \lor (s, ss) \in B \} \lor \bot \notin ss \right) \\
   (s, ss) \in \{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \mid (s, ss \cup \{ \bot \}) \in B \lor (s, ss) \in B \} \\
\end{array} \right\} \]
\[ \text{Definition of bmh}_1 \]
\[ \left\{ \begin{array}{l}
   \text{Property of sets} \\
\end{array} \right\} \]

\[ \square \]

Lemma B.1.12

\[ \text{bmh}_1 \circ \text{bmh}_3(B) \]
\[ = \left\{ \begin{array}{l}
   s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
   \left( (s, \{ \bot \}) \in B \lor (s, \emptyset) \in B \lor \bot \notin ss \right) \\
   (s, ss \cup \{ \bot \}) \in B \lor (s, ss) \in B \\
\end{array} \right\} \]
\[ \text{Definition of bmh}_1 \]

Proof.

\[ \text{bmh}_1 \circ \text{bmh}_3(B) \]
\[ = \{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \mid (s, ss \cup \{ \bot \}) \in \text{bmh}_3(B) \lor (s, ss) \in \text{bmh}_3(B) \} \]
\[ \text{Definition of bmh}_3 \]
The functions $bmh_3$ and $bmh_1$ do not necessarily commute. The following counter-example shows this for a relation that is not $BMH3$-healthy. In fact, the functional application $bmh_3 \circ bmh_1$ is not suitable as the counter-example shows that we have a fixed point.

**Counter-example 6**

$bmh_3 \circ bmh_1(\{s : State, ss : \mathbb{P} \text{State}_\bot \mid ss = \{\bot, s\} \vee ss = \{\bot\}\})$

$= \{s : State, ss : \mathbb{P} \text{State}_\bot \mid ss = \{\bot, s\} \vee ss = \{\bot\}\}$

$bmh_1 \circ bmh_3(\{s : State, ss : \mathbb{P} \text{State}_\bot \mid ss = \{\bot, s\} \vee ss = \{\bot\}\})$

$= \emptyset$

$bmh_{0,1,3,2}$

Lemma B.1.13 ($bmh_{0,1,3,2}$-idempotent)

$bmh_{0,1,3,2} \circ bmh_{0,1,3,2}(B) = bmh_{0,1,3,2}(B)$
Proof.

\[ \text{bmh}_{0,1,3,2} \circ \text{bmh}_{0,1,3,2}(B) \]

\{Definition of \text{bmh}_{0,1,3,2}\}

\[ \begin{cases} \ s : \text{State}, \ ss : \mathbb{P} \text{State}_\bot \\ (s, \emptyset) \in \text{bmh}_{0,1,3,2}(B) \land (s, \{\bot\}) \in \text{bmh}_{0,1,3,2}(B) \end{cases} \]

\{Law [3.2.13] and Law [3.2.14]\}

\[ = \begin{cases} \ s : \text{State}, \ ss : \mathbb{P} \text{State}_\bot \\ (s, \emptyset) \in B \land (s, \{\bot\}) \in B \land (s, \{\bot\}) \in B \end{cases} \]

\{Predicate calculus and definition of \text{bmh}_{0,1,3,2}\}

\[ = \begin{cases} \ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\ (s, \emptyset) \in B \land (s, \{\bot\}) \in B \land (s, \{\bot\}) \in B \end{cases} \]

\{Variable renaming and property of sets\}
\[
\begin{align*}
\{ \text{Predicate calculus} \} &= \\
\{ \text{Predicate calculus: quantifier scope} \} &\text{ and } \\
\{ \text{Predicate calculus: absorption law} \}
\end{align*}
\]
\[
\begin{align*}
\{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
((s, \emptyset) \in B \land (s, \{ \bot \}) \in B) \\
\lor \\
\left( (s, \{ \bot \}) \notin B \land (s, \emptyset) \notin B \\
\land \\
\exists ss_1, ss_0 \bullet \left( (s, ss_1) \in B \land ss_1 \subseteq ss_0 \land \bot \notin ss_1 \land \bot \notin ss_0 \\
\land ss_0 \subseteq ss \land \bot \notin ss_0 \land \bot \notin ss \right) \right) \\
\} \\
\end{align*}
\]

{Predicate calculus}

\[
\begin{align*}
\{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
((s, \emptyset) \in B \land (s, \{ \bot \}) \in B) \\
\lor \\
\left( (s, \{ \bot \}) \notin B \land (s, \emptyset) \notin B \\
\land \\
\exists ss_1 \bullet \left( (s, ss_1) \in B \land ss_1 \subseteq ss \land \bot \notin ss_1 \land \bot \notin ss \\
\land \bot \notin ss_1 \land \bot \notin ss_0 \\
\land ss_0 \subseteq ss \land \bot \notin ss_0 \land \bot \notin ss \right) \right) \\
\} \\
\end{align*}
\]

{Definition of \( \text{bmh}_{0,1,3,2} \)}

\[
\begin{align*}
\text{bmh}_{0,1,2} \circ \text{bmh}_{0,1,3,2}(B) = \text{bmh}_{0,1,3,2}(B)
\end{align*}
\]

Lemma B.1.14

Proof.

\[
\begin{align*}
\text{bmh}_{0,1,2} \circ \text{bmh}_{0,1,3,2}(B) \\
= \{ \text{Definition of } \text{bmh}_{0,1,2} \}
\end{align*}
\]

\[
\begin{align*}
\{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
\exists ss_0 : \mathbb{P} \text{State}_\bot \bullet \\
((s, ss_0) \in \text{bmh}_{0,1,3,2}(B) \lor (s, ss_0 \cup \{ \bot \}) \in \text{bmh}_{0,1,3,2}(B)) \\
\land ((s, \{ \bot \}) \in \text{bmh}_{0,1,3,2}(B) \leftrightarrow (s, \emptyset) \in \text{bmh}_{0,1,3,2}(B)) \\
\land ss_0 \subseteq ss \land (\bot \in ss_0 \leftrightarrow \bot \in ss) \\
\} \\
\{ \text{Law B.2.13 and Law B.2.14 and predicate calculus} \}
\end{align*}
\]

\[
\begin{align*}
\{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
\exists ss_0 : \mathbb{P} \text{State}_\bot \bullet \\
((s, ss_0) \in \text{bmh}_{0,1,3,2}(B) \lor (s, ss_0 \cup \{ \bot \}) \in \text{bmh}_{0,1,3,2}(B)) \\
\land ss_0 \subseteq ss \land (\bot \in ss_0 \leftrightarrow \bot \in ss) \\
\} \\
\{ \text{Predicate calculus} \}
\end{align*}
\]
\begin{align*}
\begin{cases}
  s : \text{State}, ss : \mathbb{P} \text{State}_⊥ \\
  \exists ss_0 : \mathbb{P} \text{State}_⊥ \bullet \left( (s, ss_0) \in \text{bmh}_{0,1,3,2}(B) \wedge ss_0 \subseteq ss \wedge (\bot \in ss_0 \iff \bot \in ss) \right) \\
  \vee \\
  \exists ss_0 : \mathbb{P} \text{State}_⊥ \bullet \left( (s, ss_0 \cup \{ \bot \}) \in \text{bmh}_{0,1,3,2}(B) \wedge ss_0 \subseteq ss \wedge (\bot \in ss_0 \iff \bot \in ss) \right)
\end{cases}
\end{align*}
\small{
\{\text{Law B.2.11 and Law B.2.12}\}}

\begin{align*}
\begin{cases}
  s : \text{State}, ss : \mathbb{P} \text{State}_⊥ \\
  ((s, \emptyset) \in B \wedge (s, \{ \bot \}) \in B) \\
  \vee \\
  \left( (s, \{ \bot \}) \notin B \wedge (s, \emptyset) \notin B \wedge \exists ss_0 : \mathbb{P} \text{State}_⊥ \bullet (s, ss_0) \in B \wedge ss_0 \subseteq ss \wedge \bot \notin ss_0 \wedge \bot \notin ss \right) \\
  ( (s, \emptyset) \in B \wedge (s, \{ \bot \}) \in B) 
\end{cases}
\end{align*}
\small{
\{\text{Predicate calculus}\}}

\begin{align*}
\begin{cases}
  s : \text{State}, ss : \mathbb{P} \text{State}_⊥ \\
  ((s, \emptyset) \in B \wedge (s, \{ \bot \}) \in B) \\
  \vee \\
  \left( (s, \{ \bot \}) \notin B \wedge (s, \emptyset) \notin B \wedge \exists ss_0 : \mathbb{P} \text{State}_⊥ \bullet (s, ss_0) \in B \wedge ss_0 \subseteq ss \wedge \bot \notin ss_0 \wedge \bot \notin ss \right) \\
  \{\text{Definition of bmh}_{0,1,3,2}(B)\}
\end{cases}
\end{align*}

= \text{bmh}_{0,1,3,2}(B)

\square

\section*{B.2 Auxiliary lemmas}

\begin{lemma}
provided \(B_0\) and \(B_1\) are BMH0 and BMH1-healthy.

\((B_0 ; BM_\bot B_2) \sqcup_{BM_\bot} (B_1 ; BM_\bot B_2) \subseteq_{BM_\bot} \left( (B_0 \sqcup_{BM_\bot} B_1) ; BM_\bot B_2 \right)\)
\end{lemma}
Proof.

$$((B_0 \sqcup_{BM_\perp} B_1) \ ; \ BM_\perp B_2)$$
\[
\begin{align*}
\{ & s_0 : \textit{State}, ss_0 : \mathbb{P}\textit{State}_\bot \notag \\
& \quad \left( (s_0, \textit{State}_\bot) \in B_0 \\
& \quad \lor \\
& \quad (s_0, \{ s_1 : \textit{State} \mid (s_1, ss_0) \in B_2 \}) \in B_0 \right) \\
& \quad \land \\
& \quad \left( (s_0, \textit{State}_\bot) \in B_0 \\
& \quad \lor \\
& \quad (s_0, \{ s_1 : \textit{State} \mid (s_1, ss_0) \in B_2 \}) \in B_1 \right) \} \\
& \quad \land \\
& \quad \left( (s_0, \textit{State}_\bot) \in B_1 \\
& \quad \lor \\
& \quad (s_0, \{ s_1 : \textit{State} \mid (s_1, ss_0) \in B_2 \}) \in B_0 \right) \\
& \quad \lor \\
& \quad \left( (s_0, \textit{State}_\bot) \in B_1 \\
& \quad \lor \\
& \quad (s_0, \{ s_1 : \textit{State} \mid (s_1, ss_0) \in B_2 \}) \in B_1 \right) \} \\
& \quad \{ \text{Assumption: } B_0 \text{ and } B_1 \text{ are BMH1-healthy and BMH0-healthy} \} \\
\end{align*}
\]
\[
\begin{align*}
\exists_{BM} \quad s_0 : State, s_{s_0} : \mathbb{P} State \quad &
\begin{cases}
(s_0, State) \in B_0 \land (s_0, State) \in B_1 \\
\land \\
((s_0, State) \in B_0 \lor (s_0, \{s_1 : State \mid (s_1, s_{s_0}) \in B_2\}) \in B_0) \\
\land \\
((s_0, State) \in B_1 \land (s_0, State) \in B_1) \\
\lor \\
((s_0, \{s_1 : State \mid (s_1, s_{s_0}) \in B_2\}) \in B_0 \land (s_0, State) \in B_0)
\end{cases}
\end{align*}
\]

\begin{align*}
&\in \quad (s_0, State) \in B_0 \lor (s_0, \{s_1 : State \mid (s_1, s_{s_0}) \in B_2\}) \in B_0 \\
&\land \\
&\in \quad (s_0, State) \in B_1 \lor (s_0, \{s_1 : State \mid (s_1, s_{s_0}) \in B_2\}) \in B_1
\end{align*}

\{Property of sets\}

\begin{align*}
&= \quad (s_0 : State, s_{s_0} : \mathbb{P} State \quad &
\begin{cases}
((s_0, State) \in B_0 \lor (s_0, \{s_1 : State \mid (s_1, s_{s_0}) \in B_2\}) \in B_0) \\
\land \\
((s_0, State) \in B_1 \lor (s_0, \{s_1 : State \mid (s_1, s_{s_0}) \in B_2\}) \in B_1)
\end{cases}
\end{align*}

\{Assumption: \(B_0\) and \(B_1\) are BMH0-healthy and Law 4.57\}

\begin{align*}
&= \quad (B_0 \cup_{BM} B_2) \cap (B_1 \cup_{BM} B_2) \\
&= \quad (B_0 \cup_{BM} B_2) \cup_{BM} (B_1 \cup_{BM} B_2)
\end{align*}

\{Definition of \(\cup_{BM}\)\}

BMH0

**Law B.2.1** Provided \(B\) is BMH0-healthy.

\[
\left( \exists s_0 : State, s_{s_0}, ss_1 : \mathbb{P} State \quad & \bullet ((s_0, s_{s_0}) \in B \land ss_0 \subseteq ss_1 \land \bot \in ss_0 \land \bot \in ss_1) \right)
\]
\( \exists s_0 : State, ss_0, ss_1 : \mathbb{P} State_\bot \bullet (s_0, ss_1) \in B \land \bot \in ss_1 \)\\

Proof. (Implication)

\[
\left( \exists s_0 : State, ss_0, ss_1 : \mathbb{P} State_\bot \\
\bullet ((s_0, ss_0) \in B \land ss_0 \subseteq ss_1 \land \bot \in ss_0 \land \bot \in ss_1) \right)
\]

\{Assumption: B is BMH0-healthy\}

\[
= \left( \exists s_0 : State, ss_0, ss_1 : \mathbb{P} State_\bot \\
\bullet ((s_0, ss_0) \in B \land ss_0 \subseteq ss_1 \land \bot \in ss_0 \land \bot \in ss_1 \land (s_0, ss_1) \in B) \right)
\]

\{Propositional calculus\}

\[
\Rightarrow \exists s_0 : State, ss_1 : \mathbb{P} State_\bot \bullet (\bot \in ss_1 \land (s_0, ss_1) \in B)
\]

\(\square\)

Proof. (Reverse implication)

\[
\exists s_0 : State, ss_1 : \mathbb{P} State_\bot \bullet (\bot \in ss_1 \land (s_0, ss_1) \in B)
\]

\{Propositional calculus: introduce fresh variable\}

\[
= \left( \exists s_0 : State, ss_0, ss_1 : \mathbb{P} State_\bot \bullet \\
(\bot \in ss_1 \land (s_0, ss_0) \in B \land ss_0 = ss_1 \land (s_0, ss_1) \in B \land \bot \in ss_0) \right)
\]

\{Propositional calculus: weaken predicate\}

\[
\Rightarrow \left( \exists s_0 : State, ss_0, ss_1 : \mathbb{P} State_\bot \bullet \\
(\bot \in ss_1 \land (s_0, ss_1) \in B \land ss_0 \subseteq ss_1 \land (s_0, ss_0) \in B \land \bot \in ss_0) \right)
\]

\(\square\)

bmh\(_{0,1,2}\)

Law B.2.2

\[
\text{bmh}_{0,1,2}(B)
\]

=
Proof.

\[ b \mathbf{m} \mathbf{h}_{0,1,2}(B) = \left\{ \begin{array}{l}
\exists ss_0 \bullet ((s, \downarrow) \in B \lor (s, ss_0 \cup \downarrow) \in B) \\
\land ((s, \{\downarrow\}) \in B \iff (s, \emptyset) \in B) \\
\land ss_0 \subseteq ss \land (\downarrow \in ss_0 \iff \downarrow \in ss)
\end{array} \right\} \]  

\{ Definition of \( \mathbf{b} \mathbf{m} \mathbf{h}_{0,1,2} \} \]

\[ \left\{ \begin{array}{l}
\left( \exists ss_0 \bullet (s, ss_0) \in B \\
\land ss_0 \subseteq ss \land \downarrow \in ss_0 \land \downarrow \in ss
\right) \\
\lor \\
\left( \exists ss_0 \bullet (s, ss_0) \in B \\
\land ss_0 \subseteq ss \land \downarrow \notin ss_0 \land \downarrow \notin ss
\right)
\end{array} \right\} \]  

\{ Predicate calculus \}

\[ \left( \exists ss_0 \bullet (s, ss_0 \cup \{\downarrow\}) \in B \\
\land ss_0 \subseteq ss \land (\downarrow \in ss_0 \iff \downarrow \in ss) \right) \]  

\{ Lemma \( B.3.1 \) \}
\[
\begin{align*}
\forall s : \text{State}, \forall ss : \mathbb{P} \text{State}_{\bot} & \quad ((s, \{\bot\}) \in B \iff (s, \emptyset) \in B) \\
& \quad \land \quad \left( \exists ss_0 \bullet (s, ss_0) \in B \right) \\
& \quad \land \quad (ss_0 \subseteq ss \land \bot \in ss_0 \land \bot \in ss) \\
& \quad \lor \quad (\exists ss_0 \bullet (s, ss_0) \in B) \\
& \quad \land \quad (ss_0 \subseteq ss \land \bot \notin ss_0 \land \bot \notin ss) \\
& \quad \lor \quad (\exists ss_0 \bullet (s, ss_0) \in B) \\
& \quad \land \quad (ss_0 \subseteq (ss \cup \{\bot\}) \land \bot \in ss_0)
\end{align*}
\]

\{Property of sets\}

\[
\begin{align*}
\forall s : \text{State}, \forall ss : \mathbb{P} \text{State}_{\bot} & \quad ((s, \{\bot\}) \in B \iff (s, \emptyset) \in B) \\
& \quad \land \quad \left( \exists ss_0 \bullet (s, ss_0) \in B \right) \\
& \quad \land \quad (ss_0 \subseteq (ss \cup \{\bot\}) \land \bot \in ss_0 \land \bot \in ss) \\
& \quad \lor \quad (\exists ss_0 \bullet (s, ss_0) \in B) \\
& \quad \land \quad (ss_0 \subseteq ss \land \bot \notin ss_0 \land \bot \notin ss) \\
& \quad \lor \quad (\exists ss_0 \bullet (s, ss_0) \in B) \\
& \quad \land \quad (ss_0 \subseteq (ss \cup \{\bot\}) \land \bot \in ss_0)
\end{align*}
\]

\{Predicate calculus: absorption law\}

\[
\begin{align*}
\forall s : \text{State}, \forall ss : \mathbb{P} \text{State}_{\bot} & \quad ((s, \{\bot\}) \in B \iff (s, \emptyset) \in B) \\
& \quad \land \quad \left( \exists ss_0 \bullet (s, ss_0) \in B \right) \\
& \quad \land \quad (ss_0 \subseteq ss \land \bot \notin ss_0 \land \bot \notin ss) \\
& \quad \lor \quad (\exists ss_0 \bullet (s, ss_0) \in B) \\
& \quad \land \quad (ss_0 \subseteq (ss \cup \{\bot\}) \land \bot \in ss_0)
\end{align*}
\]

\{Property of sets\}
\[
\begin{align*}
\text{Introduce fresh variable} \\
\{\text{Lemma B.3.2}\} \\
\{\text{One-point rule}\} \\
\{\text{Type: } \bot \notin ss_0, t\}
\end{align*}
\]
\[
\begin{align*}
\text{s : State, } ss & : \mathbb{P} \text{State} \perp \\
\text{=} & \left\{ \\
((s, \{\perp\}) \in B \iff (s, \emptyset) \in B) \\
\land \\
(\exists ss_0 : \mathbb{P} \text{State} \bullet (s, ss_0) \in B \land ss_0 \subseteq ss \land \perp \notin ss) \\
\lor \\
(\exists t : \mathbb{P} \text{State} \bullet (s, t \cup \{\perp\}) \in B \land t \subseteq ss) \\
\right\} \quad \text{(Variable renaming and substitution)} \\
\text{s : State, } ss & : \mathbb{P} \text{State} \perp \\
\text{=} & \left\{ \\
((s, \{\perp\}) \in B \iff (s, \emptyset) \in B) \\
\land \\
(\exists ss_0 : \mathbb{P} \text{State} \bullet (s, ss_0) \in B \land ss_0 \subseteq ss \land \perp \notin ss) \\
\lor \\
(\exists t : \mathbb{P} \text{State} \bullet (s, t \cup \{\perp\}) \in B \land t \subseteq ss) \\
\right\} \quad \text{(Predicate calculus)} \\
\text{s : State, } ss & : \mathbb{P} \text{State} \perp \\
\text{=} & \left\{ \\
((s, \{\perp\}) \in B \land (s, \emptyset) \in B) \\
\lor \\
((s, \{\perp\}) \notin B \land (s, \emptyset) \notin B) \\
\land \\
(\exists ss_0 : \mathbb{P} \text{State} \bullet (s, ss_0) \in B \land ss_0 \subseteq ss \land \perp \notin ss) \\
\lor \\
(\exists t : \mathbb{P} \text{State} \bullet (s, t \cup \{\perp\}) \in B \land t \subseteq ss) \\
\right\} \quad \text{(Instatiation: consider case where } t = \emptyset) \\
\text{s : State, } ss & : \mathbb{P} \text{State} \perp \\
\text{=} & \left\{ \\
((s, \{\perp\}) \in B \land (s, \emptyset) \in B) \\
\lor \\
((s, \{\perp\}) \notin B \land (s, \emptyset) \notin B) \\
\land \\
(\exists ss_0 : \mathbb{P} \text{State} \bullet (s, ss_0) \in B \land ss_0 \subseteq ss \land \perp \notin ss) \\
\lor \\
(\exists t : \mathbb{P} \text{State} \bullet (s, t \cup \{\perp\}) \in B \land t \subseteq ss) \\
\lor \\
(s, \emptyset) \in B \\
\right\} \quad \text{(Predicate calculus: absorption law and distribution)}
\end{align*}
\]
\[
\begin{align*}
\text{Law B.2.3} \\
(s, ss) \in \text{bmh}_{0,1,2}(B) &= \\
\text{Proof.} \\
(s, ss) \in \text{bmh}_{0,1,2}(B) &\quad \{\text{Definition of bmh}_{0,1,2}(B)\}
\end{align*}
\]
\[
(s, ss) \in \left\{ \begin{array}{l}
s : \text{State}, ss : \mathbb{P} \text{State}_1 \\
\exists ss_0 \bullet ((s, ss_0) \in B \lor (s, ss_0 \cup \{\bot\}) \in B) \\
\land ((s, \{\bot\}) \in B \leftrightarrow (s, \emptyset) \in B) \\
\land ss_0 \subseteq ss \land (\bot \in ss_0 \leftrightarrow \bot \in ss)
\end{array} \right\}
\]  
\{\text{Property of sets}\}

\[
= \left( \begin{array}{l}
((s, \{\bot\}) \in B \leftrightarrow (s, \emptyset) \in B) \\
\land \exists ss_0 \bullet ((s, ss_0) \in B \lor (s, ss_0 \cup \{\bot\}) \in B) \\
\land ss_0 \subseteq ss \land (\bot \in ss_0 \leftrightarrow \bot \in ss)
\end{array} \right) \quad \square
\]

\text{Law B.2.4}

\[
\exists ss_1 \bullet (s, ss_1) \in \text{bmh}_{0,1,2}(B) \land ss_1 \subseteq ss \land (\bot \in ss_1 \leftrightarrow \bot \in ss)
\]

\text{Proof.}

\[
\exists ss_1 \bullet (s, ss_1) \in \text{bmh}_{0,1,2}(B) \land ss_1 \subseteq ss \land (\bot \in ss_1 \leftrightarrow \bot \in ss)
\]

\[
= \exists ss_1 \bullet \left( \begin{array}{l}
(s, ss_1) \in \left\{ \begin{array}{l}
s : \text{State}, ss : \mathbb{P} \text{State}_1 \\
\exists ss_0 \bullet ((s, ss_0) \in B \lor (s, ss_0 \cup \{\bot\}) \in B) \\
\land ((s, \{\bot\}) \in B \leftrightarrow (s, \emptyset) \in B) \\
\land ss_0 \subseteq ss \land (\bot \in ss_0 \leftrightarrow \bot \in ss)
\end{array} \right\}
\right)
\{\text{Definition of \text{bmh}_{0,1,2}}\}
\]

\[
\land ss_1 \subseteq ss \land (\bot \in ss_1 \leftrightarrow \bot \in ss) \land ss_1 \subseteq ss \land (\bot \in ss_1 \leftrightarrow \bot \in ss)
\]  
\{\text{Property of sets}\}

\[
= \exists ss_1 \bullet \left( \begin{array}{l}
\exists ss_0 \bullet ((s, ss_0) \in B \lor (s, ss_0 \cup \{\bot\}) \in B) \\
\land ((s, \{\bot\}) \in B \leftrightarrow (s, \emptyset) \in B) \\
\land ss_0 \subseteq ss \land (\bot \in ss_0 \leftrightarrow \bot \in ss_1)
\end{array} \right)
\{\text{Predicate calculus: quantifier scope}\}
\]
\[
\left\{ \begin{array}{l}
((s, \{\bot\}) \in B \iff (s, \emptyset) \in B) \\
\wedge \\
\exists ss_1 \bullet \left( \begin{array}{l}
\exists ss_0 \bullet ((s, ss_0) \in B \lor (s, ss_0 \cup \{\bot\}) \in B) \\
\wedge ss_0 \subseteq ss_1 \wedge (\bot \in ss_0 \iff \bot \in ss_1) \\
\wedge ss_1 \subseteq ss \wedge (\bot \in ss_1 \iff \bot \in ss)
\end{array} \right)
\end{array} \right. \\
\{\text{Predicate calculus}\}
\]

\[
\left\{ \begin{array}{l}
((s, \{\bot\}) \in B \iff (s, \emptyset) \in B) \\
\wedge \\
\exists ss_0, ss_1 \bullet \left( \begin{array}{l}
(s, ss_0) \in B \\
\wedge ss_0 \subseteq ss_1 \wedge (\bot \in ss_0 \iff \bot \in ss_1) \\
\wedge ss_1 \subseteq ss \wedge (\bot \in ss_1 \iff \bot \in ss)
\end{array} \right) \\
\lor \\
\exists ss_0 \bullet ((s, ss_0 \cup \{\bot\}) \in B \land ss_0 \subseteq ss \land (\bot \in ss_0 \iff \bot \in ss))
\end{array} \right. \\
\{\text{Predicate calculus}\}
\]

\[
\left\{ \begin{array}{l}
((s, \{\bot\}) \in B \iff (s, \emptyset) \in B) \\
\wedge \\
\exists ss_0 \bullet ((s, ss_0) \in B \land ss_0 \subseteq ss \land (\bot \in ss_0 \iff \bot \in ss)) \\
\lor \\
\exists ss_0 \bullet ((s, ss_0 \cup \{\bot\}) \in B \land ss_0 \subseteq ss \land (\bot \in ss_0 \iff \bot \in ss))
\end{array} \right. \\
\{\text{Predicate calculus}\}
\]

\[
\left\{ \begin{array}{l}
((s, \{\bot\}) \in B \iff (s, \emptyset) \in B) \\
\wedge \\
\exists ss_0 \bullet ((s, ss_0) \in B \lor (s, ss_0 \cup \{\bot\}) \in B) \land ss_0 \subseteq ss \land (\bot \in ss_0 \iff \bot \in ss)
\end{array} \right. \\
\{\text{Predicate calculus}\}
\]

\[
(s, \emptyset) \in \text{bmh}_{0,1,2}(B) = (s, \emptyset) \in B \land (s, \{\bot\}) \in B
\]

\textbf{Proof.}

\[
(s, \emptyset) \in \text{bmh}_{0,1,2}(B) \quad \{\text{Definition of bmh}_{0,1,2}\}
\]
\[ (s, \emptyset) \in \begin{cases} \exists s_{s_0} \cdot ((s, s_{s_0}) \in B \lor (s, s_{s_0} \cup \{\bot\}) \in B) \\
\land ((s, \{\bot\}) \in B \iff (s, \emptyset) \in B) \\
\land s_{s_0} \subseteq s \land (s \in s_{s_0} \iff s \in ss) \end{cases} \]

\{\text{Property of sets}\}

\[ = \begin{cases} \exists s_{s_0} \cdot ((s, s_{s_0}) \in B \lor (s, s_{s_0} \cup \{\bot\}) \in B) \\
\land ((s, \{\bot\}) \in B \iff (s, \emptyset) \in B) \\
\land s_{s_0} \subseteq \emptyset \land (\bot \in s_{s_0} \iff \bot \in \emptyset) \end{cases} \]

\{\text{Predicate calculus}\}

\[ = ((s, \emptyset) \in B \lor (s, \emptyset \cup \{\bot\}) \in B) \land ((s, \{\bot\}) \in B \iff (s, \emptyset) \in B) \]

\{\text{Property of sets and predicate calculus}\}

\[ = (s, \{\bot\}) \in B \land (s, \emptyset) \in B \]

\[ \square \]

**Law B.2.6**

\( (s, \{\bot\}) \in \text{bmh}_{0,1,2}(B) = (s, \emptyset) \in B \land (s, \{\bot\}) \in B \)

**Proof.**

\( (s, \{\bot\}) \in \text{bmh}_{0,1,2}(B) \)

\{Definition of \text{bmh}_{0,1,2}\}

\[ = (s, \{\bot\}) \in \begin{cases} \exists s_{s_0} \cdot ((s, s_{s_0}) \in B \lor (s, s_{s_0} \cup \{\bot\}) \in B) \\
\land ((s, \{\bot\}) \in B \iff (s, \emptyset) \in B) \\
\land s_{s_0} \subseteq s \land (s \in s_{s_0} \iff s \in ss) \end{cases} \]

\{\text{Property of sets}\}

\[ = \begin{cases} \exists s_{s_0} \cdot ((s, s_{s_0}) \in B \lor (s, s_{s_0} \cup \{\bot\}) \in B) \\
\land ((s, \{\bot\}) \in B \iff (s, \emptyset) \in B) \\
\land s_{s_0} \subseteq \{\bot\} \land (\bot \in s_{s_0} \iff \bot \in \{\bot\}) \end{cases} \]

\{\text{Predicate calculus}\}

\[ = \begin{cases} \exists s_{s_0} \cdot ((s, s_{s_0}) \in B \lor (s, s_{s_0} \cup \{\bot\}) \in B) \\
\land ((s, \{\bot\}) \in B \iff (s, \emptyset) \in B) \\
\land s_{s_0} \subseteq \{\bot\} \land \bot \in s_{s_0} \end{cases} \]

\{Case analysis on \text{s}_{s_0} and one-point rule\}
\[(s, \{\bot\}) \in B \lor (s, \{\bot\} \cup \{\bot\}) \in B) \land ((s, \{\bot\}) \in B \not\iff (s, \emptyset) \in B)\]

\{Property of sets and predicate calculus\}

\[= (s, \{\bot\}) \in B \land (s, \emptyset) \in B\]

\[\square\]

**Law B.2.7**

\[B_1 \subseteq B_0\]

\[\iff \forall s : \text{State}, ss : \mathbb{P} \text{State} : (s, ss) \in B_1 \Rightarrow (s, ss) \in B_0\]

\[\land (s, ss \cup \{\bot\}) \in B_1 \Rightarrow (s, ss \cup \{\bot\}) \in B_0\]

**Proof.**

\[B_1 \subseteq B_0\]

\[\iff \forall s : \text{State}, ss : \mathbb{P} \text{State} : (s, ss) \in B_1 \Rightarrow (s, ss) \in B_0\]

\{Definition of subset inclusion\}

\[\iff \forall s : \text{State}, ss : \mathbb{P} \text{State} : ((s, ss) \in B_1 \Rightarrow (s, ss) \in B_0) \land (\bot \in ss \lor \bot \notin ss)\]

\{Predicate calculus\}

\[\iff \forall s : \text{State}, ss : \mathbb{P} \text{State} : \left(\exists t : \mathbb{P} \text{State} \bullet \bot \in ss \land t = ss \setminus \{\bot\} \Rightarrow ((s, ss) \in B_1 \Rightarrow (s, ss) \in B_0)\right)\]

\{Introduce fresh variable\}

\[\iff \left(\forall s : \text{State}, ss : \mathbb{P} \text{State} \bullet \left(\exists t : \mathbb{P} \text{State} \bullet \bot \in ss \land t = ss \setminus \{\bot\} \Rightarrow ((s, ss) \in B_1 \Rightarrow (s, ss) \in B_0)\right)\right)\]

\{Lemma B.3.2\}

Revision: 704f887 (2014-02-04 11:14:10 +0000) 197
∀ s : State, ss : \mathbb{P} State⊥ •
\left( (\exists t : \mathbb{P} State⊥ • \bot \notin t \land ss = t \cup \{\bot\} ) \Rightarrow \right.
\left( (s, ss) \in B_1 \Rightarrow (s, ss) \in B_0 \right)
\land
\left( \bot \notin ss \Rightarrow ((s, ss) \in B_1 \Rightarrow (s, ss) \in B_0) \right)
\left. \right) \{\text{Predicate calculus: quantifier scope}\}

\implies \left( (\bot \notin t \land ss = t \cup \{\bot\} ) \Rightarrow ((s, ss) \in B_1 \Rightarrow (s, ss) \in B_0) \right)
\land
\left( \bot \notin ss \Rightarrow ((s, ss) \in B_1 \Rightarrow (s, ss) \in B_0) \right) \{\text{Predicate calculus}\}

\implies \left( (\bot \notin t \Rightarrow (ss = t \cup \{\bot\} ) \Rightarrow ((s, ss) \in B_1 \Rightarrow (s, ss) \in B_0) \right)
\land
\left( \bot \notin ss \Rightarrow ((s, t) \in B_1 \Rightarrow (s, t) \in B_0) \right) \{\text{Variable renaming}\}

\implies \left( (\bot \notin t \Rightarrow \forall ss : \mathbb{P} State⊥ • \Rightarrow \right.
\left( (ss = t \cup \{\bot\} ) \Rightarrow ((s, ss) \in B_1 \Rightarrow (s, ss) \in B_0) \right)
\land
\left( \bot \notin ss \Rightarrow ((s, t) \in B_1 \Rightarrow (s, t) \in B_0) \right)
\left. \right) \{\text{Predicate calculus: quantifier scope}\}

\implies \left( (\bot \notin t \Rightarrow ((s, t \cup \{\bot\} ) \in B_1 \Rightarrow (s, t \cup \{\bot\} ) \in B_0) \right)
\land
\left( (\bot \notin t \Rightarrow ((s, t) \in B_1 \Rightarrow (s, t) \in B_0) \right) \{\text{Predicate calculus}\}

\{\text{Predicate calculus}\}

\{\text{Predicate calculus}\}

\{\text{Predicate calculus}\}

\{\text{Predicate calculus}\}

\{\text{Predicate calculus}\}
\[\forall s : \text{State}, t : \mathbb{P} \text{State} \bullet \left( (s, t \cup \{\perp\}) \in B_1 \Rightarrow (s, t \cup \{\perp\}) \in B_0 \right) \]

\[\text{bmh}_{0,1,3}\]

**Law B.2.8**

\[\text{bmh}_0 \circ \text{bmh}_1 \circ \text{bmh}_3(B) = \]

\[
\begin{cases}
  s : \text{State}, ss : \mathbb{P} \text{State}_\perp \\
  \exists ss_0 \bullet \left( (s, ss_0) \in \text{bmh}_3(B) \lor (s, ss_0 \cup \{\perp\}) \in \text{bmh}_3(B) \right) \\
  \land ss_0 \subseteq ss \land (\perp \in ss_0 \Leftrightarrow \perp \in ss)
\end{cases}
\]

**Proof.**

\[\text{bmh}_0 \circ \text{bmh}_1 \circ \text{bmh}_3(B) \quad \{\text{Definition of } \text{bmh}_0 \circ \text{bmh}_1\} \]

\[
\begin{cases}
  s : \text{State}, ss : \mathbb{P} \text{State}_\perp \\
  \exists ss_0 \bullet \left( (s, ss_0) \in \text{bmh}_3(B) \lor (s, ss_0 \cup \{\perp\}) \in \text{bmh}_3(B) \right) \\
  \land ss_0 \subseteq ss \land (\perp \in ss_0 \Leftrightarrow \perp \in ss)
\end{cases}
\]

\[
\begin{cases}
  s : \text{State}, ss : \mathbb{P} \text{State}_\perp \\
  \exists ss_0 \bullet \left( (s, ss_0) \in \text{bmh}_3(B) \lor (s, ss_0 \cup \{\perp\}) \in \text{bmh}_3(B) \right) \\
  \land ss_0 \subseteq ss \land (\perp \in ss_0 \Leftrightarrow \perp \in ss)
\end{cases}
\]

\[
\begin{cases}
  s : \text{State}, ss : \mathbb{P} \text{State}_\perp \\
  \exists ss_0 \bullet \left( (s, ss_0) \in \text{bmh}_3(B) \lor (s, ss_0 \cup \{\perp\}) \in \text{bmh}_3(B) \right) \\
  \land ss_0 \subseteq ss \land (\perp \in ss_0 \Leftrightarrow \perp \in ss)
\end{cases}
\]

\[
\begin{cases}
  s : \text{State}, ss : \mathbb{P} \text{State}_\perp \\
  \exists ss_0 \bullet \left( (s, ss_0) \in \text{bmh}_3(B) \lor (s, ss_0 \cup \{\perp\}) \in \text{bmh}_3(B) \right) \\
  \land ss_0 \subseteq ss \land (\perp \in ss_0 \Leftrightarrow \perp \in ss)
\end{cases}
\]

\[
\begin{cases}
  s : \text{State}, ss : \mathbb{P} \text{State}_\perp \\
  \exists ss_0 \bullet \left( (s, ss_0) \in \text{bmh}_3(B) \lor (s, ss_0 \cup \{\perp\}) \in \text{bmh}_3(B) \right) \\
  \land ss_0 \subseteq ss \land (\perp \in ss_0 \Leftrightarrow \perp \in ss)
\end{cases}
\]

\[
\begin{cases}
  s : \text{State}, ss : \mathbb{P} \text{State}_\perp \\
  \exists ss_0 \bullet \left( (s, ss_0) \in \text{bmh}_3(B) \lor (s, ss_0 \cup \{\perp\}) \in \text{bmh}_3(B) \right) \\
  \land ss_0 \subseteq ss \land (\perp \in ss_0 \Leftrightarrow \perp \in ss)
\end{cases}
\]

\[
\begin{cases}
  s : \text{State}, ss : \mathbb{P} \text{State}_\perp \\
  \exists ss_0 \bullet \left( (s, ss_0) \in \text{bmh}_3(B) \lor (s, ss_0 \cup \{\perp\}) \in \text{bmh}_3(B) \right) \\
  \land ss_0 \subseteq ss \land (\perp \in ss_0 \Leftrightarrow \perp \in ss)
\end{cases}
\]

\[
\begin{cases}
  s : \text{State}, ss : \mathbb{P} \text{State}_\perp \\
  \exists ss_0 \bullet \left( (s, ss_0) \in \text{bmh}_3(B) \lor (s, ss_0 \cup \{\perp\}) \in \text{bmh}_3(B) \right) \\
  \land ss_0 \subseteq ss \land (\perp \in ss_0 \Leftrightarrow \perp \in ss)
\end{cases}
\]
\[
s : \text{State}, \ ss : \mathcal{P} \ \text{State}_\bot \\
= \left\{ \begin{array}{l}
\exists s_{s0} \cdot (((s, \emptyset) \in B \lor \bot \notin s_{s0}) \land (s, s_{s0}) \in B) \\
\land \ s_{s0} \subseteq ss \land (\bot \in s_{s0} \Leftrightarrow \bot \in ss)
\end{array} \right\}
\]

\{Property of sets and predicate calculus\}

\[
s : \text{State}, \ ss : \mathcal{P} \ \text{State}_\bot \\
= \left\{ \begin{array}{l}
\exists s_{s0} \cdot (((s, \emptyset) \in B \lor \bot \notin s_{s0}) \land (s, s_{s0}) \in B) \\
\land \ s_{s0} \subseteq ss \land (\bot \in s_{s0} \Leftrightarrow \bot \in ss)
\end{array} \right\}
\]

\{Predicate calculus\}

\[
s : \text{State}, \ ss : \mathcal{P} \ \text{State}_\bot \\
= \left\{ \begin{array}{l}
\exists s_{s0} \cdot (((s, \emptyset) \in B \lor \bot \notin s_{s0}) \land (s, s_{s0}) \in B) \\
\land \ s_{s0} \subseteq ss \land (\bot \in s_{s0} \Leftrightarrow \bot \in ss)
\end{array} \right\}
\]

\{Predicate calculus\}

\[
s : \text{State}, \ ss : \mathcal{P} \ \text{State}_\bot \\
= \left\{ \begin{array}{l}
\exists s_{s0} \cdot (((s, s_{s0}) \in B \lor (s, s_{s0} \cup \{\bot\}) \in B) \\
\land \ s_{s0} \subseteq ss \land (\bot \in s_{s0} \Leftrightarrow \bot \in ss)
\end{array} \right\}
\]

\{Predicate calculus\}

\[
s : \text{State}, \ ss : \mathcal{P} \ \text{State}_\bot \\
= \left\{ \begin{array}{l}
\exists s_{s0} \cdot (((s, s_{s0}) \in B \lor (s, s_{s0} \cup \{\bot\}) \in B) \\
\land \ s_{s0} \subseteq ss \land (\bot \in s_{s0} \Leftrightarrow \bot \in ss)
\end{array} \right\}
\]

\{Predicate calculus\}

\[
s : \text{State}, \ ss : \mathcal{P} \ \text{State}_\bot \\
= \left\{ \begin{array}{l}
\exists s_{s0} \cdot (((s, s_{s0}) \in B \lor (s, s_{s0} \cup \{\bot\}) \in B) \\
\land \ s_{s0} \subseteq ss \land (\bot \in s_{s0} \Leftrightarrow \bot \in ss)
\end{array} \right\}
\]

\{Predicate calculus\}

\[
s : \text{State}, \ ss : \mathcal{P} \ \text{State}_\bot \\
= \left\{ \begin{array}{l}
\exists s_{s0} \cdot (((s, s_{s0}) \in B \lor (s, s_{s0} \cup \{\bot\}) \in B) \\
\land \ s_{s0} \subseteq ss \land (\bot \in s_{s0} \Leftrightarrow \bot \in ss)
\end{array} \right\}
\]

\{Predicate calculus\}
Law B.2.9

\[
\exists s_{s_0} \bullet \left( ((s, s_{s_0}) \in B \lor (s, s_{s_0} \cup \{\bot\}) \in B) \land s_{s_0} \subseteq ss \land (\bot \in s_{s_0} \iff \bot \in ss) \right)
\]

= \[
\exists s_{s_0} \bullet \left( ((s, s_{s_0}) \in B \lor (s, s_{s_0} \cup \{\bot\}) \in B) \land s_{s_0} \subseteq ss \land (\bot \in s_{s_0} \iff \bot \in ss) \right) \lor (s, \{\bot\}) \in B
\]

Proof.

\[
\exists s_{s_0} \bullet \left( ((s, s_{s_0}) \in B \lor (s, s_{s_0} \cup \{\bot\}) \in B) \land s_{s_0} \subseteq ss \land (\bot \in s_{s_0} \iff \bot \in ss) \right) \quad \text{\{Predicate calculus\}}
\]

= \[
\exists s_{s_0} \bullet (s, s_{s_0}) \in B \land s_{s_0} \subseteq ss \land (\bot \in s_{s_0} \iff \bot \in ss)
\lor \exists s_{s_0} \bullet (s, s_{s_0} \cup \{\bot\}) \in B \land s_{s_0} \subseteq ss \land (\bot \in s_{s_0} \iff \bot \in ss)
\]

\{Instantiation of existential quantification for \(s_{s_0} = \{\bot\}\) and \(s_{s_0} = \emptyset\}\]

= \[
\exists s_{s_0} \bullet (s, s_{s_0}) \in B \land s_{s_0} \subseteq ss \land (\bot \in s_{s_0} \iff \bot \in ss)
\lor \exists s_{s_0} \bullet (s, s_{s_0} \cup \{\bot\}) \in B \land s_{s_0} \subseteq ss \land (\bot \in s_{s_0} \iff \bot \in ss)
\lor ((s, \emptyset \cup \{\bot\}) \in B \land \emptyset \subseteq ss \land (\bot \in \emptyset \iff \bot \in ss)
\]

\{Property of sets\}

= \[
\exists s_{s_0} \bullet (s, s_{s_0}) \in B \land s_{s_0} \subseteq ss \land (\bot \in s_{s_0} \iff \bot \in ss)
\lor \exists s_{s_0} \bullet (s, s_{s_0} \cup \{\bot\}) \in B \land s_{s_0} \subseteq ss \land (\bot \in s_{s_0} \iff \bot \in ss)
\lor ((s, \{\bot\}) \in B \land \{\bot\} \subseteq ss \land \bot \in ss)
\lor ((s, \{\bot\}) \in B \land \bot \notin ss)
\]

\{Lemma [B.3.3] and predicate calculus\}
\[
\exists s_{s0} \cdot (s, s_{s0}) \in B \land s_{s0} \subseteq ss \land (∇ \in s_{s0} \Leftrightarrow ∇ \in ss)
\]

\[
\lor
\exists s_{s0} \cdot (s, s_{s0} \cup \{∇\}) \in B \land s_{s0} \subseteq ss \land (∇ \in s_{s0} \Leftrightarrow ∇ \in ss)
\]

\[
\lor
((s, \{∇\}) \in B \land ∇ \in ss)
\]

\[
\lor
((s, \{∇\}) \in B \land ∇ \notin ss)
\]

\[
\exists s_{s0} \cdot \left( \begin{array}{c}
((s, s_{s0}) \in B \lor (s, s_{s0} \cup \{∇\}) \in B) \\
\land \\
ss_{s0} \subseteq ss \land (∇ \in ss) \Leftrightarrow ∇ \in ss
\end{array} \right) \lor (s, \{∇\}) \in B
\]

\[
\text{Law B.2.10}
\]

\[
(s, ss) \in \text{bmh}_{0,1,3,2}(B)
\]

\[
\left( \begin{array}{c}
\exists s_{s0} \cdot ((s, s_{s0}) \in B \land (s, \{∇\}) \in B) \\
\lor \\
((s, \{∇\}) \notin B \land (s, \emptyset) \notin B) \\
\land \\
\exists s_{s0} \cdot ((s, s_{s0}) \in B \land s_{s0} \subseteq ss \land ⊥ \notin ss_{s0} \land ⊥ \notin ss)
\end{array} \right)
\]

\[
\text{Definition of } \text{bmh}_{0,1,3,2}
\]

\[
\text{Property of sets}
\]

\[
\text{Revision: 704f887 (2014-02-04 11:14:10 +0000) 202}
\]
\[
\begin{align*}
\exists s_{s_1} : \mathbb{P} State_{\bot} \bullet (s, s_{s_1} \cup \{\bot\} ) &\in \text{bmh}_{0,1,3,2}(B) \land s_{s_1} \subseteq s_{s} \land (\bot \in s_{s_1} \iff \bot \in s_{s}) \\
&\iff ((s, \emptyset) \in B \land (s, \{\bot\}) \in B) \\
&\quad \lor \quad (s, \{\bot\}) \notin B \land (s, \emptyset) \notin B \\
&\quad \land \quad \exists s_{s_0} \bullet ((s, s_{s_0}) \in B \land s_{s_0} \subseteq s_{s} \land \bot \notin s_{s_0} \land \bot \notin s_{s}) 
\end{align*}
\]

\begin{proof}
\begin{align*}
\exists s_{s_1} : \mathbb{P} State_{\bot} \bullet (s, s_{s_1} \cup \{\bot\} ) &\in \text{bmh}_{0,1,3,2}(B) \land s_{s_1} \subseteq s_{s} \land (\bot \in s_{s_1} \iff \bot \in s_{s}) \\
&\iff \{\text{Definition of bmh}_{0,1,3,2}\} \\
&\quad \exists s_{s_1} : \mathbb{P} State_{\bot} \bullet \\
&\quad \begin{cases}
(s, s_{s_1} \cup \{\bot\} ) \in &\{\text{Property of sets}\} \\
\quad ((s, \emptyset) \in B \land (s, \{\bot\}) \in B) \\
\quad \lor &\quad \{\text{Property of sets and predicate calculus}\} \\
\quad (s, \{\bot\}) \notin B \land (s, \emptyset) \notin B \\
\quad \land &\quad \exists s_{s_0} \bullet ((s, s_{s_0}) \in B \land s_{s_0} \subseteq s_{s} \land \bot \notin s_{s_0} \land \bot \notin s_{s}) 
\end{cases}
\end{align*}
\end{proof}
\[\exists ss_1 : \mathbb{P} \text{State}_\perp \bullet ((s, \emptyset) \in B \land (s, \{\perp\}) \in B) \land ss_1 \subseteq ss \land (\perp \in ss_1 \iff \perp \in ss)\]

{Predicate calculus: instantiation of existential quantifier for \(ss_1 = ss\)}

\[\iff ((s, \emptyset) \in B \land (s, \{\perp\}) \in B) \land \exists ss : \mathbb{P} \text{State}_\perp \bullet (s, ss) \in B \land ss_0 \subseteq ss \land (\perp \in ss_0 \iff \perp \in ss)\]

\[\iff (s, ss_1) \in \left\{\begin{array}{l}
\exists ss_1 : \mathbb{P} \text{State}_\perp \bullet \begin{array}{l}
\quad s : \text{State}, ss : \mathbb{P} \text{State}_\perp \\
\quad ((s, \emptyset) \in B \land (s, \{\perp\}) \in B)
\end{array} \\
\quad \lor \\
\quad (s, \{\perp\}) \notin B \land (s, \emptyset) \notin B \\
\quad \lor \\
\quad \exists ss_0 : \mathbb{P} \text{State}_\perp \bullet ((s, ss_0) \in B \land ss_0 \subseteq ss \land (\perp \notin ss_0 \land \perp \notin ss) \\
\end{array} \right\} \land ss_1 \subseteq ss \land (\perp \in ss_1 \iff \perp \in ss)
\]

{Definition of \(\text{bmh}_{0,1,3,2}\)}

Law B.2.12

\[\exists ss_1 : \mathbb{P} \text{State}_\perp \bullet (s, ss_1) \in \text{bmh}_{0,1,3,2}(B) \land ss_1 \subseteq ss \land (\perp \in ss_1 \iff \perp \in ss)\]

\[\iff \left\{\begin{array}{l}
\exists ss_1 : \mathbb{P} \text{State}_\perp \bullet \begin{array}{l}
\quad s : \text{State}, ss : \mathbb{P} \text{State}_\perp \\
\quad ((s, \emptyset) \in B \land (s, \{\perp\}) \in B)
\end{array} \\
\quad \lor \\
\quad (s, \{\perp\}) \notin B \land (s, \emptyset) \notin B \\
\quad \lor \\
\quad \exists ss_0 : \mathbb{P} \text{State}_\perp \bullet ((s, ss_0) \in B \land ss_0 \subseteq ss \land (\perp \notin ss_0 \land \perp \notin ss) \\
\end{array} \right\} \land ss_1 \subseteq ss \land (\perp \in ss_1 \iff \perp \in ss)
\]

{Property of sets}
\[
\exists ss_1 : \mathbb{P} \text{State}_\bot \bullet \\
\left( (s, \emptyset) \in B \land (s, \{ \bot \}) \in B \right) \\
\lor \\
\left( (s, \{ \bot \}) \notin B \land (s, \emptyset) \notin B \right) \\
\land \\
\left( \exists ss_0 \bullet ((s, ss_0) \in B \land ss_0 \subseteq ss_1 \land \bot \notin ss_0 \land \bot \notin ss_1 \right) \\
\land \\
ss_1 \subseteq ss \land (\bot \in ss_1 \iff \bot \in ss) \\
\right) \\
\end{array} \right) \right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) \\
\right) 
\]
Proof.

\[(s, \emptyset) \in bmh_{0,1,3,2}(B) \quad \{\text{Definition of } bmh_{0,1,3,2}\}\]

\[= (s, \emptyset) \in \begin{cases} s : \text{State}, ss : \mathbb{P} \text{State} \downarrow \\ ((s, \emptyset) \in B \land (s, \{\bot\}) \in B) \end{cases} \quad \{\text{Property of sets}\}\]

\[= \begin{cases} ((s, \emptyset) \in B \land (s, \{\bot\}) \in B) \\ (s, \{\bot\}) \notin B \land (s, \emptyset) \notin B \land \exists ss_0 \cdot ((s, ss_0) \in B \land ss_0 \subseteq s \land \bot \notin ss_0 \land \bot \notin ss) \end{cases} \quad \{\text{Property of sets and one-point rule}\}\]

\[= (s, \emptyset) \in B \land (s, \{\bot\}) \in B \quad \{\text{Predicate calculus}\}\]

\[= (s, \emptyset) \in B \land (s, \{\bot\}) \in B\]

\[\blacksquare\]

**Law B.2.14**

\[(s, \{\bot\}) \in bmh_{0,1,3,2}(B) = (s, \emptyset) \in B \land (s, \{\bot\}) \in B\]

Proof.

\[(s, \{\bot\}) \in bmh_{0,1,3,2}(B) \quad \{\text{Definition of } bmh_{0,1,3,2}\}\]

\[= (s, \{\bot\}) \in \begin{cases} s : \text{State}, ss : \mathbb{P} \text{State} \downarrow \\ ((s, \emptyset) \in B \land (s, \{\bot\}) \in B) \end{cases} \quad \{\text{Property of sets}\}\]

Revision: 704f887 (2014-02-04 11:14:10 +0000) 206
\[
\begin{align*}
&\left( (s, \emptyset) \in B \land (s, \{\perp\}) \in B \right) \\
&\quad \lor \left( (s, \{\perp\}) \notin B \land (s, \emptyset) \notin B \right) \\
&\quad \lor \exists ss_0 \bullet \left( (s, ss_0) \in B \land ss_0 \subseteq \{\perp\} \land \perp \notin ss_0 \land \perp \notin \{\perp\} \right)
\end{align*}
\]
\{Property of sets\}

\[
\begin{align*}
&\left( (s, \emptyset) \in B \land (s, \{\perp\}) \in B \right) \\
&\quad \lor (s, \{\perp\}) \notin B \land (s, \emptyset) \notin B \land false
\end{align*}
\{Predicate calculus\}

\[
= (s, \emptyset) \in B \land (s, \{\perp\}) \in B
\]

\square

**Law B.2.15** Provided \(B\) is **BMH0** and **BMH2**-healthy.

\[
B = (B \ni \{ss : \mathbb{P} State_\perp \mid \perp \in ss\}) \cup \{s_0 : State, ss : \mathbb{P} State_\perp \mid (s_0, \emptyset) \in B\}
\]
\[
\Leftrightarrow
\]

**BMH3**

Proof.

\[
B = (B \ni \{ss : \mathbb{P} State_\perp \mid \perp \in ss\}) \cup \{s_0 : State, ss : \mathbb{P} State_\perp \mid (s_0, \emptyset) \in B\}
\]
\{Property of sets\}

\[
\Leftrightarrow (B = \{s : State, ss : State_\perp \mid ((s, ss) \in B \land \perp \notin ss) \lor (s, \emptyset) \in B\})
\]
\{Property of sets\}

\[
\Leftrightarrow \forall s, ss \bullet \left( \begin{array}{c}
(s, ss) \in B \Rightarrow (((s, ss) \in B \land \perp \notin ss) \lor (s, \emptyset) \in B)
\end{array}\right)
\]
\{Propositional calculus\}

\[
\Leftrightarrow \forall s, ss \bullet \left( \begin{array}{c}
((s, \emptyset) \notin B \Rightarrow ((s, ss) \notin B \lor ((s, ss) \in B \land \perp \notin ss)))
\end{array}\right)
\]
\{Propositional calculus: absorption law\}
\[\forall s, ss \bullet \left( (s, \emptyset) \notin B \Rightarrow ((s, ss) \notin B \lor \bot \notin ss) \right) \]
\[\land \quad (s, \emptyset) \notin B \lor (s, ss) \in B \]  \{Propositional calculus\}

\[\forall s, ss \bullet (s, \emptyset) \notin B \Rightarrow ((s, ss) \in B \Rightarrow \bot \notin ss) \]  \{Propositional calculus: introduce term\}

\[\forall s, ss \bullet (s, \emptyset) \notin B \Rightarrow ((s, ss) \in B \Rightarrow \bot \notin ss) \]  \{Propositional calculus\}

\[\forall s, ss \bullet (s, \emptyset) \notin B \Rightarrow ((s, ss) \in B \lor (\bot \in ss \land \bot \notin ss)) \]  \{Propositional calculus\}

\[\forall s, ss \bullet ((s, \emptyset) \in B \land \bot \notin ss) \Rightarrow (s, ss) \in B \]  \{Property of sets\}

\[\forall s, ss \bullet ((s, \emptyset) \in B \land \bot \notin ss) \Rightarrow (s, ss) \in B \]  \{Property of sets\}

\[\forall s, ss \bullet ((s, \emptyset) \in B \land \bot \notin ss) \Rightarrow (s, ss) \in B \]  \{Property of sets\}

\[\forall s, ss \bullet ((s, \emptyset) \in B \land \bot \notin ss) \Rightarrow (s, ss) \in B \]  \{Property of sets\}

\[\forall s, ss \bullet ((s, \emptyset) \in B \land \bot \notin ss) \Rightarrow (s, ss) \in B \]  \{Property of sets\}

\[\forall s, ss \bullet ((s, \emptyset) \in B \land \bot \notin ss) \Rightarrow (s, ss) \in B \]  \{Property of sets\}

\[\forall s, ss \bullet ((s, \emptyset) \in B \land ss \subseteq \bot \land \bot \notin ss) \Rightarrow (s, ss) \in B \]  \{Assumption: \(B\) is BMH2-healthy and Lemma B.3.3\}

\[\forall s, ss \bullet ((s, \emptyset) \in B \land \bot \notin ss) \Rightarrow (s, ss) \in B \]  \{Assumption: \(B\) is BMH2-healthy and Lemma B.3.3\}
\[
\forall s, ss \cdot (s, \emptyset) \notin B \Rightarrow ((s, ss) \notin B \Rightarrow \bot \notin ss)
\]

\[
\iff \\
\forall s, ss \cdot ((s, \emptyset), B \land \emptyset \subseteq ss \land \bot \notin \emptyset \land \bot \notin ss) \Rightarrow (s, ss) \in B
\]

\[
\forall s, ss \cdot ((s, \bot), B \land \bot \subseteq ss \land \bot \in ss \Rightarrow (s, ss) \in B) \quad \text{Assumption: } B \text{ is BMH0-healthy}
\]

\[
\iff \forall s, ss \cdot (s, \emptyset) \notin B \Rightarrow ((s, ss) \notin B \Rightarrow \bot \notin ss)
\]

\[
\iff \forall s \cdot (s, \emptyset) \notin B \Rightarrow \forall ss \cdot ((s, ss) \in B \Rightarrow \bot \notin ss)
\]

\[
\iff \Box
\]

\[
bmb2bm
\]

**Law B.2.16**

\[
bm2bmb(bmh\text{\_upclosed}(B))
\]

\[
= \\
\left\{ s : \text{State}, ss : \mathbb{P} \text{State}_{\bot} \quad \exists ss_0 \cdot (s, ss_0) \in B \land \bot \notin ss_0 \land ss_0 \subseteq ss \land \bot \notin ss \right\} \lor (s, \emptyset) \in B
\]

**Proof.**

\[
bm2bmb(bmh\text{\_upclosed}(B))
\]

\[
= \left\{ s : \text{State}, ss : \mathbb{P} \text{State}_{\bot} \quad (s, ss) \in \left\{ s : \text{State}, ss : \mathbb{P} \text{State}_{\bot} \quad \exists ss_0 \cdot (s, ss_0) \in B \land \bot \notin ss_0 \land ss_0 \subseteq ss \land \bot \notin ss \right\} \lor (s, \emptyset) \in \left\{ s : \text{State}, ss : \mathbb{P} \text{State}_{\bot} \quad \exists ss_0 \cdot (s, ss_0) \in B \land \bot \notin ss_0 \land ss_0 \subseteq ss \land \bot \notin ss \right\}ight\}
\]

\[
\quad \land \bot \notin ss
\]

\[
\quad \lor (s, \emptyset) \in \left\{ s : \text{State}, ss : \mathbb{P} \text{State}_{\bot} \quad \exists ss_0 \cdot (s, ss_0) \in B \land \bot \notin ss_0 \land ss_0 \subseteq ss \land \bot \notin ss \right\}
\]

\[
\quad \text{Property of sets and predicate calculus}
\]
\[
\begin{align*}
\{ 
& s : \text{State}, ss : \mathbb{P}\text{State} \\
& \exists ss_0 \cdot (s, ss_0) \in B \land \bot \notin ss_0 \land ss_0 \subseteq ss \land \bot \notin ss \\
& \lor \\
& \exists ss_0 \cdot (s, ss_0) \in B \land \bot \notin ss_0 \land ss_0 \subseteq \emptyset \\
\} 
\end{align*}
\]
{Case-analysis on \( ss_0 \) and one-point rule}

\[
\begin{align*}
\{ 
& s : \text{State}, ss : \mathbb{P}\text{State} \\
& \exists ss_0 \cdot (s, ss_0) \in B \land \bot \notin ss_0 \land ss_0 \subseteq ss \land \bot \notin ss \\
& \lor \\
& (s, \emptyset) \in B \\
\} 
\end{align*}
\]

**Theorem B.2.1**

\[
\text{bmh}_{0,1,3,2} \circ \text{bm2bmb(bmhupclosed}(B)) = \text{bm2bmb(bmhupclosed}(B))
\]

**Proof.**

\[
\begin{align*}
\{ \text{Definition of \text{bmh}_{0,1,3,2}} \} \\
\{ \text{Law B.2.19 and Law B.2.18} \} \\
\{ \text{Predicate calculus and definition of \text{bm2bmb(bmhupclosed}(B)) (Law B.2.16)} \}
\end{align*}
\]
\[
\begin{array}{l}
\{ \begin{array}{ll}
\text{s : State, ss : } \mathbb{P} \text{ State}_\bot \\
(s, \emptyset) \in B \\
\lor \\
\exists ss_0 \bullet \begin{cases} 
\begin{array}{l}
\exists ss_1 \bullet (s, ss_1) \in B \land \bot \notin ss \land ss_1 \subseteq ss \land \bot \notin ss_1 \\
\lor \\
(s, \emptyset) \in B \\
\land ss_0 \subseteq ss \land \bot \notin ss \land ss_0 \subseteq ss \\
\lor \\
\exists ss_0 \bullet (s, \emptyset) \in B \land ss_0 \subseteq ss \land \bot \notin ss_0 \land \bot \notin ss
\end{array}
\end{cases}
\end{array}
\} \\
\{ \text{Variable renaming and property of sets} \}
\end{array}
\]

\[
\begin{array}{l}
\{ \begin{array}{ll}
\text{s : State, ss : } \mathbb{P} \text{ State}_\bot \\
(s, \emptyset) \in B \\
\lor \\
\exists ss_0 \bullet \begin{cases}
\begin{array}{l}
\exists ss_1 \bullet (s, ss_1) \in B \land \bot \notin ss \land ss_1 \subseteq ss \land \bot \notin ss_1 \\
\lor \\
(s, \emptyset) \in B \\
\land ss_0 \subseteq ss \land \bot \notin ss \land ss_0 \subseteq ss \\
\lor \\
\exists ss_0 \bullet (s, \emptyset) \in B \land ss_0 \subseteq ss \land \bot \notin ss_0 \land \bot \notin ss
\end{array}
\end{cases}
\end{array}
\} \\
\{ \text{Predicate calculus} \}
\end{array}
\]

\[
\begin{array}{l}
\{ \begin{array}{ll}
\text{s : State, ss : } \mathbb{P} \text{ State}_\bot \\
(s, \emptyset) \in B \\
\lor \\
\exists ss_0 \bullet \begin{cases}
\begin{array}{l}
\exists ss_1 \bullet (s, ss_1) \in B \land \bot \notin ss \land ss_1 \subseteq ss \land \bot \notin ss_1 \\
\lor \\
(s, \emptyset) \in B \\
\land ss_0 \subseteq ss \land \bot \notin ss \land ss_0 \subseteq ss \\
\lor \\
\exists ss_0 \bullet (s, \emptyset) \in B \land ss_0 \subseteq ss \land \bot \notin ss_0 \land \bot \notin ss
\end{array}
\end{cases}
\end{array}
\} \\
\{ \text{Predicate calculus} \}
\end{array}
\]

\[
\begin{array}{l}
\{ \begin{array}{ll}
\text{s : State, ss : } \mathbb{P} \text{ State}_\bot \\
(s, \emptyset) \in B \\
\lor \\
\exists ss_0 \bullet \begin{cases}
\begin{array}{l}
\exists ss_1 \bullet (s, ss_1) \in B \land \bot \notin ss \land ss_1 \subseteq ss \land \bot \notin ss \\
\lor \\
((s, \emptyset) \in B \land \exists ss_0 \bullet ss_0 \subseteq ss \land \bot \notin ss_0 \land \bot \notin ss)
\end{array}
\end{cases}
\end{array}
\} \\
\{ \text{Predicate calculus: absorption law} \}
\end{array}
\]
\[
\begin{align*}
= & \left\{ s : \text{State}, \; ss : \mathbb{P} \text{State}_\bot \\
& (s, \emptyset) \in B \\
& \lor \\
& (\exists ss_1 \cdot (s, ss_1) \in B \land \bot \notin ss_1 \land ss_1 \subseteq ss \land \bot \notin ss) \right\} \\
\{\text{Law B.2.16}\}
\end{align*}
\]

\[
= bm2bmb(bm_{\text{upclosed}}(B))
\]

\[ \square \]

\textbf{Law B.2.17}

\[
bm2bm(bm_{0,1,3,2}(B))
\]

\[
= \left\{ s : \text{State}, \; ss : \mathbb{P} \text{State}_\bot \\
& ((s, \emptyset) \in B \land (s, \{ \bot \}) \in B) \land \bot \notin ss \\
& \lor \\
& \left( (s, \{ \bot \}) \notin B \land (s, \emptyset) \notin B \\
& \land \\
& (\exists ss_0 \cdot (s, ss_0) \in B \land ss_0 \subseteq ss \land \bot \notin ss_0 \land \bot \notin ss) \right) \right\}
\]

\[ \{\text{Definition of bm2bm}\}
\]

\[ \{\text{Definition of bm_{0,1,3,2}}\} \]

\textbf{Proof.}

\[
bm2bm(bm_{0,1,3,2}(B))
\]

\[
= \{ s : \text{State}, \; ss : \mathbb{P} \text{State}_\bot \mid ((s, ss) \in bm_{0,1,3,2}(B) \land \bot \notin ss) \}
\]

\[ \{\text{Definition of bm_{0,1,3,2}}\} \]

\[
= \left\{ s : \text{State}, \; ss : \mathbb{P} \text{State}_\bot \\
& ((s, \emptyset) \in B \land (s, \{ \bot \}) \in B) \\
& \lor \\
& \left( (s, \{ \bot \}) \notin B \land (s, \emptyset) \notin B \\
& \land \\
& (\exists ss_0 \cdot (s, ss_0) \in B \land ss_0 \subseteq ss \land \bot \notin ss_0 \land \bot \notin ss) \right) \right\}
\]

\[ \{\text{Property of sets}\} \]
Theorem B.2.2 (bmb2bm-is-bmhapclosed)

\[
\text{bmhapclosed} \circ \text{bmb2bm} (\text{bmh}_{0,1,3,2}(B)) = \text{bmb2bm} (\text{bmh}_{0,1,3,2}(B))
\]

Proof.

\[
\text{bmhapclosed} \circ \text{bmb2bm} (\text{bmh}_{0,1,3,2}(B)) = \text{bmb2bm} (\text{bmh}_{0,1,3,2}(B))
\]

\[
= \left\{ \begin{array}{l}
  s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
  \left\{ \begin{array}{l}
  ((s, \emptyset) \in B \land (s, \bot) \notin B) \\
  \lor \\
  (s, \bot) \notin B \land (s, \emptyset) \notin B \\
  \land \\
  \exists ss_0 \bullet (s, ss_0) \in B \land ss_0 \subseteq ss \land \bot \notin ss_0 \land \bot \notin ss \\
  \land \bot \notin ss_0 \land ss_0 \subseteq ss \land \bot \notin ss \\
  \end{array} \right. \\
  \end{array} \right. 
\]

\[
\{ \text{Predicate calculus} \}
\]
\[
\begin{align*}
&\{ s : State, ss : \mathbb{P} State \} \\
&= \{ s : State, ss : \mathbb{P} State \} \\
&= \{ s : State, ss : \mathbb{P} State \} \\
&= \{ s : State, ss : \mathbb{P} State \} \\
&= \{ s : State, ss : \mathbb{P} State \} \\
&= bmb2bm(bmh_{0,1,3,2}(B))
\end{align*}
\]
Law B.2.18

\[(s, \emptyset) \in bmb2bm(bmh_{\text{upclosed}}) = (s, \emptyset) \in B\]

Proof.

\[(s, \emptyset) \in bmb2bm(bmh_{\text{upclosed}}) \]

\[
\begin{align*}
\{\text{Definition of } bmb2bm(bmh_{\text{upclosed}}} \text{ (Law B.2.16)}\} \\
= (s, \emptyset) \in \begin{cases} 
    \quad \text{s : State, ss : } \mathbb{P} \text{ State} \\
    \quad \quad \exists ss_0 \bullet (s, ss_0) \in B \land \bot \notin ss_0 \land ss_0 \subseteq ss \land \bot \notin ss \\
    \quad \quad \quad \lor \\
    \quad \quad (s, \emptyset) \in B 
\end{cases} \\
\{\text{Property of sets}\} \\
= \left( \exists ss_0 \bullet (s, ss_0) \in B \land \bot \notin ss_0 \land ss_0 \subseteq \emptyset \land \bot \notin \emptyset \right) \\
\{\text{Predicate calculus and one-point rule}\} \\
= (s, \emptyset) \in B \\
\end{align*}
\]

\[\square\]

Law B.2.19

\[(s, \{\bot\}) \in bmb2bm(bmh_{\text{upclosed}}) = (s, \emptyset) \in B\]

Proof.

\[(s, \{\bot\}) \in bmb2bm(bmh_{\text{upclosed}})\]

\[
\begin{align*}
\{\text{Definition of } bmb2bm(bmh_{\text{upclosed}}} \text{ (Law B.2.16)}\} \\
= (s, \{\bot\}) \in \begin{cases} 
    \quad \text{s : State, ss : } \mathbb{P} \text{ State} \\
    \quad \quad \exists ss_0 \bullet (s, ss_0) \in B \land \bot \notin ss_0 \land ss_0 \subseteq ss \land \bot \notin ss \\
    \quad \quad \quad \lor \\
    \quad \quad (s, \emptyset) \in B 
\end{cases} \\
\{\text{Property of sets}\} \\
\end{align*}
\]
\[
\exists s_{s_0} \cdot (s, s_{s_0}) \in B \land \bot \notin s_{s_0} \land s_{s_0} \subseteq \{\bot\} \land \bot \notin \{\bot\}
\]
\[
= \left( \exists s_{s_0} \cdot (s, s_{s_0}) \in B \land \bot \notin s_{s_0} \land s_{s_0} \subseteq \{\bot\} \land \bot \notin \{\bot\} \right)
\]
\[
= (s, \emptyset) \in B \quad \{\text{Property of sets and predicate calculus}\}
\]

**B.3 Set theory**

**Lemma B.3.1**

\[
\exists s_{s_0} \cdot (s, s_{s_0} \cup \{\bot\}) \in B \land s_{s_0} \subseteq ss \land (\bot \in ss_0 \iff \bot \in ss)
\]

\[
\iff
\exists s_{s_0} \cdot (s, s_{s_0}) \in B \land s_{s_0} \subseteq (ss \cup \{\bot\}) \land \bot \in s_{s_0}
\]

**Proof.**

\[
\exists s_{s_0} \cdot (s, s_{s_0} \cup \{\bot\}) \in B \land s_{s_0} \subseteq ss \land (\bot \in ss_0 \iff \bot \in ss)
\]

\[
= \exists s_{s_0} \cdot \left( (s, s_{s_0} \cup \{\bot\}) \in B \land s_{s_0} \subseteq ss \land \bot \in s_{s_0} \land \bot \in ss \right)
\]

\[
= \exists s_{s_0} \cdot \left( (s, s_{s_0} \cup \{\bot\}) \in B \land s_{s_0} \subseteq ss \land \bot \notin s_{s_0} \land \bot \notin ss \right)
\]

\[
\{\text{Predicate calculus and property of sets}\}
\]

\[
= \exists s_{s_0} \cdot \left( (s, s_{s_0} \cup \{\bot\}) \in B \land s_{s_0} \subseteq (ss \cup \{\bot\}) \land \bot \in s_{s_0} \land \bot \in ss \right)
\]

\[
\{\text{Predicate calculus}\}
\]

\[
= \exists s_{s_0} \cdot \left( (s, s_{s_0} \cup \{\bot\}) \in B \land s_{s_0} \subseteq (ss \cup \{\bot\}) \land \bot \in s_{s_0} \land \bot \in ss \right)
\]

\[
\{\text{Predicate calculus and property of sets}\}
\]

\[
= \exists s_{s_0} \cdot \left( (s, s_{s_0} \cup \{\bot\}) \in B \land s_{s_0} \subseteq (ss \cup \{\bot\}) \land \bot \in s_{s_0} \land \bot \in ss \right)
\]

\[
\{\text{Predicate calculus}\}
\]

\[
= \exists s_{s_0} \cdot \left( (s, s_{s_0} \cup \{\bot\}) \in B \land s_{s_0} \subseteq (ss \cup \{\bot\}) \land \bot \in s_{s_0} \land \bot \in ss \right)
\]

\[
\{\text{Property of sets (Lemma B.3.2)}\}
\]

\[
= \exists t, s_{s_0} \cdot (s, t) \in B \land t = s_{s_0} \cup \{\bot\} \land s_{s_0} \subseteq ss \land \bot \notin s_{s_0} \land \bot \notin ss
\]

\[
\{\text{Property of sets}\}
\]

\[
= \exists t, s_{s_0} \cdot (s, t) \in B \land t \setminus \{\bot\} = s_{s_0} \land s_{s_0} \subseteq ss \land \bot \notin t \land \bot \notin ss
\]

\[
\{\text{One-point rule and substitution}\}
\]
Lemma B.3.2 (A-setminus-x)

\[ (A = B \cup \{x\} \land x \notin B) \iff (A \setminus \{x\} = B \land x \in A) \]

Proof.
\[
A = B \cup \{x\} \land x \notin B \quad \{\text{Set equality}\}
\]
\[
= (\forall y \cdot y \in A \iff y \in (B \cup \{x\})) \land x \notin B \quad \{\text{Propositional calculus}\}
\]
\[
= (\forall y \cdot (y \in A \Rightarrow y \in (B \cup \{x\})) \land (y \notin (B \cup \{x\})) \Rightarrow y \in A) \land x \notin B \quad \{\text{Property of sets}\}
\]
\[
= \left( (\forall y \cdot \left( (y \in A \land y \notin \{x\}) \Rightarrow y \in B \right) \land (y \in B \Rightarrow y \in A) \Rightarrow y \in A) \right) \land x \notin B \quad \{\text{Propositional calculus}\}
\]
\[
= \left( \forall y \cdot \left( (y \in A \land y \notin \{x\}) \Rightarrow y \in B \right) \land (y \in B \Rightarrow y \in A) \Rightarrow y \in A \right) \land x \notin B \quad \{\text{Lemma B.3.4 and propositional calculus}\}
\]
\[
= \left( \forall y \cdot \left( (y \in A \land y \notin \{x\}) \Rightarrow y \in B \right) \land (y \in B \Rightarrow y \in A) \land (y \notin \{x\}) \Rightarrow y \in A \right) \land x \notin B \quad \{\text{Propositional calculus}\}
\]
\[
\begin{align*}
&= \left( \forall y \cdot ((y \in A \land y \notin \{x\}) \Leftrightarrow (y \in B)) \land (y \in \{x\} \Rightarrow y \in A) \right) \\
&\quad \text{\{Property of sets\}} \\
&= (A \setminus \{x\} = B \land \{x\} \subseteq A) \quad \text{\{Lemma B.3.3 and propositional calculus\}} \\
&= (A \setminus \{x\} = B \land x \in A) \quad \square
\end{align*}
\]

**Lemma B.3.3 (set-membership-subset-1)**

\[
\{x\} \subseteq A \Leftrightarrow x \in A
\]

**Proof.**

\[
\begin{align*}
\{x\} \subseteq A & \quad \text{\{Definition of subset inclusion\}} \\
= \forall y \cdot y \in \{x\} \Rightarrow y \in A & \quad \text{\{Propositional calculus\}} \\
= \forall y \cdot \neg (y \in \{x\} \land y \notin A) & \quad \text{\{Propositional calculus\}} \\
= \neg \exists y \cdot y = x \land y \notin A & \quad \text{\{One-point rule\}} \\
= \neg (x \notin A) & \quad \text{\{Propositional calculus\}} \\
= x \in A & 
\end{align*}
\]

\square

**Lemma B.3.4 (set-membership-subset-2)**

\[
x \notin A \Leftrightarrow (\forall y \cdot y \in A \Rightarrow y \notin \{x\})
\]

**Proof.**

\[
\begin{align*}
x \notin A & \quad \text{\{Propositional calculus\}} \\
= \neg (x \in A) & \quad \text{\{Introduce fresh variable\}} \\
= \neg (\exists y \cdot y = x \land y \in A) & \quad \text{\{Property of sets\}} \\
= \neg (\exists y \cdot y \in \{x\} \land y \in A) & \quad \text{\{Propositional calculus\}} \\
= \forall y \cdot y \in A \Rightarrow y \notin \{x\} & 
\end{align*}
\]

\square

**Lemma B.3.5**

\[
(A = (B \cup \{x\}) \land x \in B) \Leftrightarrow (A = B \land x \in B)
\]
Proof. (Implication)

\[ A = B \cup \{x\} \land x \in B \]  \quad \{\text{Property of sets}\}

\[ = (A \subseteq (B \cup \{x\}) \land (B \cup \{x\}) \subseteq A \land x \in B) \]  \quad \{\text{Lemma B.3.3}\}

\[ = (A \subseteq (B \cup \{x\}) \land (B \cup \{x\}) \subseteq A \land \{x\} \subseteq B) \]  \quad \{\text{Property of sets}\}

\[ = (A \subseteq (B \cup \{x\}) \land B \subseteq A \land \{x\} \subseteq A \land \{x\} \subseteq B) \]  \quad \{\text{Property of sets}\}

\[ = (A \subseteq (B \cup \{x\}) \land B \subseteq A \land \{x\} \subseteq A \land (\{x\} \cup B = B)) \]  \quad \{\text{Propositional calculus}\}

\[ = (A \subseteq (B \cup \{x\}) \land B \subseteq A \land \{x\} \subseteq A \land (\{x\} \cup B) \subseteq B) \land B \subseteq (\{x\} \cup B) \]  \quad \{\text{Transitivity of subset inclusion and propositional calculus}\}

\[ = (A \subseteq (B \cup \{x\}) \land B \subseteq A \land \{x\} \subseteq A \land (\{x\} \cup B) \subseteq B \land A \subseteq B \land B \subseteq (\{x\} \cup B)) \]  \quad \{\text{Propositional calculus}\}

\[ \Rightarrow B \subseteq A \land A \subseteq B \land \{x\} \cup B \subseteq B \land B \subseteq (\{x\} \cup B) \]  \quad \{\text{Property of sets}\}

\[ = (B = A \land \{x\} \cup B) \]  \quad \{\text{Lemma B.3.3}\}

\[ = (B = A \land x \in B) \]

\[ \square \]

Proof. (Reverse implication)

\[ (B = A \land x \in B) \]  \quad \{\text{Lemma B.3.3}\}

\[ (B = A \land \{x\} \subseteq B) \]  \quad \{\text{Property of sets}\}

\[ = (A \subseteq B \land B \subseteq A \land \{x\} \subseteq B) \]  \quad \{\text{Transitivity of subset inclusion and propositional calculus}\}

\[ = (A \subseteq B \land B \subseteq A \land \{x\} \subseteq B) \land \{x\} \subseteq A \]  \quad \{\text{Property of sets}\}

\[ = (A \subseteq B \land B \subseteq A \land \{x\} \subseteq B \land \{x\} \subseteq A \land (B \cup \{x\}) \subseteq A \land (A \cup \{x\}) \subseteq B \]  \quad \{\text{Property of sets}\}

\[ = (A \subseteq B \land B \subseteq A \land \{x\} \subseteq B \land (\{x\} \cup B = B) \land \{x\} \subseteq A \land (B \cup \{x\}) \subseteq A \land (A \cup \{x\}) \subseteq B \]  \quad \{\text{Property of sets and weaken predicate}\}

\[ \Rightarrow (A \subseteq B \land B \subseteq A \land \{x\} \subseteq B \land B \subseteq (\{x\} \cup B) \land \{x\} \subseteq A \land (B \cup \{x\}) \subseteq A \land (A \cup \{x\}) \subseteq B \]  \quad \{\text{Transitivity of subset inclusion and propositional calculus}\}

\[ \Rightarrow (A \subseteq B \land B \subseteq A \land \{x\} \subseteq B \land B \subseteq (\{x\} \cup B) \land \{x\} \subseteq A \land (B \cup \{x\}) \subseteq A \land (A \cup \{x\}) \subseteq A \]  \quad \{\text{Property of sets}\}

\[ = (A = B \land B \subseteq (\{x\} \cup B) \land \{x\} \subseteq B \land \{x\} \subseteq A \land (B \cup \{x\}) = A \]  \quad \{\text{Propositional calculus}\}
⇒ (\{x\} ⊆ B \land (B \cup \{x\}) = A) \quad \text{Lemma B.3.3}

= ((B \cup \{x\}) = A \land x \in B)

□

Lemma B.3.6

((A \cup \{x\}) \subseteq (B \cup \{x\}) \land x \notin A \land x \notin B) \iff (A \subseteq B \land x \notin A \land x \notin B)

Proof.

(A \cup \{x\}) \subseteq (B \cup \{x\}) \land x \notin A \land x \notin B \quad \text{Definition of subset inclusion}

= \forall y \bullet y \in (A \cup \{x\}) \Rightarrow y \in (B \cup \{x\}) \land x \notin A \land x \notin B \quad \text{Property of sets}

= \forall y \bullet (y \in A \lor y \in \{x\}) \Rightarrow (y \in B \lor y \in \{x\}) \land x \notin A \land x \notin B \quad \text{Propositional calculus}

= \forall y \bullet (\forall y \bullet y \in A \Rightarrow (y \in B \lor y \in \{x\}) \land x \notin A \land x \notin B) \quad \text{Lemma B.3.4}

= \forall y \bullet (\forall y \bullet y \in A \lor y \notin \{x\}) \land (\forall y \bullet y \in B \lor y \notin \{x\}) \land x \notin A \land x \notin B \quad \text{Propositional calculus}

= \forall y \bullet (y \in A \lor y \in B) \land ((y \in A \lor y \in B) \Rightarrow (y \notin \{x\})) \quad \text{Propositional calculus and definition of subset inclusion}

= A \subseteq B \land \forall y \bullet ((y \in A \lor y \in B) \Rightarrow (y \notin \{x\})) \quad \text{Property of sets and Lemma B.3.4}

= A \subseteq B \land x \notin (A \cup B) \quad \text{Propositional calculus and property of sets}

= A \subseteq B \land x \notin A \land x \notin B

□
Appendix C

Predicative model

C.1 \( d2bmb \)

Lemma C.1.1 (\( d2bmb\)-A-healthy)

\[
d2bmb(A(P)) = \left\{ s : \text{State}, ss : \mathbb{P}\text{State}_\perp \left| \begin{array}{l}
\exists ac_0 : \mathbb{P}\text{State} \bullet \\
(Pf[ac_0/ac'] \lor (Pt[ac_0/ac'] \land \perp \notin ss \land ss \neq \emptyset)) \land ac_0 \subseteq ss
\end{array} \right. \right\}
\]

Proof.

\[
d2bmb(A(P)) = d2bmb(\neg \text{PBMH1}(P^t) \vdash \text{PBMH1}(P^t) \land ac' \neq \emptyset) \quad \{\text{Definition of PBMH1}\}
\]

\[
d2bmb(\neg (P^f ; ac \subseteq ac') \vdash (P^t ; ac \subseteq ac') \land ac' \neq \emptyset) \quad \{\text{Definition of } d2bmb \text{ (Definition 55)}\}
\]

\[
= \left\{ s : \text{State}, ss : \mathbb{P}\text{State}_\perp \left| \begin{array}{l}
((\neg (P^f ; ac \subseteq ac') \Rightarrow ((P^t ; ac \subseteq ac') \land ac' \neq \emptyset))[ss/ac'] \land \perp \notin ss) \\
\lor ((P^f ; ac \subseteq ac')|ss\setminus\{\perp\}/ac' \land \perp \in ss)
\end{array} \right. \right\} \quad \{\text{Definition of sequential composition}\}
\]
\[
\begin{array}{c}
s : State, \: ss : \mathbb{P} \mathit{State}_\bot \\
a \equiv \left\{ \begin{array}{c}
\left( \neg \exists a c_0 : \mathbb{P} \mathit{State} \bullet P^f[ac_0/ac'] \land ac_0 \subseteq ac' \right) \\
\land \bot \notin ss \\
\end{array} \right\} [ss/ac'] \\
\lor \left( \exists a c_0 : \mathbb{P} \mathit{State} \bullet P^f[ac_0/ac'] \land ac_0 \subseteq ac' \land ac' \neq \emptyset \\
\land \bot \notin ss \\
\right\} [ss/ac'] \\
\end{array}
\]

\[
\begin{array}{c}
s : State, \: ss : \mathbb{P} \mathit{State}_\bot \\
a \equiv \left\{ \begin{array}{c}
\left( \exists a c_0 : \mathbb{P} \mathit{State}_\bot \bullet P^f[ac_0/ac'] \land ac_0 \subseteq ac' \right) \\
\land \bot \notin ss \\
\end{array} \right\} [ss/ac'] \\
\lor \left( \exists a c_0 : \mathbb{P} \mathit{State}_\bot \bullet P^f[ac_0/ac'] \land ac_0 \subseteq ac' \land \bot \notin ss \\
\land \bot \notin ss \\
\right\} [ss \setminus \{\bot\}/ac'] \\
\end{array}
\]

\[
\begin{array}{c}
s : State, \: ss : \mathbb{P} \mathit{State}_\bot \\
a \equiv \left\{ \begin{array}{c}
\left( \exists a c_0 : \mathbb{P} \mathit{State}_\bot \bullet P^f[ac_0/ac'] \land ac_0 \subseteq ss \right) \\
\land \bot \notin ac_0 \land \bot \notin ss \\
\end{array} \right\} [ss \setminus \{\bot\}/ac'] \\
\lor \left( \exists a c_0 : \mathbb{P} \mathit{State}_\bot \bullet P^f[ac_0/ac'] \land ac_0 \subseteq ss \land \bot \notin ss \\
\land \bot \notin ss \\
\right\} [ss \setminus \{\bot\}/ac'] \\
\end{array}
\]

\[
\begin{array}{c}
s : State, \: ss : \mathbb{P} \mathit{State}_\bot \\
a \equiv \left\{ \begin{array}{c}
\left( \exists a c_0 : \mathbb{P} \mathit{State}_\bot \bullet P^f[ac_0/ac'] \land ac_0 \subseteq \left( ss \setminus \{\bot\} \right) \right) \\
\land \bot \notin ss \\
\end{array} \right\} \{\text{Propositional calculus and property of sets}\}
\end{array}
\]
\[
\begin{align*}
\exists a_0 : \mathbb{P} \text{State}_\bot & \Rightarrow P^f[a_0/ac'] \land a_0 \subseteq ss \\
\land \perp \notin a_0 & \land \perp \notin ss \\
\end{align*}
\]
\[
\begin{align*}
\exists a_0 : \mathbb{P} \text{State}_\bot & \Rightarrow P^f[a_0/ac'] \land a_0 \subseteq ss \\
\land \perp \notin a_0 & \land \perp \notin ss \land ss \neq \emptyset \\
\end{align*}
\]
\[
\begin{align*}
\exists a_0 : \mathbb{P} \text{State}_\bot & \Rightarrow P^f[a_0/ac'] \land a_0 \subseteq (ss \setminus \{\perp\}) \\
\land \perp \notin a_0 & \land \perp \in ss \\
\end{align*}
\]
{\{Property of sets\}}

\[
\begin{align*}
\exists a_0 : \mathbb{P} \text{State}_\bot & \Rightarrow P^f[a_0/ac'] \land a_0 \subseteq ss \\
\land \perp \notin a_0 & \land \perp \notin ss \land ss \neq \emptyset \\
\end{align*}
\]
\[
\begin{align*}
\exists a_0 : \mathbb{P} \text{State}_\bot & \Rightarrow (\forall x : \mathbb{P} \text{State}_\bot \cdot x \in a_0 \Rightarrow x \in ss) \\
\land (\forall x : \mathbb{P} \text{State}_\bot \cdot x \in ac \Rightarrow x \notin \{\perp\}) \\
\land \perp \notin a_0 & \land \perp \in ss \\
\end{align*}
\]
{\{Propositional calculus, property of sets and Lemma B.3.4\}}

\[
\begin{align*}
\exists a_0 : \mathbb{P} \text{State}_\bot & \Rightarrow P^f[a_0/ac'] \land a_0 \subseteq ss \\
\land \perp \notin a_0 & \land \perp \notin ss \land ss \neq \emptyset \\
\end{align*}
\]
\[
\begin{align*}
\exists a_0 : \mathbb{P} \text{State}_\bot & \Rightarrow P^f[a_0/ac'] \land a_0 \subseteq ss \\
\land \perp \notin a_0 & \land \perp \notin ss \land ss \neq \emptyset \\
\end{align*}
\]
{\{Propositional calculus\}}

\[
\begin{align*}
\exists a_0 : \mathbb{P} \text{State}_\bot & \Rightarrow P^f[a_0/ac'] \land a_0 \subseteq ss \\
\land \perp \notin a_0 & \land \perp \notin ss \land ss \neq \emptyset \\
\end{align*}
\]
{\{Propositional calculus\}}
Lemma C.1.2

\[ \exists s_{0} : \text{State}_{\bot} \bullet (s, s_{0} \cup \{\bot\}) \in d2bmb(\mathbf{A}(P)) \land s_{0} \subseteq ss \land (\bot \in s_{0} \iff \bot \in ss) \]

\[ = \exists a_{0} : \text{State} \bullet P^{f}[a_{0}/a'] \land a_{0} \subseteq ss \]

Proof.

\[ \exists s_{0} : \text{State}_{\bot} \bullet (s, s_{0} \cup \{\bot\}) \in d2bmb(\mathbf{A}(P)) \land s_{0} \subseteq ss \land (\bot \in s_{0} \iff \bot \in ss) \]

\[ = \left( \exists s_{0} : \text{State}_{\bot} \bullet \right) \]

\[ \left( \exists a_{0} : \text{State} \bullet \right) \]

\[ (P^{f}[a_{0}/a'] \lor (P^{f}[a_{0}/a'] \land ss \neq \emptyset \land \bot \notin ss)) \land a_{0} \subseteq ss \]

\[ \land ss_{0} \subseteq ss \land (\bot \in ss_{0} \iff \bot \in ss) \]

\[ = \left( \exists s_{0} : \text{State}_{\bot}, a_{0} : \text{State} \bullet \right) \]

\[ (P^{f}[a_{0}/a'] \lor (P^{f}[a_{0}/a'] \land (ss_{0} \cup \{\bot\}) \neq \emptyset \land \bot \notin (ss_{0} \cup \{\bot\}))) \]

\[ \land a_{0} \subseteq (ss_{0} \cup \{\bot\}) \land ss_{0} \subseteq ss \land (\bot \in ss_{0} \iff \bot \in ss) \]

\[ = \left( \exists s_{0} : \text{State}_{\bot}, a_{0} : \text{State} \bullet \right) \]

\[ P^{f}[a_{0}/a'] \land a_{0} \subseteq (ss_{0} \cup \{\bot\}) \land ss_{0} \subseteq ss \land (\bot \in ss_{0} \iff \bot \in ss) \]

\[ \text{Property of sets and predicate calculus} \]

\[ \text{Property of sets} \]
\[
\exists s_{s0} : P\text{ State} \land s_{s0} \subseteq \text{ss} \land (\bot \in s_{s0} \leftrightarrow \bot \in \text{ss})
\]

\[
\exists s_{s0} : P\text{ State} \land Pf[(ac_0/\ac') \land ac_0 \subseteq s_{s0} \land \bot \in s_{s0} \leftrightarrow \bot \in \text{ss})
\]

\[
\exists s_{s0} : P\text{ State} \land Pf[(ac_0/\ac') \land ac_0 \subseteq s_{s0} \land \bot \in s_{s0} \leftrightarrow \bot \in \text{ss})
\]

\[
\exists s_{s0} : P\text{ State} \land s_{s0} \subseteq \text{ss} \land (\bot \in s_{s0} \leftrightarrow \bot \in \text{ss})
\]

\[
\exists s_{s0} : P\text{ State} \land (s, s_{s0}) \in d^2bmb(A(P)) \land s_{s0} \subseteq s_{s0} \land (\bot \in s_{s0} \leftrightarrow \bot \in \text{ss})
\]

\[
\exists s_{s0} : P\text{ State} \land (s, s_{s0}) \in d^2bmb(A(P)) \land s_{s0} \subseteq s_{s0} \land (\bot \in s_{s0} \leftrightarrow \bot \in \text{ss})
\]

\[
\exists s_{s0} : P\text{ State} \land s_{s0} \subseteq \text{ss} \land (\bot \in s_{s0} \leftrightarrow \bot \in \text{ss})
\]


Lemma C.1.3

\[
\exists s_{s0} : P\text{ State} \land (s, s_{s0}) \in d^2bmb(A(P)) \land s_{s0} \subseteq s_{s0} \land (\bot \in s_{s0} \leftrightarrow \bot \in \text{ss})
\]

Proof.

\[
\exists s_{s0} : P\text{ State} \land (s, s_{s0}) \in d^2bmb(A(P)) \land s_{s0} \subseteq s_{s0} \land (\bot \in s_{s0} \leftrightarrow \bot \in \text{ss})
\]
\[
= \left( \exists s_0 : \mathbb{P} \text{State} \land, a_0 : \mathbb{P} \text{State} \bullet \right.
\left. \left( P^f[ac_0/ac'] \lor \left( P^t[ac_0/ac'] \land s_0 \neq \emptyset \land \bot \notin s_0 \right) \right) \land a_0 \subseteq s_0 \land s_0 \subseteq s \land (\bot \in s_0 \Leftrightarrow \bot \in ss) \right) \}
\text{Predicate calculus}
\]
\[
= \left( \exists s_0 : \mathbb{P} \text{State} \land, a_0 : \mathbb{P} \text{State} \bullet \right.
\left. \left( P^f[ac_0/ac'] \land a_0 \subseteq s_0 \land s_0 \subseteq s \land (\bot \in s_0 \Leftrightarrow \bot \in ss) \right) \lor \left( \exists s_0 : \mathbb{P} \text{State} \land, a_0 : \mathbb{P} \text{State} \bullet \right.
\left. \left( P^f[ac_0/ac'] \land a_0 \subseteq s_0 \land s_0 \subseteq s \land \bot \notin s_0 \land \bot \notin ss \right) \right) \right) \}
\text{Predicate calculus}
\]
\[
= \left( \exists s_0 : \mathbb{P} \text{State} \land, a_0 : \mathbb{P} \text{State} \bullet \right.
\left. \left( P^f[ac_0/ac'] \land a_0 \subseteq s_0 \land s_0 \subseteq s \land (\bot \in s_0 \Leftrightarrow \bot \in ss) \right) \lor \left( \exists s_0 : \mathbb{P} \text{State} \land, a_0 : \mathbb{P} \text{State} \bullet \right.
\left. \left( P^f[ac_0/ac'] \land a_0 \subseteq s_0 \land s_0 \subseteq s \land \bot \notin s_0 \land \bot \notin ss \right) \right) \right) \}
\text{Predicate calculus}
\]
\[
= \left( \exists ac_0 : \mathbb{P} \text{State} \bullet \right.
\left. \left( P^f[ac_0/ac'] \land a_0 \subseteq s \land \bot \in ss \right) \lor \left( \exists ac_0 : \mathbb{P} \text{State} \bullet \right.
\left. \left( P^f[ac_0/ac'] \land a_0 \subseteq s \land \bot \notin ss \right) \right) \right) \}
\text{Predicate calculus}
\]
\[
= \left( \exists ac_0 : \mathbb{P} \text{State} \bullet \right.
\left. \left( P^f[ac_0/ac'] \lor \left( P^f[ac_0/ac'] \land s_0 \neq \emptyset \land \bot \notin ss \right) \right) \land a_0 \subseteq s \right) \}
\text{Predicate calculus}
\]

\textbf{Lemma C.1.4}

\[ (s, \{\bot\}) \in d2bmb(A(P)) \Leftrightarrow (s, \emptyset) \in d2bmb(A(P)) \]

\textbf{Proof.}

\[ (s, \{\bot\}) \in d2bmb(A(P)) \Leftrightarrow (s, \emptyset) \in d2bmb(A(P)) \]

\{Lemma C.1.5 and Lemma C.1.6\}

\[ = \text{true} \]

\[ \square \]
Lemma C.1.5

\((s, \{\bot\}) \in d2bmb(A(P)) = P^f[\emptyset/ac']\)

Proof.

\((s, \{\bot\}) \in d2bmb(A(P)) \quad \text{\{Lemma C.1.1\}}\)

\[
= (s, \{\bot\}) \in \left\{ \begin{array}{l}
s : \text{State}, ss : P\text{ State}_{\bot} \\
\exists ac_0 : P\text{ State} \bullet \\
(P^f[ac_0/ac'] \lor (P^f[ac_0/ac'] \land ss \neq \emptyset \land \bot \notin ss)) \\
\land \ ac_0 \subseteq ss \\
\end{array} \right\} \quad \text{\{Property of sets\}}
\]

\[
= \exists ac_0 : P\text{ State} \bullet (P^f[ac_0/ac'] \lor (P^f[ac_0/ac'] \land \bot \notin \{\bot\})) \land ac_0 \subseteq \{\bot\} \quad \text{\{Property of sets and predicate calculus\}}
\]

\[
= \exists ac_0 : P\text{ State} \bullet P^f[ac_0/ac'] \land ac_0 \subseteq \{\bot\} \quad \text{\{Case-analysis on } ac_0 \text{ and one-point rule\}}
\]

\[
= P^f[\emptyset/ac']
\]

\]

Lemma C.1.6

\((s, \emptyset) \in d2bmb(A(P)) = P^f[\emptyset/ac']\)

Proof.

\((s, \emptyset) \in d2bmb(A(P)) \quad \text{\{Definition of } d2bmb \text{ for } P \text{ that is } A\text{-healthy\}}\)

\[
= (s, \emptyset) \in \left\{ \begin{array}{l}
s : \text{State}, ss : P\text{ State}_{\bot} \\
\exists ac_0 : P\text{ State} \bullet \\
(P^f[ac_0/ac'] \lor (P^f[ac_0/ac'] \land ss \neq \emptyset \land \bot \notin ss)) \\
\land \ ac_0 \subseteq ss \\
\end{array} \right\} \quad \text{\{Property of sets\}}
\]

\[
= \exists ac_0 : P\text{ State} \bullet (P^f[ac_0/ac'] \lor (P^f[ac_0/ac'] \land \emptyset \neq \emptyset)) \land ac_0 \subseteq \emptyset \quad \text{\{Property of sets and predicate calculus\}}
\]

\[
= \exists ac_0 : P\text{ State} \bullet P^f[ac_0/ac'] \land ac_0 \subseteq \emptyset \quad \text{\{Property of sets and one-point rule\}}
\]

\[
= P^f[\emptyset/ac']
\]

\]

Revision: 704f887 (2014-02-04 11:14:10 +0000) 227
Lemma C.1.7

\((s, \emptyset) \in d2bmb(A(P)) \iff (s, \{\bot\}) \in d2bmb(A(P)) = \text{true}\)

Proof.

\((s, \emptyset) \in d2bmb(A(P)) \iff (s, \{\bot\}) \in d2bmb(A(P)) \quad \{\text{Lemma C.1.6 and Lemma C.1.5}\}

= \text{true}

\hfill \square

C.2 bmb2d

Lemma C.2.1

\(((s, ac') \in B \ ; ac \subseteq ac') \land (s, \emptyset) \notin B \iff ((s, ac') \in B \ ; ac \subseteq ac') \land ac' \neq \emptyset \land (s, \emptyset) \notin B\)

Proof.

\(((s, ac') \in B \ ; ac \subseteq ac') \land (s, \emptyset) \notin B \quad \{\text{Definition of sequential composition}\}

\iff \left( \exists ac_0 : \mathbb{P} \text{ State} \bullet (s, ac_0) \in B \land ac_0 \subseteq ac' \land (s, \emptyset) \notin B \right) \quad \{\text{Predicate calculus}\}

\iff \left( \exists ac_0 : \mathbb{P} \text{ State} \bullet (s, ac_0) \in B \land ac_0 \subseteq ac' \land ac' = \emptyset \land (s, \emptyset) \notin B \right) \quad \{\text{Predicate calculus}\}

\iff \left( \exists ac_0 : \mathbb{P} \text{ State} \bullet (s, ac_0) \in B \land ac_0 \subseteq ac' \land ac' \neq \emptyset \right) \land (s, \emptyset) \notin B \quad \{\text{Property of sets and case analysis on } ac_0\}

\iff \left( (s, \emptyset) \in B \land ac' = \emptyset \right) \lor \left( \exists ac_0 : \mathbb{P} \text{ State} \bullet (s, ac_0) \in B \land ac_0 \subseteq ac' \land ac' \neq \emptyset \right) \quad \{\text{Predicate calculus}\}
\( \exists ac_0 : \mathbb{P} \text{State} \bullet (s, ac_0) \in B \land ac_0 \subseteq ac' \land ac' \neq \emptyset \land (s, \emptyset) \notin B \) \hspace{1cm} \{\text{Definition of sequential composition}\}

\( \equiv ((s, ac') \in B ; ac \subseteq ac' \land ac' \neq \emptyset) \land (s, \emptyset) \notin B \)

Lemma C.2.2 Provided \( B \) satisfies \( \text{bmh}_{0,1,2} \).

\[ bmb2d(B) = \begin{pmatrix} \neg ((s, ac' \cup \{\bot\}) \in B ; ac \subseteq ac') \\ \vdash ((s, ac') \in B ; ac \subseteq ac') \land (s, \emptyset) \notin B \end{pmatrix} \]

Proof.

\[ bmb2d(B) = bmb2d(\text{bmh}_{0,1,2}(B)) \]

\[ = \begin{pmatrix} \neg ((s, \{\bot\}) \in B \land (s, \emptyset) \in B) \\ \land \neg \begin{pmatrix} ((s, ac' \cup \{\bot\}) \in B ; ac \subseteq ac') \\ \land ((s, \{\bot\}) \notin B \land (s, \emptyset) \notin B) \end{pmatrix} \vdash ((s, ac') \in B ; ac \subseteq ac') \land (s, \{\bot\}) \notin B \land (s, \emptyset) \notin B \end{pmatrix} \]

\[ \vdash ((s, \{\bot\}) \notin B \land (s, \emptyset) \notin B) \]

\[ \vdash ((s, ac') \in B ; ac \subseteq ac') \land (s, \{\bot\}) \notin B \land (s, \emptyset) \notin B \]

\[ \{\text{Predicate calculus}\} \]

\[ \vdash ((s, \{\bot\}) \notin B \land (s, \emptyset) \notin B) \]

\[ \vdash ((s, ac') \in B ; ac \subseteq ac') \land (s, \{\bot\}) \notin B \land (s, \emptyset) \notin B \]

\[ \{\text{Predicate calculus}\} \]
\[
\begin{align*}
\text{Lemma C.2.3} \\
bmb\! 2d(bmh_{0,1,2}(B)) \\
\end{align*}
\]
Proof.

\[
\begin{align*}
\neg ((s, \{ \bot \}) \in B \land (s, \emptyset) \in B) \\
\land \\
\neg (((s, ac' \cup \{ \bot \}) \in B \land ac \subseteq ac') \land (s, \{ \bot \}) \notin B \land (s, \emptyset) \notin B) \\
\Rightarrow \\
((s, ac') \in B \land ac \subseteq ac') \land (s, \{ \bot \}) \notin B \land (s, \emptyset) \notin B
\end{align*}
\]

\[
bmb2d(bmho_{1,2}(B)) = \text{ok} \Rightarrow \left\{(s, ac') \in bmho_{1,2}(B) \land \bot \notin ac' \land ok' \right\} \quad \text{[Definition of bmb2d]}
\]

\[
ok \Rightarrow \left\{\begin{array}{l}
(s, ac') \in \{ \exists ss_0 \bullet ((s, ss_0) \in B \lor (s, ss_0 \cup \{ \bot \}) \in B) \\
\land ((s, \{ \bot \}) \in B \equiv (s, \emptyset) \in B) \\
\land ss_0 \subseteq ss \land \bot \in ss_0 \Rightarrow \bot \in ss\}
\land \bot \notin ac'
\end{array}\right\} \quad \text{[Definition of bmho_{1,2}(B)]}
\]

\[
ok \Rightarrow \left\{\begin{array}{l}
(s, ac' \cup \{ \bot \}) \in \{ \exists ss_0 \bullet ((s, ss_0) \in B \lor (s, ss_0 \cup \{ \bot \}) \in B) \\
\land ((s, \{ \bot \}) \in B \equiv (s, \emptyset) \in B) \\
\land ss_0 \subseteq ss \land \bot \in ss_0 \Rightarrow \bot \in ss\}
\land \bot \notin ac'
\end{array}\right\} \quad \text{[Property of sets and predicate calculus]}
\]

\[
ok \Rightarrow \left\{\begin{array}{l}
\exists ss_0 \bullet ((s, ss_0) \in B \lor (s, ss_0 \cup \{ \bot \}) \in B) \\
\land ((s, \{ \bot \}) \in B \equiv (s, \emptyset) \in B) \\
\land ss_0 \subseteq ac' \land (\bot \in ss_0 \equiv \bot \in ac') \\
\land \bot \notin ac' \land ok'
\end{array}\right\} \quad \text{[Predicate calculus]}
\]
\[
\begin{align*}
\text{ok} & \Rightarrow \\
& \left( \exists \, ss_0 \bullet \left( (s, ss_0) \in B \lor (s, ss_0 \cup \{\bot\}) \in B \right) \\
& \quad \land \left( (s, \{\bot\}) \in B \iff (s, \emptyset) \in B \right) \\
& \quad \land \ ss_0 \subseteq ac' \land \bot \notin ss_0 \land \bot \notin ac' \land ok' \right)
\end{align*}
\]

\[
\begin{align*}
\text{Predicate calculus}
\end{align*}
\]

\[
\begin{align*}
\text{ok} & \Rightarrow \\
& \left( \exists \, ss_0 \bullet \left( (s, ss_0) \in B \lor (s, ss_0 \cup \{\bot\}) \in B \right) \\
& \quad \land \left( (s, \{\bot\}) \in B \iff (s, \emptyset) \in B \right) \\
& \quad \land \ ss_0 \subseteq (ac' \lor \{\bot\}) \land \bot \notin ss_0 \land \bot \notin ac' \right)
\end{align*}
\]

\[
\begin{align*}
\text{Predicate calculus}
\end{align*}
\]

\[
\begin{align*}
\text{ok} & \Rightarrow \\
& \left( \exists \, ss_0 \bullet \left( (s, ss_0) \in B \lor (s, ss_0 \cup \{\bot\}) \in B \right) \\
& \quad \land \left( (s, \{\bot\}) \in B \iff (s, \emptyset) \in B \right) \\
& \quad \land \ ss_0 \subseteq (ac' \cup \{\bot\}) \land \bot \in ss_0 \land \bot \notin ac' \right)
\end{align*}
\]

\[
\begin{align*}
\text{Property of sets}
\end{align*}
\]

\[
\begin{align*}
\text{Predicate calculus}
\end{align*}
\]

\[
\begin{align*}
\text{ok} & \Rightarrow \\
& \left( \exists \, ss_0 \bullet \left( (s, ss_0) \in B \lor (s, ss_0 \cup \{\bot\}) \in B \right) \\
& \quad \land \left( (s, \{\bot\}) \in B \iff (s, \emptyset) \in B \right) \\
& \quad \land \ ss_0 \subseteq (ac' \cup \{\bot\}) \land \bot \in ss_0 \land \bot \notin ac' \right)
\end{align*}
\]

\[
\begin{align*}
\text{Predicate calculus}
\end{align*}
\]

\[
\begin{align*}
\text{ok} & \Rightarrow \\
& \left( \exists \, ss_0 \bullet \left( (s, ss_0) \in B \lor (s, ss_0 \cup \{\bot\}) \in B \right) \\
& \quad \land \left( (s, \{\bot\}) \in B \iff (s, \emptyset) \in B \right) \\
& \quad \land \ ss_0 \subseteq (ac' \cup \{\bot\}) \land \bot \in ss_0 \land \bot \notin ac' \right)
\end{align*}
\]

\[
\begin{align*}
\text{Predicate calculus: introduce fresh variable}
\end{align*}
\]
\[
\begin{align*}
= \text{ok} & \Rightarrow \\
& \left( \begin{array}{c}
\exists s_{s_0} \cdot (s, ss_0) \in B \land ss_0 \subseteq \emptyset \land \not\in \emptyset \land \not\in ac' \land ok'
\end{array} \right) \\
& \left( \begin{array}{c}
\exists t, ss_0 \cdot (s, t) \in B \land t = ss_0 \cup \{\bot\} \land ss_0 \subseteq ac' \land \not\in \emptyset\land \not\in ac' \land ok'
\end{array} \right) \\
& \left( \begin{array}{c}
\exists ss_0 \cdot (s, ss_0) \in B \land ss_0 \subseteq (ac' \cup \{\bot\}) \land \not\in ss_0 \land \not\in ac'
\end{array} \right) \\
& \text{Lemma [3.3.2]} \\
\{\text{One-point rule and substitution}\}
\end{align*}
\]

\[
\begin{align*}
= \text{ok} & \Rightarrow \\
& \left( \begin{array}{c}
\exists s_{s_0} \cdot (s, ss_0) \in B \land ss_0 \subseteq \emptyset \land \not\in \emptyset \land \not\in ac' \land ok'
\end{array} \right) \\
& \left( \begin{array}{c}
\exists t, ss_0 \cdot (s, t) \in B \land t \setminus \{\bot\} = ss_0 \land ss_0 \subseteq ac' \land \not\in t \land \not\in ac' \land ok'
\end{array} \right) \\
& \left( \begin{array}{c}
\exists ss_0 \cdot (s, ss_0) \in B \land ss_0 \subseteq (ac' \cup \{\bot\}) \land \not\in ss_0 \land \not\in ac'
\end{array} \right) \\
& \text{Property of sets and variable renaming} \\
\end{align*}
\]

\[
\begin{align*}
= \text{ok} & \Rightarrow \\
& \left( \begin{array}{c}
\exists s_{s_0} \cdot (s, ss_0) \in B \land ss_0 \subseteq \emptyset \land \not\in \emptyset \land \not\in ac' \land ok'
\end{array} \right) \\
& \left( \begin{array}{c}
\exists t \cdot (s, t) \in B \land (t \setminus \{\bot\}) \subseteq ac' \land \not\in t \land \not\in ac' \land ok'
\end{array} \right) \\
& \left( \begin{array}{c}
\exists ss_0 \cdot (s, ss_0) \in B \land ss_0 \subseteq (ac' \cup \{\bot\}) \land \not\in ss_0 \land \not\in ac'
\end{array} \right) \\
& \text{Predicate calculus: absorption law}
\end{align*}
\]
\[ (s, \{ \bot \}) \in B \iff (s, \emptyset) \in B \]
\[ \Rightarrow \left( \begin{array}{l}
(\exists \, s_0 \bullet (s, s_0) \in B \land s_0 \subseteq ac' \land \bot \not\in s_0 \land \bot \not\in ac' \land ok' ) \\
(\exists \, s_0 \bullet (s, s_0) \in B \land s_0 \subseteq (ac' \cup \{ \bot \}) \land \bot \in s_0 \land \bot \not\in ac' \\
\end{array} \right) \]
\[ \{ \text{Predicate calculus} \} \]
\[ = ok \Rightarrow \left( \begin{array}{l}
((s, \{ \bot \}) \in B \iff (s, \emptyset) \in B) \\
\land \\
(\exists \, s_0 \bullet (s, s_0) \in B \land s_0 \subseteq ac' \land \bot \not\in s_0 \land \bot \not\in ac' \land ok' ) \\
(\exists \, s_0 \bullet (s, s_0 \cup \{ \bot \}) \in B \land s_0 \subseteq ac' \land \bot \not\in s_0 \land \bot \not\in ac' \\
\end{array} \right) \]
\[ \{ \text{Instantiation: consider case where } s_0 = \emptyset \} \]
\[ = ok \Rightarrow \left( \begin{array}{l}
(((s, \{ \bot \}) \in B \land (s, \emptyset) \in B) \lor ((s, \{ \bot \}) \not\in B \land (s, \emptyset) \not\in B) \\
\land \\
(\exists \, s_0 \bullet (s, s_0) \in B \land s_0 \subseteq ac' \land \bot \not\in s_0 \land \bot \not\in ac' \land ok' ) \\
(\exists \, s_0 \bullet (s, s_0 \cup \{ \bot \}) \in B \land s_0 \subseteq ac' \land \bot \not\in s_0 \land \bot \not\in ac' \\
\end{array} \right) \]
\[ \{ \text{Predicate calculus: distribution} \} \]
\[
= \text{ok} \Rightarrow \left\{ \begin{array}{l}
\left( \begin{array}{l}
(s, \emptyset) \\
\exists ss_0 \bullet (s, ss_0) \in B \land ss_0 \subseteq ac' \land \bot \notin ss_0 \\
\land \bot \notin ac' \land ok' \land (s, \{\bot\}) \in B \land (s, \emptyset) \in B
\end{array} \right)
\end{array} \right.
\]

\[
= \text{ok} \Rightarrow \left\{ \begin{array}{l}
\left( \begin{array}{l}
(s, \emptyset) \\
\exists ss_0 \bullet (s, ss_0) \in B \land ss_0 \subseteq ac' \land \bot \notin ss_0 \\
\land \bot \notin ac' \land ok' \land (s, \{\bot\}) \in B \land (s, \emptyset) \in B
\end{array} \right)
\end{array} \right.
\]

\[
\left\{ \text{Predicate calculus: absorption law} \right\}
\]

\[
\left( \begin{array}{l}
\bot \notin ac' \land ok' \land (s, \{\bot\}) \in B \land (s, \emptyset) \in B
\end{array} \right)
\]

\[
= \text{ok} \Rightarrow \left\{ \begin{array}{l}
\left( \begin{array}{l}
(s, \emptyset) \\
\exists ss_0 \bullet (s, ss_0) \in B \land ss_0 \subseteq ac' \land \bot \notin ss_0 \\
\land \bot \notin ac' \land ok' \land (s, \{\bot\}) \notin B \land (s, \emptyset) \notin B
\end{array} \right)
\end{array} \right.
\]

\[
\left\{ \text{Predicate calculus: absorption law} \right\}
\]

\[
\left( \begin{array}{l}
\bot \notin ac' \land (s, \{\bot\}) \in B \land (s, \emptyset) \in B
\end{array} \right)
\]

\[
= \text{ok} \Rightarrow \left\{ \begin{array}{l}
\left( \begin{array}{l}
(s, \emptyset) \\
\exists ss_0 \bullet (s, ss_0) \in B \land ss_0 \subseteq ac' \land \bot \notin ss_0 \\
\land \bot \notin ac' \land (s, \{\bot\}) \notin B \land (s, \emptyset) \notin B
\end{array} \right)
\end{array} \right.
\]

\[
\left\{ \text{Predicate calculus: absorption law} \right\}
\]

\[
\left( \begin{array}{l}
\bot \notin ac' \land (s, \{\bot\}) \in B \land (s, \emptyset) \in B
\end{array} \right)
\]

\[
= \text{ok} \Rightarrow \left\{ \begin{array}{l}
\left( \begin{array}{l}
(s, \emptyset) \\
\exists ss_0 \bullet (s, ss_0) \in B \land ss_0 \subseteq ac' \land \bot \notin ss_0 \\
\land \bot \notin ac' \land (s, \{\bot\}) \notin B \land (s, \emptyset) \notin B
\end{array} \right)
\end{array} \right.
\]

\[
\left\{ \text{Predicate calculus: absorption law} \right\}
\]

\[
\left( \begin{array}{l}
\bot \notin ac' \land (s, \{\bot\}) \in B \land (s, \emptyset) \in B
\end{array} \right)
\]

\[
= \text{ok} \Rightarrow \left\{ \begin{array}{l}
\left( \begin{array}{l}
(s, \emptyset) \\
\exists ss_0 \bullet (s, ss_0) \in B \land ss_0 \subseteq ac' \land \bot \notin ss_0 \\
\land \bot \notin ac' \land (s, \{\bot\}) \notin B \land (s, \emptyset) \notin B
\end{array} \right)
\end{array} \right.
\]

\[
\left\{ \text{Predicate calculus: absorption law} \right\}
\]

\[
\left( \begin{array}{l}
\bot \notin ac' \land (s, \{\bot\}) \in B \land (s, \emptyset) \in B
\end{array} \right)
\]
\[
\begin{align*}
\Rightarrow & \quad \left( \perp \notin ac' \land ok' \land (s, \{\perp\}) \in B \land (s, \emptyset) \notin B \right) \\
\Rightarrow & \quad \left( \exists ss_0 \bullet (s, ss_0) \in B \land ss_0 \subseteq ac' \land \perp \notin ss_0 \right. \\
& \quad \left. \land (s, \{\perp\}) \notin B \land (s, \emptyset) \notin B \right) \\
\Rightarrow & \quad \left( \perp \notin ac' \land (s, \{\perp\}) \in B \land (s, \emptyset) \notin B \right) \\
\Rightarrow & \quad \left( \exists ss_0 \bullet (s, ss_0 \cup \{\perp\}) \in B \land ss_0 \subseteq ac' \land \perp \notin ss_0 \right. \\
& \quad \left. \land (s, \{\perp\}) \notin B \land (s, \emptyset) \notin B \right) \\
\Rightarrow & \quad \left( \exists ss_0 \bullet (s, ss_0) \in B \land ss_0 \subseteq ac' \land \perp \notin ss_0 \right. \\
& \quad \left. \land (s, \{\perp\}) \notin B \land (s, \emptyset) \notin B \right)
\end{align*}
\]
\[
\begin{align*}
\{\text{Definition of sequential composition and type of } ac': \bot \notin ac'\} \\
\Rightarrow &\nonumber \\exists ss_0 \bullet (((s, ac') \in B \land \bot \notin ac')[ss_0/ac'] \land (ac \subseteq ac')[ss_0/ac] \\
&\wedge \bot \notin ac' \\
&\wedge (s, \{\bot\}) \notin B \land (s, \emptyset) \notin B \\
&\{\text{Definition of design}\}
\end{align*}
\]

Lemma C.2.4

\[ (s, \{s_1 : \text{State}_\bot | \text{true}\}) \in d2bmb(P) = P^f[\{s_1 : \text{State} | \text{true}\}/ac'] \]

Proof.

\[ (s, \{s_1 : \text{State}_\bot | \text{true}\}) \in d2bmb(P) \quad \{\text{Definition of } d2bmb\} \]
= \left( s, \{s_1 : \text{State}_\bot \ | \ \text{true}\} \right) \in \left\{ s : \text{State}, ss : \mathbb{P} \text{State}_\bot \right. \\
\left. \begin{array}{l}
(\neg P^l \Rightarrow P^t)[ss/ac'] \land \bot \notin ss \\
\lor \\
(P^f[ss \setminus \{\bot\}] / ac' \land \bot \in ss)
\end{array} \right\} \quad \left\{ \text{Property of sets} \right\}

= \left( (\neg P^f \Rightarrow P^t)[ss/ac'][\{s_1 : \text{State}_\bot \ | \ \text{true}\} / ss] \\
\land \bot \notin \{s_1 : \text{State}_\bot \ | \ \text{true}\} \right) \\
\lor \\
(P^f[ss \setminus \{\bot\}] / ac'[\{s_1 : \text{State}_\bot \ | \ \text{true}\} / ss] \\
\land \bot \in \{s_1 : \text{State}_\bot \ | \ \text{true}\} \right) \\
\quad \left\{ \text{Property of sets and propositional calculus} \right\}

= P^f[ss \setminus \{\bot\} / ac'][\{s_1 : \text{State}_\bot \ | \ \text{true}\} / ss] \quad \left\{ \text{Substitution} \right\}

= P^f[\{s_1 : \text{State}_\bot \ | \ \text{true}\} \setminus \{\bot\} / ac'] \quad \left\{ \text{Property of sets} \right\}

= P^f[\{s_1 : \text{State} \ | \ \text{true}\} / ac']

\square

Lemma C.2.5 Provided $\bot \notin ac'$.

$\{s : \text{State} \mid (s, ac' \cup \{\bot\}) \in d2bmb(P)\} = \{s : \text{State} \mid P^f\}$

Proof.

$\{s : \text{State} \mid (s, ac' \cup \{\bot\}) \in d2bmb(P)\} \quad \left\{ \text{Definition of d2bmb} \right\}

= \left\{ s : \text{State} \mid (s, ac' \cup \{\bot\}) \in \left( s : \text{State}, ss : \mathbb{P} \text{State}_\bot \\
\left. \begin{array}{l}
(\neg P^f \Rightarrow P^t)[ss/ac'] \land \bot \notin ss \\
\lor \\
(P^f[ss \setminus \{\bot\}] / ac' \land \bot \in ss)
\end{array} \right\} \right\} \quad \left\{ \text{Property of sets} \right\}

= \left\{ s : \text{State} \\
\left. \begin{array}{l}
(\neg P^f \Rightarrow P^t)[ss/ac'][ac' \cup \{\bot\} / ss] \land \bot \notin (ac' \cup \{\bot\}) \\
\lor \\
(P^f[ss \setminus \{\bot\}] / ac'[ac' \cup \{\bot\} / ss] \land \bot \in (ac' \cup \{\bot\})
\end{array} \right\} \right\} \quad \left\{ \text{Property of sets} \right\}

= \{ s : \text{State} \mid (P^f[ss \setminus \{\bot\} / ac'][ac' \cup \{\bot\} / ss]) \} \quad \left\{ \text{Substitution} \right\}

= \{ s : \text{State} \mid (P^f[ac' \cup \{\bot\} \setminus \{\bot\} / ac']) \} \quad \left\{ \text{Property of sets, and assumption that } \bot \notin ac' \right\}

= \{ s : \text{State} \mid P^f \}$
Lemma C.2.6  Provided $\bot \not\in ac'$.

$$\{ s : State \mid (s, ac') \in d2bmb(P) \} = \{ s : State \mid (\neg P^f \Rightarrow P^t) \}$$

Proof.

$$\{ s : State \mid (s, ac') \in d2bmb(P) \} = \{ s : State \mid (\neg P^f \Rightarrow P^t)[ss/ac'] \land \bot \not\in ss \} \lor \{ P^f[ss \setminus \{ \bot \}] / ac' \land \bot \in ss \}$$

Property of sets

$$\{ s : State \mid (\neg P^f \Rightarrow P^t)[ss/ac'] \land \bot \not\in ac' \} \lor \{ P^f[ss \setminus \{ \bot \}] / ac' \land \bot \in ac' \}$$

Substitution

$$\{ s : State \mid (\neg P^f \Rightarrow P^t) \land \bot \not\in ac' \} \lor \{ P^f[ac' \setminus \{ \bot \}] / ac' \land \bot \in ac' \}$$

Assumption: $\bot \not\in ac'$

$$\{ s : State \mid (\neg P^f \Rightarrow P^t) \}$$

□

Lemma C.2.7

$$(s, \{ s : State \mid (s, ac' \cup \{ \bot \}) \in d2bmb(P) \}) \in d2bmb(Q)$$

$$= (\neg Q^f \Rightarrow Q^t)[\{ s : State \mid P^f \}/ac']$$

Proof.

$$(s, \{ s : State \mid (s, ac' \cup \{ \bot \}) \in d2bmb(P) \}) \in d2bmb(Q)$$

Lemma [C.2.5]

$$= (s, \{ s : State \mid P^f \}) \in d2bmb(Q)$$

Definition of $d2bmb$

Revision: 704f887 (2014-02-04 11:14:10 +0000) 239
\[
= (s, \{ s : \text{State} \mid P^f \}) \in \left\{ \begin{array}{c}
  s : \text{State}, ss : \mathbb{P} \text{State}_\perp \\
  \left( \neg Q^f \Rightarrow Q^i \right)[ss/ac'] \land \perp \notin ss \\
  \lor \\
  (Q^f[ss \setminus \{ \perp \}]/ac') \land \perp \in ss
\end{array} \right\}
\text{Property of sets}
\]

\[
= \left( \left( \neg Q^f \Rightarrow Q^i \right)[ss/ac'][\{ s : \text{State} \mid P^f \}]/ss \right) \land \perp \notin \{ s : \text{State} \mid P^f \} \\
\lor \\
(Q^f[ss \setminus \{ \perp \}]/ac')[\{ s : \text{State} \mid P^f \}]/ss \land \perp \in \{ s : \text{State} \mid P^f \}
\right)
\text{Property of sets}
\]

\[
= (\neg Q^f \Rightarrow Q^i)[\{ s : \text{State} \mid (\neg P^f \Rightarrow P^i) \}]/ac'
\]

\[
\square
\]

**Lemma C.2.8**

\[
(s, \{ s : \text{State} \mid (s, ac') \in d2bmb(P) \}) \in d2bmb(Q)
\]

\[
= (\neg Q^f \Rightarrow Q^i)[\{ s : \text{State} \mid (\neg P^f \Rightarrow P^i) \}]/ac'
\]

**Proof.**

\[
(s, \{ s : \text{State} \mid (s, ac') \in d2bmb(P) \}) \in d2bmb(Q) \quad \text{Lemma C.2.6}
\]

\[
= (s, \{ s : \text{State} \mid (\neg P^f \Rightarrow P^i) \}) \in d2bmb(Q) \quad \text{Definition of d2bmb}
\]

\[
= (s, \{ s : \text{State} \mid (\neg P^f \Rightarrow P^i) \}) \in \left\{ \begin{array}{c}
  s : \text{State}, ss : \mathbb{P} \text{State}_\perp \\
  \left( \neg Q^f \Rightarrow Q^i \right)[ss/ac'] \land \perp \notin ss \\
  \lor \\
  (Q^f[ss \setminus \{ \perp \}]/ac') \land \perp \in ss
\end{array} \right\}
\text{Property of sets}
\]

\[
= \left( \left( \neg Q^f \Rightarrow Q^i \right)[ss/ac'][\{ s : \text{State} \mid (\neg P^f \Rightarrow P^i) \}]/ss \right) \\
\lor \\
(Q^f[ss \setminus \{ \perp \}]/ac')[\{ s : \text{State} \mid (\neg P^f \Rightarrow P^i) \}]/ss \land \perp \in \{ s : \text{State} \mid (\neg P^f \Rightarrow P^i) \}
\right)
\text{Property of sets and propositional calculus}
\]

\[
= (\neg Q^f \Rightarrow Q^i)[ss/ac'][\{ s : \text{State} \mid (\neg P^f \Rightarrow P^i) \}]/ss \\
\lor \\
(\neg Q^f \Rightarrow Q^i)[\{ s : \text{State} \mid (\neg P^f \Rightarrow P^i) \}]/ac'
\text{Substitution}
\]

\[
= (\neg Q^f \Rightarrow Q^i)[\{ s : \text{State} \mid (\neg P^f \Rightarrow P^i) \}]/ac'
\]
Lemma C.2.9

\[ bmb2d(B_0 ; B_1) = \]

\[ \begin{align*}
& \text{ok} \Rightarrow \left( \left( (s, \{ s_1 : State \mid (s_1, ac') \in B_1 \}) \in B_0 \land \perp \notin ac' \land ok' \right) \\
& \lor \left( (s, \{ s_1 : State_{\perp} \mid true \}) \in B_0 \land \perp \notin ac' \right) \\
& \lor \left( (s, \{ s_1 : State \mid (s_1, ac' \cup \{ \perp \}) \in B_1 \}) \in B_0 \land \perp \notin ac' \right) \right)
\end{align*} \]

Proof.

\[ bmb2d(B_0 ; B_1) \]

\[ = \text{ok} \Rightarrow \left( \left( (s, ac') \in (B_0 ; B_1) \land \perp \notin ac' \land ok' \right) \\
\lor \left( (s, ac' \cup \{ \perp \}) \in (B_0 ; B_1) \land \perp \notin ac' \right) \right) \]

\[ = \text{ok} \Rightarrow \left( \left( (s, ac') \in \left( \{ s : State, ss : \mathbb{P} State_{\perp} \mid (s, \{ s_1 : State_{\perp} \mid true \}) \in B_0 \} \right) \\
\lor \left( (s, ac') \in \left( \{ s : State, ss : \mathbb{P} State_{\perp} \mid (s, \{ s_1 : State \mid (s_1, ss) \in B_1 \}) \in B_0 \} \right) \right) \right) \]

\[ = \text{ok} \Rightarrow \left( \left( (s, ac') \in \left( \{ s : State, ss : \mathbb{P} State_{\perp} \mid (s, \{ s_1 : State_{\perp} \mid true \}) \in B_0 \} \right) \right) \]

\[ \lor \left( (s, ac') \in \left( \{ s : State, ss : \mathbb{P} State_{\perp} \mid (s, \{ s_1 : State \mid (s_1, ss) \in B_1 \}) \in B_0 \} \right) \right) \]

\[ = \text{ok} \Rightarrow \left( \left( (s, ac') \in \left( \{ s : State, ss : \mathbb{P} State_{\perp} \mid (s, \{ s_1 : State_{\perp} \mid true \}) \in B_0 \} \right) \right) \]

\[ \lor \left( (s, ac') \in \left( \{ s : State, ss : \mathbb{P} State_{\perp} \mid (s, \{ s_1 : State \mid (s_1, ss) \in B_1 \}) \in B_0 \} \right) \right) \]

\[ \lor \left( (s, ac') \in \left( \{ s : State_{\perp} \mid true \}) \in B_0 \land \perp \notin ac' \land ok' \right) \]

\[ \lor \left( (s, \{ s_1 : State \mid (s_1, ac') \in B_1 \}) \in B_0 \land \perp \notin ac' \land ok' \right) \]

\[ \lor \left( (s, \{ s_1 : State_{\perp} \mid true \}) \in B_0 \land \perp \notin ac' \right) \]

\[ \lor \left( (s, \{ s_1 : State \mid (s_1, ac' \cup \{ \perp \}) \in B_1 \}) \in B_0 \land \perp \notin ac' \right) \]

\{Propositional calculus: absorption law\}
C.3 Other lemmas

Lemma C.3.1

\[ ([\exists ac' \cdot P^f] = \neg P^f] \iff ([\exists ac' \cdot \neg P^f] = \neg P^f] \]

Proof.

\[
[\exists ac' \cdot \neg P^f] = \neg P^f] \quad \{\text{Universal quantification}\}
\]

\[
\iff (\forall ok, ok', ac', s \cdot (\exists ac' \cdot \neg P^f) \Rightarrow \neg P^f) \quad \{\text{Predicate calculus}\}
\]

\[
\iff \forall ok, ok', ac', s \cdot (\exists ac' \cdot \neg P^f) \Rightarrow (\exists ac' \cdot \neg P^f)) \quad \{\text{Predicate calculus}\}
\]

\[
\iff \forall ok, s \cdot (\exists ac' \cdot \neg P^f) \Rightarrow (\forall ac' \cdot \neg P^f) \quad \{\text{Predicate calculus}\}
\]

\[
\iff \forall ok, s \cdot (\exists ac' \cdot P^f) \Rightarrow (\forall ac' \cdot P^f) \quad \{\text{Predicate calculus}\}
\]

\[
\iff \forall ok, s, ac', ok' \cdot (\exists ac' \cdot P^f) \Rightarrow P^f \quad \{\text{Predicate calculus}\}
\]

\[
\iff (\forall ok, s, ac', ok' \cdot (\exists ac' \cdot P^f) \Rightarrow P^f) \quad \{\text{Universal quantification}\}
\]

\[= ([\exists ac' \cdot P^f] = P^f] \]

\[ \square \]

Lemma C.3.2

Provided \( B_0 \) and \( B_1 \) are of type \( BM_{\bot} \).

\[
\left[ (s, ac') \in B_1 \Rightarrow (s, ac') \in B_0 \right] \land \left[ (s, ac' \cup \{ \bot \}) \in B_1 \Rightarrow (s, ac' \cup \{ \bot \}) \in B_0 \right] \iff B_1 \subseteq B_0
\]
Proof.

\[ B_1 \subseteq B_0 \quad \{ \text{Definition of subset inclusion} \} \]
\[ \iff \forall s : \text{State}, ss : \mathbb{P} \text{State}_{\bot} \bullet (s, ss) \in B_1 \Rightarrow (s, ss) \in B_0 \quad \{ \text{Predicate calculus} \} \]
\[ \iff \left( \forall s : \text{State}, ss : \mathbb{P} \text{State}_{\bot} \bullet \right. \]
\[ \left. \left( ((s, ss) \in B_1 \Rightarrow (s, ss) \in B_0) \land (\bot \in ss \lor \bot \notin ss) \right) \quad \{ \text{Predicate calculus} \} \right) \]
\[ \iff \left( \forall s : \text{State}, ss : \mathbb{P} \text{State}_{\bot} \bullet \right. \]
\[ \left. \left( \bot \in ss \Rightarrow ((s, ss) \in B_1 \Rightarrow (s, ss) \in B_0) \right) \right) \quad \{ \text{Predicate calculus} \} \]
\[ \land \left( \forall s : \text{State}, ss : \mathbb{P} \text{State}_{\bot} \bullet \right. \]
\[ \left. \left( \bot \notin ss \Rightarrow ((s, ss) \in B_1 \Rightarrow (s, ss) \in B_0) \right) \right) \quad \{ \text{Predicate calculus} \} \]
\[ \iff \left( \forall s : \text{State}, ss : \mathbb{P} \text{State}_{\bot} \bullet \right. \]
\[ \left. \left( \exists t : \text{State}, ss : \mathbb{P} \text{State} \bullet \bot \in t \cup \{ \bot \} \land \bot \in ss \Rightarrow ((s, ss) \in B_1 \Rightarrow (s, ss) \in B_0) \right) \right) \]
\[ \land \left( \forall s : \text{State}, ss : \mathbb{P} \text{State}_{\bot} \bullet \right. \]
\[ \left. \left( \bot \notin ss \Rightarrow ((s, ss) \in B_1 \Rightarrow (s, ss) \in B_0) \right) \right) \quad \{ \text{Lemma B.3.2} \} \]
\[ \iff \left( \forall s : \text{State}, ss : \mathbb{P} \text{State}_{\bot} \bullet \right. \]
\[ \left. \left( \exists t : \text{State}, ss : \mathbb{P} \text{State} \bullet \bot \notin t \land t \cup \{ \bot \} = ss \Rightarrow ((s, ss) \in B_1 \Rightarrow (s, ss) \in B_0) \right) \right) \]
\[ \land \left( \forall s : \text{State}, ss : \mathbb{P} \text{State}_{\bot} \bullet \right. \]
\[ \left. \left( \bot \notin ss \Rightarrow ((s, ss) \in B_1 \Rightarrow (s, ss) \in B_0) \right) \right) \quad \{ \text{Type: } \bot \notin t \} \]
∀ s : State, ss : \( \mathbb{P} \) State \( \perp \) • \\
(∃ t : State, ss : \( \mathbb{P} \) State • \( t \cup \{ \perp \} = ss \))
⇒ \\
((s, ss) ∈ B_1 ⇒ (s, ss) ∈ B_0)
∧ \\
∀ s : State, ss : \( \mathbb{P} \) State \( \perp \) • \\
(\( s \notin ss \) ⇒ ((s, ss) ∈ B_1 ⇒ (s, ss) ∈ B_0))
{Predicate calculus}

∀ s : State, ss : \( \mathbb{P} \) State \( \perp \) • \\
(\( t \cup \{ \perp \} = ss \))
⇒ \\
((s, ss) ∈ B_1 ⇒ (s, ss) ∈ B_0)
∧ \\
∀ s : State, ss : \( \mathbb{P} \) State \( \perp \) • \\
(\( s \notin ss \) ⇒ ((s, ss) ∈ B_1 ⇒ (s, ss) ∈ B_0))
{Predicate calculus: one-point rule}

∀ s : State, t : \( \mathbb{P} \) State • \\
((s, t \cup \{ \perp \}) ∈ B_1 ⇒ (s, t \cup \{ \perp \}) ∈ B_0)
∧ \\
∀ s : State, ss : \( \mathbb{P} \) State • \\
((s, ss) ∈ B_1 ⇒ (s, ss) ∈ B_0)
{Variable renaming and predicate calculus}

∀ s : State, ac' : \( \mathbb{P} \) State • \\
((s, ac' \cup \{ \perp \}) ∈ B_1 ⇒ (s, ac' \cup \{ \perp \}) ∈ B_0)
∧ \\
((s, ac') ∈ B_1 ⇒ (s, ac') ∈ B_0)
{Universal quantification}

∀ s : State, ac' : \( \mathbb{P} \) State • \\
((s, ac' \cup \{ \perp \}) ∈ B_1 ⇒ (s, ac' \cup \{ \perp \}) ∈ B_0)
∧ \\
((s, ac') ∈ B_1 ⇒ (s, ac') ∈ B_0)
Appendix D

PBMH

D.1 Properties

Law D.1.1 (PBMH-idempotent)

\[ \text{PBMH} \circ \text{PBMH}(P) = \text{PBMH}(P) \]

Proof.

\[
\begin{align*}
\text{PBMH} \circ \text{PBMH}(P) & \quad \text{(Definition of PBMH)} \\
= & \text{PBMH}(P ; ac \subseteq ac') \quad \text{(Definition of PBMH)} \\
= & ((P ; ac \subseteq ac') ; ac \subseteq ac') \quad \text{(Associativity of sequential composition)} \\
= & (P ; (ac \subseteq ac' ; ac \subseteq ac')) \quad \text{(Definition of sequential composition)} \\
= & (P ; (\exists ac_0 \bullet ac \subseteq ac_0 \land ac_0 \subseteq ac')) \quad \text{(Transitivity of subset inclusion)} \\
= & (P ; ac \subseteq ac') \quad \text{(Definition of PBMH)} \\
= & \text{PBMH}(P)
\end{align*}
\]

\[\square\]

Lemma D.1.1

\[ \text{PBMH}(P) = \exists ac_0 \bullet P[ac_0/ac'] \land ac_0 \subseteq ac' \]

Proof.

\[
\begin{align*}
\text{PBMH}(P) & \quad \text{(Definition of PBMH)} \\
= & P ; ac \subseteq ac' \quad \text{(Definition of sequential composition)} \\
= & \exists ac_0 \bullet P[ac_0/ac'] \land ac_0 \subseteq ac'
\end{align*}
\]
D.2 Distribution properties

Law D.2.1 (PBMH-distribute-disjunction)

\[ \text{PBMH}(P \lor Q) = \text{PBMH}(P) \lor \text{PBMH}(Q) \]

Proof.

\[
\begin{align*}
\text{PBMH}(P \lor Q) & \quad \{\text{Definition of PBMH}\} \\
= (P \lor Q) \ ; \ ac \subseteq ac' & \quad \{\text{Definition of sequential composition}\} \\
= \exists ac_0 \bullet (P[ac_0/ac'] \lor Q[ac_0/ac']) \land ac_0 \subseteq ac' & \quad \{\text{Predicate calculus}\} \\
= \exists ac_0 \bullet (P[ac_0/ac'] \land ac_0 \subseteq ac') \lor (Q[ac_0/ac'] \land ac_0 \subseteq ac') & \quad \{\text{Predicate calculus}\} \\
= \exists ac_0 \bullet (P[ac_0/ac'] \land ac_0 \subseteq ac') \lor \exists ac_0 \bullet (Q[ac_0/ac'] \land ac_0 \subseteq ac') & \quad \{\text{Definition of sequential composition}\} \\
= (P \ ; \ ac \subseteq ac') \lor (Q \ ; \ ac \subseteq ac') & \quad \{\text{Definition of PBMH}\} \\
= \text{PBMH}(P) \lor \text{PBMH}(Q)
\end{align*}
\]

Law D.2.2 (PBMH-distribute-conjunction) Provided \( P \) and \( Q \) satisfy PBMH.

\[ \text{PBMH}(P \land Q) = \text{PBMH}(P) \land \text{PBMH}(Q) \]

Proof.

\[
\begin{align*}
\text{PBMH}(P \land Q) & \quad \{\text{Assumption: } P \text{ and } Q \text{ satisfy PBMH}\} \\
= \text{PBMH}((\text{PBMH}(P) \land \text{PBMH}(Q))) & \quad \{\text{Definition of PBMH}\} \\
= ((P \ ; \ ac \subseteq ac') \land (Q \ ; \ ac \subseteq ac')) \ ; \ ac \subseteq ac' & \quad \{\text{Definition of sequential composition}\} \\
= \exists ac_0 \bullet \left( \exists ac_1 \bullet (P[ac_1/ac'] \land ac_1 \subseteq ac') \right) \land \left( \exists ac_2 \bullet (Q[ac_2/ac'] \land ac_2 \subseteq ac') \right) \ [ac_0/ac'] \land ac_0 \subseteq ac' & \quad \{\text{Substitution}\}
\end{align*}
\]
\[
\exists \text{ac}_0 \cdot 
\left( \exists \text{ac}_1 \cdot (P[\text{ac}_1/\text{ac}'] \land \text{ac}_1 \subseteq \text{ac}_0) \right) \\
\land \\
\left( \exists \text{ac}_2 \cdot (Q[\text{ac}_2/\text{ac}'] \land \text{ac}_2 \subseteq \text{ac}_0) \right)
\]
\[\land \text{ac}_0 \subseteq \text{ac}'\]

\{Transitivity of subset inclusion\}

\[
= \left( \exists \text{ac}_1 \cdot (P[\text{ac}_1/\text{ac}'] \land \text{ac}_1 \subseteq \text{ac}') \right) \\
\land \\
\left( \exists \text{ac}_2 \cdot (Q[\text{ac}_2/\text{ac}'] \land \text{ac}_2 \subseteq \text{ac}') \right)
\]
\{Definition of sequential composition\}

\[
= (P ; \text{ac} \subseteq \text{ac}') \land (Q ; \text{ac} \subseteq \text{ac}')
\]
\{Definition of PBMH\}

\[
= \text{PBMH}(P) \land \text{PBMH}(Q)
\]

\[
\square
\]

D.3 Closure properties

**Law D.3.1 (PB\textit{MH}-disjunction-closure)** *Provided P and Q satisfy PB\textit{MH}.*

\[
\text{PBMH}(P \lor Q) = P \lor Q
\]

*Proof.*

\[
\text{PBMH}(P \lor Q) \quad \{\text{Law D.2.1}\}
\]

\[
= \text{PBMH}(P) \lor \text{PBMH}(Q) \quad \{\text{Assumption: } P \text{ and } Q \text{ satisfy PBMH}\}
\]

\[
= P \lor Q
\]

\[
\square
\]

**Law D.3.2 (PB\textit{MH}-conjunction-closure)** *Provided P and Q satisfy PB\textit{MH}.*

\[
\text{PBMH}(P \land Q) = P \land Q
\]

*Proof.*

\[
\text{PBMH}(P \land Q) \quad \{\text{Assumption: } P \text{ and } Q \text{ satisfy PBMH and Law D.2.2}\}
\]

\[
= \text{PBMH}(P) \land \text{PBMH}(Q) \quad \{\text{Assumption: } P \text{ and } Q \text{ satisfy PBMH}\}
\]

\[
= P \land Q
\]

\[
\square
\]
D.4 Lemmas

Lemma D.4.1

\[ \text{PBMH}(true) = true \]

Proof.

\[ \text{PBMH}(true) \quad \{\text{Definition of PBMH}\} \]
\[ = true ; ac \subseteq ac' \quad \{\text{Definition of sequential composition}\} \]
\[ = \exists ac_0 \bullet true[ac_0/ac'] \land ac_0 \subseteq ac' \]
\[ \{\text{Property of substitution and predicate calculus}\} \]
\[ = true \]

\[ \square \]

Lemma D.4.2

\[ \text{PBMH}(false) = false \]

Proof.

\[ \text{PBMH}(false) \quad \{\text{Definition of PBMH}\} \]
\[ = false ; ac \subseteq ac' \quad \{\text{Definition of sequential composition}\} \]
\[ = \exists ac_0 \bullet false[ac_0/ac'] \land ac_0 \subseteq ac' \]
\[ \{\text{Property of substitution and predicate calculus}\} \]
\[ = false \]

\[ \square \]

Lemma D.4.3

\[ \text{PBMH}(s \in ac') = s \in ac' \]

Proof.

\[ \text{PBMH}(s \in ac') \quad \{\text{Definition of PBMH}\} \]
\[ = s \in ac' ; ac \subseteq ac' \quad \{\text{Definition of sequential composition}\} \]
\[ = \exists ac_0 \bullet s \in ac_0 \land ac_0 \subseteq ac' \]
\[ \{\text{Property of sets}\} \]
\[ = s \in ac' \]

\[ \square \]
Lemma D.4.4

\[(P \land ac' \neq \emptyset) \; ; \; \mathcal{A} \; (Q \land ac' \neq \emptyset)\]

\[(P \land ac' \neq \emptyset) \; ; \; \mathcal{A} \; (Q \land ac' \neq \emptyset)) \land ac' \neq \emptyset\]

Proof.

\[(P \land ac' \neq \emptyset) \; ; \; \mathcal{A} \; (Q \land ac' \neq \emptyset)\]

\[= (P \land ac' \neq \emptyset)[\{z \mid Q \land ac' \neq \emptyset\}/ac']\]  \{Definition of \; ; \; \mathcal{A}\}

\[= (P \land ac' \neq \emptyset)[\{z \mid Q[z/s] \land ac' \neq \emptyset\}/ac']\]  \{Substitution\}

\[= \left(\left(P[\{z \mid Q[z/s] \land ac' \neq \emptyset\}/ac']\right)\right)\]  \{Propositional calculus\}

\[= \left(\left(P[\{z \mid Q[z/s] \land ac' \neq \emptyset\}/ac']\right)\right)\]  \{Property of sets\}

\[= \left(\left(P[\{z \mid Q[z/s] \land ac' \neq \emptyset\}/ac']\right)\right)\]  \{Predicate calculus: quantifier scope and duplicate term\}

\[= \left(\left(P[\{z \mid Q[z/s] \land ac' \neq \emptyset\}/ac']\right)\right)\]  \{Property of sets\}

\[= \left(\left(P[\{z \mid Q[z/s] \land ac' \neq \emptyset\}/ac']\right)\right)\]  \{Re-introduce ac' and substitution\}

\[= (\left((P \land ac' \neq \emptyset)[\{z \mid Q[z/s] \land ac' \neq \emptyset\}/ac']\right)) \land ac' \neq \emptyset\]  \{Substitution\}

\[= (\left((P \land ac' \neq \emptyset)[\{z \mid (Q \land ac' \neq \emptyset)[z/s]/ac']\right)) \land ac' \neq \emptyset\]  \{Definition of \; ; \; \mathcal{A}\}

\[= (\left((P \land ac' \neq \emptyset) \; ; \; \mathcal{A} \; (Q \land ac' \neq \emptyset)) \land ac' \neq \emptyset\]

□
Lemma D.4.5

\[ \text{PBMH}(ac' \neq \emptyset) = ac' \neq \emptyset \]

Proof.

\[ \text{PBMH}(ac' \neq \emptyset) \]
\[ = ac' \neq \emptyset \land ac \subseteq ac' \]
\[ = \exists ac_0 \bullet ac_0 \neq \emptyset \land ac_0 \subseteq ac' \]
\[ = ac' \neq \emptyset \]

\[ \square \]

Lemma D.4.6  Provided \( ac' \) is not free in \( P \).

\[ \text{PBMH}(P) = P \]

Proof.

\[ \text{PBMH}(P) \]
\[ = P \land \exists ac_0 \bullet ac_0 \subseteq ac' \]
\[ = P \]

\[ \square \]

Lemma D.4.7

\[ P \Rightarrow \text{PBMH}(P) \]

Proof.

\[ P \]
\[ = \exists ac_0 \bullet P[ac_0/ac'] \land ac_0 = ac' \]
\[ \Rightarrow \exists ac_0 \bullet P[ac_0/ac'] \land ac_0 \subseteq ac' \]
\[ = P \land \exists ac_0 \bullet ac_0 \subseteq ac' \]
\[ = \text{PBMH}(P) \]

\[ \square \]
Lemma D.4.8

\[ \text{PBMH}(P \ ; \ ac = \emptyset) = P \ ; \ ac = \emptyset \]

Proof.

\[ \begin{align*}
\text{PBMH}(P \ ; \ ac = \emptyset) & \quad \{ \text{Definition of PBMH} \} \\
= (P \ ; \ ac = \emptyset) \ ; \ ac \subseteq ac' & \quad \{ \text{Associativity of sequential composition} \} \\
= P \ ; (ac = \emptyset \ ; \ ac \subseteq ac') & \quad \{ \text{Definition of sequential composition} \} \\
= P \ ; (\exists ac_0 \bullet ac = \emptyset \land ac_0 \subseteq ac') & \quad \{ \text{Propositional calculus} \} \\
= P \ ; (ac = \emptyset \land \exists ac_0 \bullet ac_0 \subseteq ac') & \quad \{ \text{Choose } ac_0 = \emptyset \} \\
= P \ ; (ac = \emptyset \land \text{true}) & \quad \{ \text{Propositional calculus} \} \\
= P \ ; ac = \emptyset & 
\end{align*} \]

\[ \square \]

D.5 Set theory

Lemma D.5.1 ($\subseteq$-transitivity-multiple)

\[ \exists D \bullet (\exists A \bullet P(A) \land A \subseteq D) \land (\exists B \bullet P(B) \land B \subseteq D) \land D \subseteq E \]

\[ = (\exists A \bullet P(A) \land A \subseteq E) \land (\exists B \bullet P(B) \land B \subseteq E) \]

Proof. (Implication)

\[ \exists D \bullet (\exists A \bullet P(A) \land A \subseteq D) \land (\exists B \bullet P(B) \land B \subseteq D) \land D \subseteq E \]

\[ \implies (\exists D, A \bullet P(A) \land A \subseteq D \land D \subseteq E) \land (\exists D, B \bullet P(B) \land B \subseteq D \land D \subseteq E) \]

\[ = (\exists A \bullet P(A) \land A \subseteq E) \land (\exists B \bullet P(B) \land B \subseteq E) \]

\[ \square \]

Proof. (Reverse implication)

\[ \left( (\exists A \bullet P(A) \land A \subseteq E) \land (\exists B \bullet P(B) \land B \subseteq E) \right) \]

\[ \implies \exists D \bullet (\exists A \bullet P(A) \land A \subseteq D) \land (\exists B \bullet P(B) \land B \subseteq D) \land D \subseteq E \]

\[ \{ \text{Set } D = E \} \]
\[
= \left( (\exists A \cdot P(A) \land A \subseteq E) \land (\exists B \cdot P(B) \land B \subseteq E) \Rightarrow (\exists A \cdot P(A) \land A \subseteq E) \land (\exists B \cdot P(B) \land B \subseteq E) \land E \subseteq E \right)
\]
{Reflexivity of subset inclusion and propositional calculus}

= true

Lemma D.5.2

\[s \in A \Rightarrow A \neq \emptyset\]

Proof.

\[s \in A \Rightarrow A \neq \emptyset\] \hspace{1cm} \{Property of sets\}
= \[s \in A \Rightarrow \exists z \cdot z \in A\] \hspace{1cm} \{Choose \(z = s\}\}
= \[s \in A \Rightarrow s \in A\] \hspace{1cm} \{Propositional calculus\}
= \[true\]

Lemma D.5.3

\[\exists B \cdot B \neq \emptyset \land B \subseteq C \iff C \neq \emptyset\]

Proof. (Implication) By contradiction: Suppose the consequent is false yet the antecedent is true. Then \(C = \emptyset\).

\[\exists B \cdot B \neq \emptyset \land B \subseteq C\] \hspace{1cm} \{Assumption: \(C = \emptyset\}\}
= \[\exists B \cdot B \neq \emptyset \land B \subseteq \emptyset\] \hspace{1cm} \{Property of subset inclusion\}
= \[\exists B \cdot B \neq \emptyset \land B = \emptyset\] \hspace{1cm} \{Propositional calculus\}
= \[false\]

Proof. (Reverse implication)

\[C \neq \emptyset \Rightarrow \exists B \cdot B \neq \emptyset \land B \subseteq C\] \hspace{1cm} \{Choose \(B = C\}\}
= \[C \neq \emptyset \Rightarrow C \neq \emptyset \land C \subseteq C\] \hspace{1cm} \{Reflexivity of subset inclusion\}
= \[C \neq \emptyset \Rightarrow C \neq \emptyset\] \hspace{1cm} \{Propositional calculus\}
= \[true\]
Lemma D.5.4

\[ \exists a_{c_0} \bullet s \in a_{c_0} \land a_{c_0} \subseteq a_{c'} \iff s \in a_{c'} \]

Proof. (Implication)

\[ \exists a_{c_0} \bullet s \in a_{c_0} \land a_{c_0} \subseteq a_{c'} \]  \{Definition of subset inclusion\}
\[ = \exists a_{c_0} \bullet s \in a_{c_0} \land (\forall z \bullet z \in a_{c_0} \Rightarrow z \in a_{c'}) \]  \{Assume \( s \in a_{c_0} \) then there is a case when \( z = s \)\}
\[ = \exists a_{c_0} \bullet s \in a_{c_0} \land (\forall z \bullet z \in a_{c_0} \Rightarrow z \in a_{c'}) \land (s \in a_{c_0} \Rightarrow s \in a_{c'}) \]  \{Assume \( s \in a_{c_0} \) and propositional calculus\}
\[ \Rightarrow s \in a_{c'} \]

\[ \square \]

Proof. (Reverse implication)

\[ s \in a_{c'} \Rightarrow (\exists a_{c_0} \bullet s \in a_{c_0} \land a_{c_0} \subseteq a_{c'}) \]  \{Choose \( a_{c_0} = a_{c'} \)\}
\[ = (s \in a_{c'}) \Rightarrow (s \in a_{c'} \land a_{c'} \subseteq a_{c'}) \]  \{Reflexivity of subset inclusion and propositional calculus\}
\[ = \text{true} \]

\[ \square \]
Appendix E

Sequential composition ($\mathcal{A}$)

E.1 Algebraic properties

Law E.1.1 ( ; $\mathcal{A}$-not-free) Provided $ac'$ is not free in $P$.

\[ P ; \mathcal{A} Q = P \]

Proof.

\[ P ; \mathcal{A} Q \]
\[ = P[\{z : State \mid Q[z/s]\}/ac'] \]
\[ = P \]

\[ \Box \]

Law E.1.2 ( ; $\mathcal{A}$-associativity) Provided $P$ and $Q$ satisfy PBMH.

\[ P ; \mathcal{A} (Q ; \mathcal{A} R) = (P ; \mathcal{A} Q) ; \mathcal{A} R \]

Proof.

\[ (P ; \mathcal{A} Q) ; \mathcal{A} R \]
\[ = (P[\{z \mid Q[z/s]\}/ac'])[\{z \mid R[z/s]\}/ac'] \]
\[ = (P ; ac \subseteq ac')[\{z \mid Q[z/s]\}/ac)][\{z \mid R[z/s]\}/ac'] \]
\[ = (P ; ac \subseteq \{z \mid Q[z/s]\})[\{z \mid R[z/s]\}/ac'] \]

{Definition of subset inclusion and property of sets}
\[ (P ; \forall z \cdot z \in ac \Rightarrow Q[z/s])[\{z \mid R[z/s]\}/ac'] \]

{Assumption: \( Q \) satisfies \textbf{PBMH}}

\[ = (P ; \forall z \cdot z \in ac \Rightarrow (Q[z/s] ; ac \subseteq ac'))[\{z \mid R[z/s]\}/ac'] \]

{Substitution}

\[ = (P ; \forall z \cdot z \in ac \Rightarrow (Q[z/s] ; ac \subseteq ac'))[\{z \mid R[z/s]\}/ac'] \quad \text{Re-introduce \( ac' \)} \]

{Assumption: \( Q \) satisfies \textbf{PBMH}, and definition of sequential composition}

\[ = (P ; \forall z \cdot z \in ac \Rightarrow (Q[z/s] ; \_ A R)) \quad \text{Re-introduce \( ac' \)} \]

\[ = (P ; \forall z \cdot z \in ac \Rightarrow z \in ac')[\{z \mid Q[z/s] ; \_ A R\}/ac'] \quad \text{Definition of subset inclusion and sequential composition} \]

\[ = (P ; ac \subseteq ac') ; A (Q ; \_ A R) \quad \text{Assumption: \( P \) satisfies \textbf{PBMH}} \]

\[ = P ; A (Q ; \_ A R) \]

\[ \square \]

**Law E.1.3 \( ; \_ A\)-negation**

\[ \neg (P ; \_ A Q) = (\neg P ; \_ A Q) \]

**Proof.**

\[ \neg (P ; \_ A Q) \quad \text{Definition of sequential composition} \]

\[ = \neg (P[\{z \mid Q[z/s]\}/ac']) \quad \text{Propositional calculus} \]

\[ = (\neg P[\{z \mid Q[z/s]\}/ac']) \quad \text{Definition of sequential composition} \]

\[ = (\neg P ; \_ A Q) \]

\[ \square \]

### E.2 Closure properties

**Law E.2.1 \( ; \_ A\)-closure**  \( \) Provided \( P \) and \( Q \) satisfy \textbf{PBMH}.

\[ \textbf{PBMH}(P ; A Q) = P ; A Q \]

**Proof.** (Implication)

\[ \textbf{PBMH}(P ; A Q) \quad \text{Definition of \textbf{PBMH}} \]

Revision: 704f887 (2014-02-04 11:14:10 +0000) 255
\[
(P ; A Q) \ ; ac \subseteq ac'
\]
\[\text{Assumption: } P \text{ satisfies } \text{PBMH}\]
\[
= ((P ; ac \subseteq ac') ; A Q) \ ; ac \subseteq ac'
\]
\[\text{Definition of sequential composition}\]
\[
= ((\exists ac_0 \cdot P[ac_0/ac'] \land ac_0 \subseteq ac') ; A Q) \ ; ac \subseteq ac'
\]
\[\text{Definition of } ; A \text{ and substitution}\]
\[
= (\exists ac_0 \cdot P[ac_0/ac'] \land ac_0 \subseteq \{z \mid Q[z/s]\}) \ ; ac \subseteq ac'
\]
\[\text{Assumption: } Q \text{ satisfies } \text{PBMH}\]
\[
= \left(\exists ac_0 \cdot P[ac_0/ac'] \land ac_0 \subseteq \{z \mid (Q ; ac \subseteq ac')[z/s]\}\right) \ ; ac \subseteq ac'
\]
\[\text{Definition of subset inclusion}\]
\[
= \exists ac_1 \cdot \left(\exists ac_0 \cdot P[ac_0/ac'] \land \forall z \cdot z \in ac_0 \Rightarrow ((Q ; ac \subseteq ac')[z/s]) \land ac_1 \subseteq ac'\right)
\]
\[\text{Definition of sequential composition}\]
\[
= \exists ac_1 \cdot \left(\exists ac_0 \cdot P[ac_0/ac'] \land \forall z \cdot z \in ac_0 \Rightarrow ((Q ; ac \subseteq ac')[z/s]) \land ac_1 \subseteq ac'\right)
\]
\[\text{Substitution}\]
\[
= \exists ac_0, ac_1 \cdot P[ac_0/ac'] \land \forall z \cdot z \in ac_0 \Rightarrow ((Q ; ac \subseteq ac')[z/s]) \land ac_1 \subseteq ac'
\]
\[\text{Predicate calculus: quantifier scope}\]
\[
= \exists ac_0 \cdot P[ac_0/ac'] \land \exists ac_1 \cdot \forall z \cdot (z \in ac_0 \Rightarrow (Q ; ac \subseteq ac')[z/s]) \land ac_1 \subseteq ac'
\]
\[\text{Predicate calculus}\]
\[
\begin{align*}
\exists ac_0 \cdot P[ac_0/ac'] \\
\wedge \\
\exists ac_1 \cdot \forall z \cdot (z \notin ac_0 \land ac_1 \subseteq ac') \\
\vee \\
(\exists ac_0 \cdot Q[ac_0/ac'][z/s] \land ac_0 \subseteq ac_1 \land ac_1 \subseteq ac')
\end{align*}
\] 
{\text{Predicate calculus}}

\[
\Rightarrow \end{align*}
\begin{align*}
\exists ac_0 \cdot P[ac_0/ac'] \\
\wedge \\
\forall z \cdot \exists ac_1 \cdot (z \notin ac_0 \land ac_1 \subseteq ac') \\
\vee \\
(\exists ac_0, ac_1 \cdot Q[ac_0/ac'][z/s] \land ac_0 \subseteq ac_1 \land ac_1 \subseteq ac')
\end{align*}
\] 
{\text{Predicate calculus}}

\[
\begin{align*}
\exists ac_0 \cdot P[ac_0/ac'] \\
\wedge \\
\forall z \cdot (z \notin ac_0) \\
\vee \\
(\exists ac_0 \cdot Q[ac_0/ac'][z/s] \land ac_0 \subseteq ac')
\end{align*}
\] 
{\text{Property of sets and predicate calculus}}

\[
\begin{align*}
\exists ac_0 \cdot P[ac_0/ac'] \\
\wedge \\
\forall z \cdot (z \notin ac_0) \\
\vee \\
(\exists ac_0 \cdot Q[ac_0/ac'][z/s] \land ac_0 \subseteq ac')
\end{align*}
\] 
{\text{Predicate calculus}}

\[
\begin{align*}
\exists ac_0 \cdot P[ac_0/ac'] \\
\wedge \\
\forall z \cdot (z \in ac_0 \Rightarrow (\exists ac_0 \cdot Q[ac_0/ac'][z/s] \land ac_0 \subseteq ac')
\end{align*}
\] 
{\text{Substitution}}

\[
\begin{align*}
\exists ac_0 \cdot P[ac_0/ac'] \\
\wedge \\
\forall z \cdot (z \in ac_0 \Rightarrow (\exists ac_0 \cdot Q[ac_0/ac'][z/s] \land ac_0 \subseteq ac')
\end{align*}
\] 
{\text{Definition of sequential composition}}

\[
\begin{align*}
\exists ac_0 \cdot P[ac_0/ac'] \\
\wedge \\
\forall z \cdot (z \in ac_0 \Rightarrow (Q \land ac \subseteq ac')[z/s]
\end{align*}
\] 
{\text{Assumption: Q satisfies PBMM}}
\[
\exists \, ac_0 \bullet P[ac_0/ac'] \land \forall \, z \bullet z \in ac_0 \Rightarrow Q[z/s] \\
\{ \text{Definition of subset inclusion and re-introduce set comprehension} \}
\]
\[
= \exists \, ac_0 \bullet P[ac_0/ac'] \land ac_0 \subseteq \{z \mid Q[z/s]\} \\
\{ \text{Re-introduce } ac', \text{ definition of } ;_A \text{ and substitution} \}
\]
\[
= (\exists \, ac_0 \bullet P[ac_0/ac'] \land ac_0 \subseteq ac')\{\{z \mid Q[z/s]\}/ac'\} \\
\{ \text{Definition of sequential composition} \}
\]
\[
= (P ; ac_0 \subseteq ac') ;_A Q \\
\{ \text{Assumption: } P \text{ satisfies } \text{PBMH} \}
\]
\[
= P \, \text{seq} \, Q
\]

\text{Proof.} (Reverse implication)

\[
P ;_A Q \\
\Rightarrow \text{PBMH}(P ;_A Q)
\]

\text{E.3 Distributivity with respect to disjunction}

\textbf{Law E.3.1} ( ;_A\text{-right-distributivity-disjunction})

\[
(P \lor Q) ;_A R = (P ;_A R) \lor (Q ;_A R)
\]

\text{Proof.}

\[
(P \lor Q) ;_A R \\
= (P \lor Q)[\{z \mid R[z/s]\}/ac'] \\
\{ \text{Definition of sequential composition} \}
\]
\[
= (P[\{z \mid R[z/s]\}/ac'] \lor Q[\{z \mid R[z/s]\}/ac']) \\
\{ \text{Definition of sequential composition} \}
\]
\[
= (P ;_A R) \lor (Q ;_A R)
\]
E.4 Distributivity with respect to conjunction

Law E.4.1 (\(\cdot\_\text{\(A\)-right-distributivity-conjunction}\))

\[(P \land Q) \cdot_\mathcal{A} R = (P \cdot_\mathcal{A} R) \land (Q \cdot_\mathcal{A} R)\]

Proof.

\[(P \land Q) \cdot_\mathcal{A} R \quad \{\text{Definition of } \cdot_\mathcal{A}\} \]
\[= (P \land Q)[\{z \mid R[z/s]\}/ac'] \quad \{\text{Property of substitution}\} \]
\[= (P[\{z \mid R[z/s]\}/ac'] \land Q[\{z \mid R[z/s]\}/ac']) \quad \{\text{Definition of } \cdot_\mathcal{A}\} \]
\[= (P \cdot_\mathcal{A} R) \land (Q \cdot_\mathcal{A} R) \]

\[\square\]

Law E.4.2 Provided \(P\) satisfies PB
MH2. This property does not necessarily hold in the opposite direction (See Example 19).

\[P \cdot_\mathcal{A} (Q \land R) \Rightarrow (P \cdot_\mathcal{A} Q) \land (P \cdot_\mathcal{A} R)\]

Proof. (Implication) To be revised. \(\square\)

Example 19

\[((ac' \neq \emptyset \cdot_\mathcal{A} s.x = 1) \land (ac' \neq \emptyset \cdot_\mathcal{A} s.x = 2)) \Rightarrow (ac' \neq \emptyset \cdot_\mathcal{A} (s.x = 1 \land s.x = 2))\]
\{Propositional calculus\}
\[= ((ac' \neq \emptyset \cdot_\mathcal{A} s.x = 1) \land (ac' \neq \emptyset \cdot_\mathcal{A} s.x = 2)) \Rightarrow (ac' \neq \emptyset \cdot_\mathcal{A} false)\]
\{Definition of sequential composition\}
\[= ((ac' \neq \emptyset \cdot_\mathcal{A} s.x = 1) \land (ac' \neq \emptyset \cdot_\mathcal{A} s.x = 2)) \Rightarrow (ac' \neq \emptyset[\{z \mid false\}/ac'])\]
\{Property of sets, substitution and propositional calculus\}
\[= ((ac' \neq \emptyset \cdot_\mathcal{A} s.x = 1) \land (ac' \neq \emptyset \cdot_\mathcal{A} s.x = 2)) \Rightarrow false\]
\{Definition of sequential composition and substitution\}
\[= ((\{z \mid z.x = 1\} \neq \emptyset) \land (\{z \mid z.x = 2\} \neq \emptyset)) \Rightarrow false\]
\{Property of sets\}
\[= ((\exists z \cdot z.x = 1) \land (\exists z \cdot z.x = 2)) \Rightarrow false\]
\{One-point\}
\[= true \Rightarrow false\]
\{Propositional calculus\}
\[= false\]
E.5 Extreme points

Law E.5.1 (; \textit{A}-P-sequence-\textit{false}:1) Provided $P$ satisfies PBHM.

$$P ; \textit{A} \text{false} = P[\emptyset/ac']$$

Proof.

\begin{align*}
P ; \textit{A} \text{false} & \quad \{\text{Assumption: } P \text{ satisfies PBHM}\} \\
= (P ; ac \subseteq ac') ; \textit{A} \text{false} & \quad \{\text{Definition of sequential composition}\} \\
= (\exists ac_0 \cdot P[ac_0/ac'] \land ac_0 \subseteq ac') ; \textit{A} \text{false} & \quad \{\text{Definition of } ; \textit{A}\} \\
= \exists ac_0 \cdot P[ac_0/ac'] \land ac_0 \subseteq \emptyset & \quad \{\text{Property of sets and one-point rule}\} \\
= P[\emptyset/ac'] & \end{align*}

\[\square\]

Law E.5.2 (; \textit{A}-P-sequence-\textit{true}) Provided $P$ satisfies PBHM.

$$P ; \textit{A} \text{true} = \exists ac' \cdot P$$

Proof.

\begin{align*}
P ; \textit{A} \text{true} & \quad \{\text{Assumption: } P \text{ satisfies PBHM}\} \\
= (P ; ac \subseteq ac') ; \textit{A} \text{true} & \quad \{\text{Definition of sequential composition}\} \\
= (\exists ac_0 \cdot P[ac_0/ac'] \land ac_0 \subseteq ac') ; \textit{A} \text{true} & \quad \{\text{Definition of } ; \textit{A}\} \\
= \exists ac_0 \cdot P[ac_0/ac'] \land ac_0 \subseteq \{z \mid \text{true}\} & \quad \{\text{Property of sets}\} \\
= \exists ac_0 \cdot P[ac_0/ac'] \land (\forall z \cdot z \in ac_0 \Rightarrow \text{true}) & \quad \{\text{Propositional calculus}\} \\
= \exists ac_0 \cdot P[ac_0/ac'] & \quad \{\text{One-point rule}\} \\
= \exists ac_0 \cdot (\exists ac' \cdot P \land ac' = ac_0) & \quad \{\text{One-point rule: } ac_0 \text{ not free in } P\} \\
= \exists ac' \cdot P & \end{align*}

\[\square\]

E.6 Algebraic properties and sequential composition

Law E.6.1 (; \textit{A}-sequence-left-associativity) Provided ok and ac are not free in R.

$$(P ; Q) ; \textit{A} R = P ; (Q ; \textit{A} R)$$

Revision: 704f887 (2014-02-04 11:14:10 +0000) 260
Proof.

\[(P ; Q) ; _A R \quad \{\text{Definition of sequential composition}\}\]
\[= (\exists ok_0, ac_0 \cdot P[ok_0, ac_0/ok, ac'] \land Q[ok_0, ac_0/ok, ac]) ; _A R \quad \{\text{Definition of } ; _A\}\]
\[= (\exists ok_0, ac_0 \cdot P[ok_0, ac_0/ok, ac'] \land Q[ok_0, ac_0/ok, ac])\{\{z | R[z/s]\}/ac'\} \quad \{\text{Substitution: } ac' \text{ not free in } ac_0\}\]
\[= (\exists ok_0, ac_0 \cdot P[ok_0, ac_0/ok, ac'] \land Q[ok_0, ac_0/ok, ac])\{\{z | R[z/s]\}/ac'\} \quad \{\text{Assumption: } \{ok, ac\} \text{ not free in } R\}\]
\[= (\exists ok_0, ac_0 \cdot P[ok_0, ac_0/ok, ac'] \land Q[\{z | R[z/s]\}/ac'][ok_0, ac_0/ok, ac]) \quad \{\text{Definition of sequential composition}\}\]
\[= P ; Q[\{z | R[z/s]\}/ac'] \quad \{\text{Definition of } ; _A\}\]
\[= P ; (Q ; _A R) \]

\[\square\]

### E.7 Skip

**Definition 68**

\[\Pi_A \equiv s \in ac'\]

**Law E.7.1** \(\Pi_A\) is a fixed point of PBMH.

\[\text{PBMH}(\Pi_A) = \Pi_A\]

**Proof.**

\[\text{PBMH}(\Pi_A) \quad \{\text{Definition of } \Pi_A \text{ and } \text{PBMH}\}\]
\[= (s \in ac') ; (ac \subseteq ac') \quad \{\text{Definition of sequential composition and substitution}\}\]
\[= \exists ac_0 \cdot s \in ac_0 \land ac_0 \subseteq ac' \quad \{\text{Law [D.5.4]}\}\]
\[= s \in ac' \]

\[\square\]

**Law E.7.2** \(( ; _A\Pi_A\text{-left-unit})\)

\[\Pi_A \; _A P\]

Revision: 704f887 (2014-02-04 11:14:10 +0000) 261
Proof.

\[ \Pi_A ; A P \]
\[ = s \in ac' ; A P \] \{Definition of \( ; A \) and substitution\}
\[ = s \in \{z | P[z/s]\} \] \{Property of sets\}
\[ = P[z/s][s/z] \] \{Substitution\}
\[ = P \]

\[ \square \]

Law E.7.3 (\( ; A-\Pi_A\)-right-unit) Provided \( P \) satisfies PBMH.

\[ P ; A \Pi_A \]

Proof.

\[ P ; A \Pi_A \] \{Definition of \( \Pi_A \)\}
\[ = P ; A (s \in ac') \] \{Assumption: \( P \) satisfies PBMH\}
\[ = (P ; ac \subseteq ac') ; A (s \in ac') \] \{Definition of sequential composition\}
\[ = (\exists ac_0 \bullet P[ac_0/ac'] \land ac_0 \subseteq ac') ; A (s \in ac') \] \{Definition of \( ; A \)\}
\[ = \exists ac_0 \bullet P[ac_0/ac'] \land ac_0 \subseteq \{z | z \in ac'\} \] \{Property of sets\}
\[ = \exists ac_0 \bullet P[ac_0/ac'] \land (\forall z \bullet z \in ac_0 \Rightarrow z \in ac') \] \{Definition of subset inclusion\}
\[ = \exists ac_0 \bullet P[ac_0/ac'] \land ac_0 \subseteq ac' \] \{Definition of sequential composition\}
\[ = P ; ac \subseteq ac' \] \{Assumption: \( P \) satisfies PBMH\}
\[ = P \]

\[ \square \]

E.8 Other lemmas

Lemma E.8.1 Provided \( P \) is PBMH-healthy and \( s \) is not free in \( e \).

\[ P ; A (Q \Rightarrow (R \land e)) = (P ; A \neg Q) \lor ((P ; A (Q \Rightarrow R)) \land e) \]
Proof.

\[ P \land \mathcal{A} (Q \Rightarrow (R \land e)) \]

{Assumption: \( P \) is \textbf{PBMH}-healthy (Lemma \textbf{D.1.1})}

\[ = (\exists ac_0 \cdot P[ac_0/ac'] \land ac_0 \subseteq ac') \land \mathcal{A} (Q \Rightarrow (R \land e)) \]

{Definition of \( \land \mathcal{A} \) and substitution}

\[ = \exists ac_0 \cdot P[ac_0/ac'] \land ac_0 \subseteq \{s \mid Q \Rightarrow (R \land e)\} \]

{Property of sets and \( s \) not free in \( e \)}

\[ = \exists ac_0 \cdot P[ac_0/ac'] \land \forall z \cdot z \in ac_0 \Rightarrow (Q[z/s] \Rightarrow (R[z/s] \land e)) \]

{Lemma \textbf{E.8.4}}

\[ = \exists ac_0 \cdot P[ac_0/ac'] \land \forall z \cdot z \in ac_0 \Rightarrow \neg Q[z/s] \]

{Predicate calculus}

\[ \lor \]

\[ = \exists ac_0 \cdot P[ac_0/ac'] \land ((\forall z \cdot z \in ac_0 \Rightarrow (Q[z/s] \Rightarrow R[z/s])) \land e) \]

{Property of sets}

\[ = \exists ac_0 \cdot P[ac_0/ac'] \land ((\forall z \cdot z \in ac_0 \Rightarrow z \in \{s \mid \neg Q\}) \land e) \]

{Property of sets}

\[ = \exists ac_0 \cdot P[ac_0/ac'] \land ac_0 \subseteq \{s \mid \neg Q\} \]

{Predicate calculus}

\[ \lor \]

\[ = \exists ac_0 \cdot P[ac_0/ac'] \land ac_0 \subseteq \{s \mid Q \Rightarrow R\} \land e \]

{Definition of \( \land \mathcal{A} \) and substitution}

\[ = (\exists ac_0 \cdot P[ac_0/ac'] \land ac_0 \subseteq ac') \land \mathcal{A} \neg Q \]

{Assumption: \( P \) is \textbf{PBMH}-healthy (Lemma \textbf{D.1.1})}
\[ (P \land \lnot Q) \lor ((P \land (Q \Rightarrow R)) \land e) \]

\[ \square \]

**Lemma E.8.2** Provided \( P \) satisfies PBMH.

\[ P \land \mathcal{A} (Q \land ok') = (P \land \mathcal{A} \text{false}) \lor ((P \land \mathcal{A} Q) \land ok') \]

*Proof.*

\[ P \land \mathcal{A} (Q \land ok') \quad \{ \text{Assumption: } P \text{ satisfies PBMH} \} \]

\[ = \text{PBMH}(P) \land \mathcal{A} (Q \land ok') \quad \{ \text{Definition of PBMH} \} \]

\[ = (P \land ac \subseteq ac') \land \mathcal{A} (Q \land ok') \quad \{ \text{Definition of sequential composition and substitution} \} \]

\[ = (\exists ac_0 \cdot P[ac_0/ac'] \land ac_0 \subseteq ac') \land \mathcal{A} (Q \land ok') \quad \{ \text{Definition of } \land \mathcal{A} \text{ and substitution} \} \]

\[ = \exists ac_0 \cdot P[ac_0/ac'] \land ac_0 \subseteq \{ z \mid (Q \land ok')[z/s] \} \quad \{ \text{Property of substitution} \} \]

\[ = \exists ac_0 \cdot P[ac_0/ac'] \land ac_0 \subseteq \{ z \mid Q[z/s] \land ok' \} \quad \{ \text{Propositional calculus} \} \]

\[ = \exists ac_0 \cdot P[ac_0/ac'] \land (\forall z \cdot z \in ac_0 \Rightarrow Q[z/s]) \land (\forall z \cdot z \in ac_0 \Rightarrow ok') \quad \{ \text{Propositional calculus} \} \]

\[ = \exists ac_0 \cdot P[ac_0/ac'] \land (\forall z \cdot z \in ac_0 \Rightarrow Q[z/s]) \land (\forall z \cdot z \notin ac_0 \lor ok') \quad \{ \text{Predicate calculus: ok' not in z, move quantifier} \} \]

\[ = \exists ac_0 \cdot P[ac_0/ac'] \land (\forall z \cdot z \in ac_0 \Rightarrow Q[z/s]) \land ((\forall z \cdot z \notin ac_0) \lor ok') \quad \{ \text{Predicate calculus: distribution} \} \]

\[ = \exists ac_0 \cdot P[ac_0/ac'] \land \left( (\forall z \cdot z \in ac_0 \Rightarrow Q[z/s]) \land (\forall z \cdot z \notin ac_0) \right) \quad \{ \text{Propositional calculus} \} \]

\[ \lor \]

\[ (\forall z \cdot z \in ac_0 \Rightarrow Q[z/s]) \land ok' \quad \{ \text{Propositional calculus} \} \]

\[ = \exists ac_0 \cdot P[ac_0/ac'] \land ((\forall z \cdot z \notin ac_0) \lor (\forall z \cdot z \in ac_0 \Rightarrow Q[z/s] \land ok')) \quad \{ \text{Propositional calculus} \} \]

\[ \]
Proof. 

\[
\begin{align*}
= & \left( (\exists a_{c_0} \bullet P[a_{c_0}/ac'] \land \forall z \bullet z \notin ac_0) \\
\lor & \left( (\exists a_{c_0} \bullet P[a_{c_0}/ac'] \land (\forall z \bullet z \in ac_0 \Rightarrow Q[z/s]) \land ok') \right) \\
\{\text{Property of sets and introduce set comprehension}\}
\right) \\
= & \left( (\exists a_{c_0} \bullet P[a_{c_0}/ac'] \land ac_0 = \emptyset) \\
\lor & \left( (\exists a_{c_0} \bullet P[a_{c_0}/ac'] \land ac_0 \subseteq \{z \mid Q[z/s]\}) \land ok') \right) \\
\{\text{One-point rule and substitution}\}
\right) \\
= & \left( P[\emptyset/ac'] \\
\lor & \left( (\exists a_{c_0} \bullet P[a_{c_0}/ac'] \land ac_0 \subseteq \{z \mid Q[z/s]\}) \land ok') \right) \\
\{\text{Re-introduce } ac'\}
\right) \\
= & \left( P[\emptyset/ac'] \\
\lor & \left( (\exists a_{c_0} \bullet P[a_{c_0}/ac'] \land ac_0 \subseteq ac'\{\{z \mid Q[z/s]\}/ac'] \land ok') \right) \\
\{\text{Definition of } \land A\}
\right) \\
= & \left( P[\emptyset/ac'] \\
\lor & \left( (\exists a_{c_0} \bullet P[a_{c_0}/ac'] \land ac_0 \subseteq ac'\ A Q) \land ok') \right) \\
\{\text{Definition of sequential composition}\}
\right) \\
= & P[\emptyset/ac'] \lor ((P \land ac \subseteq ac') \land ac' \land A Q) \land ok') \\
\{\text{Assumption: } P \text{ satisfies PBMH}\}
\right) \\
= & P[\emptyset/ac'] \lor ((P \land A Q) \land ok') \\
\{\text{Law [E.5.1]}\}
\right) \\
= & (P \land A \text{false}) \lor ((P \land A Q) \land ok') \\
\right)
\end{align*}
\]

\[\square\]

**Lemma E.8.3** Provided \( P \) is \textbf{PBMH}-healthy.

\[ P \land A (Q \Rightarrow (R \land ok')) = (P \land A \neg Q) \lor ((P \land A (Q \Rightarrow R)) \land ok') \]

Proof. 

\[ P \land A (Q \Rightarrow (R \land ok')) \]

\{Assumption: \( P \) is \textbf{PBMH}-healthy (Lemma D.1.1)\}

\[ = (\exists a_{c_0} \bullet P[a_{c_0}/ac'] \land ac_0 \subseteq ac') \land A (Q \Rightarrow (R \land ok')) \]

\{Definition of \land A \text{ and substitution}\}
\[
\exists a_0 \cdot P[a_0/ac'] \land a_0 \subseteq \{s \mid Q \Rightarrow (R \land ok')\} \quad \{\text{Property of sets}\}
\]

\[
= \exists a_0 \cdot P[a_0/ac'] \land \forall z \cdot z \in a_0 \Rightarrow (Q[z/s] \Rightarrow (R[z/s] \land ok')) \quad \{\text{Lemma E.8.4}\}
\]

\[
= \exists a_0 \cdot P[a_0/ac'] \land \left( (\forall z \cdot z \in a_0 \Rightarrow \neg Q[z/s]) \lor \left( (\forall z \cdot z \in a_0 \Rightarrow (Q[z/s] \Rightarrow R[z/s])) \land ok' \right) \right)
\quad \{\text{Predicate calculus}\}
\]

\[
= \left( (\exists a_0 \cdot P[a_0/ac'] \land (\forall z \cdot z \in a_0 \Rightarrow \neg Q[z/s])) \lor \left( (\exists a_0 \cdot P[a_0/ac'] \land ((\forall z \cdot z \in a_0 \Rightarrow (Q[z/s] \Rightarrow R[z/s]))) \land ok' \right) \right)
\quad \{\text{Property of sets}\}
\]

\[
= \left( (\exists a_0 \cdot P[a_0/ac'] \land \forall z \cdot z \in a_0 \Rightarrow z \in \{s \mid \neg Q\}) \lor \left( (\exists a_0 \cdot P[a_0/ac'] \land ((\forall z \cdot z \in a_0 \Rightarrow z \in \{s \mid Q \Rightarrow R\}) \land ok' \right) \right)
\quad \{\text{Property of sets}\}
\]

\[
= \left( (\exists a_0 \cdot P[a_0/ac'] \land a_0 \subseteq \{s \mid \neg Q\}) \lor \left( (\exists a_0 \cdot P[a_0/ac'] \land a_0 \subseteq \{s \mid Q \Rightarrow R\} \land ok' \right) \right)
\quad \{\text{Predicate calculus}\}
\]

\[
= \left( (\exists a_0 \cdot P[a_0/ac'] \land a_0 \subseteq \{s \mid \neg Q\}) \lor \left( ((\exists a_0 \cdot P[a_0/ac'] \land a_0 \subseteq ac') ; \ A \neg Q) \lor \left( ((\exists a_0 \cdot P[a_0/ac'] \land a_0 \subseteq ac') ; \ A \ (Q \Rightarrow R) \land ok' \right) \right) \right)
\quad \{\text{Definition of ; \ A and substitution}\}
\]

\[
= \left( (\exists a_0 \cdot P[a_0/ac'] \land a_0 \subseteq ac') ; \ A \neg Q \right) \lor \left( ((\exists a_0 \cdot P[a_0/ac'] \land a_0 \subseteq ac') ; \ A \ (Q \Rightarrow R) \land ok' \right) \right)
\quad \{\text{Assumption: \ P is \ \textbf{PBMH}-healthy (Lemma D.1.1)\}\}
\]

\[
= (P ; \ A \neg Q) \lor ((P ; \ A \ (Q \Rightarrow R)) \land ok')
\]

\[\square\]

**Lemma E.8.4** \( \forall x \ \checkmark_P \ \checkmark_A \) Provided \( x \) is not free in \( e \)

\[
\forall x \cdot P \Rightarrow (Q \Rightarrow (R \land e))
\]

\[
= (\forall x \cdot P \Rightarrow \neg Q) \lor ((\forall x \cdot P \Rightarrow (Q \Rightarrow R)) \land e)
\]

Revision: 704f887 (2014-02-04 11:14:10 +0000) 266
Proof.

\[ \forall x \cdot P \Rightarrow (Q \Rightarrow (R \land e)) \]
\[ = \forall x \cdot (P \land Q) \Rightarrow (R \land e) \]  \quad \{ \text{Predicate calculus} \}
\[ = \forall x \cdot ((P \land Q) \Rightarrow R) \land ((P \land Q) \Rightarrow e) \]  \quad \{ \text{Predicate calculus} \}
\[ = \forall x \cdot ((P \land Q) \Rightarrow R) \land (\neg (P \land Q) \lor e) \]  \quad \{ \text{Predicate calculus} \}
\[ = (\forall x \cdot (P \land Q) \Rightarrow R) \land (\forall x \cdot \neg (P \land Q) \lor e) \]  \quad \{ \text{Predicate calculus: } x \text{ is not free in } e \}
\[ = (\forall x \cdot (P \land Q) \Rightarrow R) \land ((\forall x \cdot \neg (P \land Q)) \lor e) \]  \quad \{ \text{Predicate calculus} \}
\[ = \left( \left( (\forall x \cdot (P \land Q) \Rightarrow R) \land (\forall x \cdot \neg (P \land Q)) \right) \lor \left( (\forall x \cdot (P \land Q) \Rightarrow R) \land e \right) \right) \]  \quad \{ \text{Predicate calculus} \}
\[ = \left( \left( (\forall x \cdot \neg (P \land Q)) \lor (\forall x \cdot (P \land Q) \Rightarrow R) \right) \land e \right) \]  \quad \{ \text{Predicate calculus} \}
\[ = (\forall x \cdot P \Rightarrow \neg Q) \lor ((\forall x \cdot P \Rightarrow (Q \Rightarrow R)) \land e) \]

\[ \square \]

Lemma E.8.5  \( \forall_P \forall_A \) Provided \( P \) is \( \text{PBMH-healthy} \).

\[ (P ; _A Q) \lor (P ; _A R) \Rightarrow (P ; _A (Q \lor R)) \]

Proof.

\[ (P ; _A Q) \lor (P ; _A R) \]
\[ \{ \text{Assumption: } P \text{ is } \text{PBMH-healthy} \text{ (Lemma [D.1.1])} \} \]
\[ = \left( (\exists a_{c_0} \cdot P[a_{c_0}/ac'] \land a_{c_0} \subseteq ac') ; _A Q \right) \lor \left( (\exists a_{c_0} \cdot P[a_{c_0}/ac'] \land a_{c_0} \subseteq ac') ; _A R \right) \]  \quad \{ \text{Definition of } ; _A \text{ and substitution} \}
\[ = \left( (\exists a_{c_0} \cdot P[a_{c_0}/ac'] \land a_{c_0} \subseteq \{s \mid Q\}) \right) \lor \left( (\exists a_{c_0} \cdot P[a_{c_0}/ac'] \land a_{c_0} \subseteq \{s \mid R\}) \right) \]  \quad \{ \text{Predicate calculus} \}
\[
\begin{align*}
  &= \exists \mathit{ac}_0 \cdot P[\mathit{ac}_0/\mathit{ac}'] \land (\mathit{ac}_0 \subseteq \{s \mid Q\} \lor \mathit{ac}_0 \subseteq \{s \mid R\}) \\
  &\quad \text{(Property of sets and predicate calculus)} \\
  \Rightarrow \quad &\exists \mathit{ac}_0 \cdot P[\mathit{ac}_0/\mathit{ac}'] \land \mathit{ac}_0 \subseteq \{s \mid Q \lor R\} \\
  &\quad \text{(Property of sets)} \\
  &= \exists \mathit{ac}_0 \cdot P[\mathit{ac}_0/\mathit{ac}'] \land \mathit{ac}_0 \subseteq \{s \mid Q \lor R\} \\
  &\quad \text{(Definition of \( \cdot \mathit{A} \) and substitution)} \\
  &= (\exists \mathit{ac}_0 \cdot P[\mathit{ac}_0/\mathit{ac}'] \land \mathit{ac}_0 \subseteq \mathit{ac}') \cdot \mathit{A} (Q \lor R) \\
  &\quad \text{(Assumption: \( P \) is PBMM-healthy (Lemma D.1.1))} \\
  &= P \cdot \mathit{A} (Q \lor R) \\
  \square
\end{align*}
\]

**Theorem E.8.1** Provided \( P \) is PBMM-healthy.

\[
(P \cdot \mathit{A} Q) \lor (P \cdot \mathit{A} \text{true}) = P \cdot \mathit{A} \text{true}
\]

Proof.

\[
\begin{align*}
(P \cdot \mathit{A} Q) \lor (P \cdot \mathit{A} \text{true}) &\quad \text{(Lemma E.8.5)} \\
= ((P \cdot \mathit{A} Q) \lor (P \cdot \mathit{A} \text{true})) \land (P \cdot \mathit{A} (Q \lor \text{true})) &\quad \text{(Predicate calculus)} \\
= ((P \cdot \mathit{A} Q) \lor (P \cdot \mathit{A} \text{true})) \land (P \cdot \mathit{A} \text{true}) &\quad \text{(Predicate calculus: absorption law)} \\
= (P \cdot \mathit{A} \text{true}) \\
\square
\end{align*}
\]

**Theorem E.8.2** Provided \( P \) is PBMM-healthy.

\[
(P \cdot \mathit{A} Q) \lor (P \cdot \mathit{A} \text{false}) = P \cdot \mathit{A} Q
\]

Proof.

\[
\begin{align*}
(P \cdot \mathit{A} Q) \lor (P \cdot \mathit{A} \text{false}) &\quad \text{(Lemma E.8.5)} \\
= ((P \cdot \mathit{A} Q) \lor (P \cdot \mathit{A} \text{false})) \land (P \cdot \mathit{A} (Q \lor \text{false})) &\quad \text{(Predicate calculus)} \\
= ((P \cdot \mathit{A} Q) \lor (P \cdot \mathit{A} \text{false})) \land (P \cdot \mathit{A} Q) &\quad \text{(Predicate calculus: absorption law)} \\
= P \cdot \mathit{A} Q \\
\square
\end{align*}
\]
Appendix F

Sequential composition (; Dac)

F.1 Properties

Law F.1.1 Provided P, Q and R are A-healthy.

\[(P; D_{ac} Q) ; D_{ac} R = P ; D_{ac} (Q ; D_{ac} R)\]

Proof.

\[(P; D_{ac} Q) ; D_{ac} R = \]

\[\{\text{Assumption: } P, D \text{ and } R \text{ are designs}\}\]

\[= ((\neg P^f \vdash P^t) ; D_{ac} (\neg Q^f \vdash Q^t)) ; D_{ac} (\neg R^f \vdash R^t)\]

\[\{\text{Definition of sequential composition (via Theorem 5.5.1)}\}\]

\[= ((\neg P^f ; A \text{ true}) \wedge (\neg P^t ; A Q^f) \wedge (\neg P^t ; A \text{ false}) \vdash P^t ; A Q^t) ; D_{ac} (\neg R^f \vdash R^t)\]

\[\{\text{Definition of sequential composition (via Theorem 5.5.1)}\}\]

\[= \left(\neg P^f ; A \text{ true} \wedge (\neg P^t ; A Q^f) \wedge (\neg P^t ; A \text{ false}) \wedge (\neg (P^t ; A Q^t) ; A \text{ true}) \wedge (P^t ; A Q^t) ; A R^t \right)\]

\[\{\text{Right-distributivity of sequential composition (Law 5.4.1)}\}\]

\[= \left(\neg P^f ; A \text{ true} \wedge (\neg P^t ; A Q^f) \wedge (\neg P^t ; A \text{ false}) \wedge \neg \neg \neg (P^t ; A Q^t) ; A \text{ true} \wedge (P^t ; A Q^t) ; A \text{ true} \right)\]

\[\{\text{Negation (Law 5.1.3)}\}\]

269
\( (\neg P^f ; A \text{ true}) \land A \text{ true} \land \neg ((P^t ; A Q^f) ; A \text{ true}) \land \neg ((P^t ; A false) ; A \text{ true}) \)

\( \vdash (P^t ; A Q^t) ; A R^t \)

\{Associativity (Law E.1.2)\}

\( (\neg P^f ; A (true ; A \text{ true})) \land \neg (P^t ; A (Q^f ; A \text{ true})) \land \neg (P^t ; A (false ; A \text{ true})) \)

\( \vdash (P^t ; A (Q^t ; A R^t)) \)

\{Property of ; A\}

\( (\neg P^f ; A \text{ true}) \land \neg (P^t ; A (Q^f ; A \text{ true})) \land \neg (P^t ; A (false ; A \text{ true})) \)

\( \vdash (P^t ; A (Q^t ; A R^t)) \)

\{Propositional calculus\}

\( \land \neg ((P^t ; A (Q^f ; A \text{ true})) \lor (P^t ; A (Q^t ; A R^t)) \lor (P^t ; A (Q^t ; A false))) \)

\( \vdash (P^t ; A (Q^t ; A R^t)) \)

\{Distributivity of sequential composition over disjunction (Law ??)\}

\( (\neg P^f ; A \text{ true}) \land \neg (P^t ; A false) \land \neg (P^t ; A (\neg Q^f ; A \text{ true}) \land (\neg Q^t ; A R^t) \land (\neg Q^t ; A false)) \)

\( \vdash (P^t ; A (Q^t ; A R^t)) \)

\{Propositional calculus\}

\( (\neg P^f \vdash P^t) ; \vdash_{\text{ac}} (P^t ; A ((\neg Q^f ; A \text{ true}) \land (\neg Q^t ; A R^t) \land (\neg Q^t ; A false)) \vdash Q^t ; A R^t) \)

\{Definition of ; \vdash_{\text{ac}}\}

\( (\neg P^f \vdash P^t) ; \vdash_{\text{ac}} ((\neg Q^f \vdash Q^t) ; \vdash_{\text{ac}} (\neg R^t \vdash R^t)) \)

\{Definition of ; \vdash_{\text{ac}}\}

\( = P ; \vdash_{\text{ac}} (Q) \)

\{Property of designs\}

\( \square \)
Appendix G

Linking theories

G.1 acdash2ac and ac2acdashed

Law G.1.1 (acdashedac-subset)

\[ \text{acdashedac}(t) \subseteq \text{acdashedac}(z) \iff t \subseteq z \]

Proof. To be established. \qed

Law G.1.2 (ac2acdashed-subset)

\[ \text{ac2acdashed}(t) \subseteq \text{ac2acdashed}(z) \iff t \subseteq z \]

Proof. To be established. \qed

Law G.1.3 (acdashedac-\emptyset)

\[ \text{acdashedac}(\emptyset) = \emptyset \]

Proof.

\[
\text{acdashedac}(\emptyset) = \{ z_0 : S_{\text{ma}_P}, z_1 : S_{\text{out}_P} \mid z_0 \in \emptyset \land (\forall x : \alpha P \cdot z_0.x = z_1.(x')) \cdot z_1 \} \quad \text{\{Definition of acdash2ac\}}
\]

\[
= \{ z_0 : S_{\text{ma}_P}, z_1 : S_{\text{out}_P} \mid \text{false} \} \quad \text{\{Property of sets\}}
\]

\[
= \emptyset \quad \text{\{Property of sets\}}
\]

\qed
Law G.1.4 (ac2acdash-∅)

\[ ac2acdash(∅) = ∅ \]

Proof. Similar to that of Law G.1.3

G.2 PBMH

Law G.2.1 Provided \( P \) satisfies PBMH.

\[ P[∅/ac'] \lor P = P \]

Proof.

\[
\begin{align*}
P[∅/ac'] \lor P & \quad \{\text{Assumption: } P \text{ is PBMH-healthy}\} \\
= (P ; ac \subseteq ac')[∅/ac'] \lor (P ; ac \subseteq ac') & \quad \{\text{Substitution}\} \\
= (P ; ac \subseteq ∅) \lor (P ; ac \subseteq ac') & \quad \{\text{Distributivity of sequential composition w.r.t. disjunction}\} \\
= P ; (ac \subseteq ∅ \lor ac \subseteq ac') & \quad \{\text{Property of subset inclusion}\} \\
= P ; (ac \subseteq ac') & \quad \{\text{Assumption: } P \text{ is PBMH-healthy}\} \\
= P & \\
\end{align*}
\]

G.3 Lemmas

Lemma G.3.1

\[
pbmh2d(P) = \left( \neg P[∅/ac'][s/in\alpha] \quad \vdash \quad \exists ac_0 \bullet P[ac_0/ac'][s/in\alpha] \land \text{acdash2ac}(ac_0) \subseteq ac' \land ac' \neq ∅ \right)
\]

Proof.

\[
pbmh2d(P) \quad \{\text{Definition of } pmh2d\}
= \left( \neg P[∅/ac'][s/in\alpha] \quad \vdash \quad \exists ac_0 \bullet P[ac_0/ac'][s/in\alpha] \land \text{acdash2ac}(ac_0) \subseteq ac' \right) \quad \{\text{Predicate calculus}\}
\]
\[
\begin{align*}
\neg P[\emptyset/ac'][s/in\alpha] & \\
\vdash & \\
\exists ac_0 \bullet P[ac_0/ac'][s/in\alpha] \land acdash2 ac(ac_0) \subseteq ac' \\
& \land (ac' = \emptyset \lor ac' \neq \emptyset) \\
\end{align*}
\]
\{Predicate calculus\}

\[
\begin{align*}
\neg P[\emptyset/ac'][s/in\alpha] & \\
\vdash & \\
(\exists ac_0 \bullet P[ac_0/ac'][s/in\alpha] \land acdash2 ac(ac_0) \subseteq ac' \land ac' \neq \emptyset) \\
\lor & \\
(\exists ac_0 \bullet P[ac_0/ac'][s/in\alpha] \land \neg acdash2 ac(ac_0) \subseteq ac' \land ac' = \emptyset) \\
\end{align*}
\]
\{Property of sets\}

\[
\begin{align*}
\neg P[\emptyset/ac'][s/in\alpha] & \\
\vdash & \\
(\exists ac_0 \bullet P[ac_0/ac'][s/in\alpha] \land acdash2 ac(ac_0) \subseteq ac' \land ac' \neq \emptyset) \\
\lor & \\
(\exists ac_0 \bullet P[ac_0/ac'][s/in\alpha] \land \\
\forall z \bullet (z \in acdash2 ac(ac_0) \Rightarrow z \in ac') \land \forall z \bullet z \notin ac') \\
\end{align*}
\]
\{Predicate calculus\}

\[
\begin{align*}
\neg P[\emptyset/ac'][s/in\alpha] & \\
\vdash & \\
(\exists ac_0 \bullet P[ac_0/ac'][s/in\alpha] \land acdash2 ac(ac_0) \subseteq ac' \land ac' \neq \emptyset) \\
\lor & \\
(\exists ac_0 \bullet P[ac_0/ac'][s/in\alpha] \land \\
\forall z \bullet (z \notin acdash2 ac(ac_0) \land z \notin ac') \\
\end{align*}
\]
\{Property of sets\}

\[
\begin{align*}
\neg P[\emptyset/ac'][s/in\alpha] & \\
\vdash & \\
(\exists ac_0 \bullet P[ac_0/ac'][s/in\alpha] \land acdash2 ac(ac_0) \subseteq ac' \land ac' \neq \emptyset) \\
\lor & \\
(\exists ac_0 \bullet P[ac_0/ac'][s/in\alpha] \land \\
\forall z \bullet \left(ac2 acdash \circ acdash2 ac(ac_0) = ac2 acdash(\emptyset) \land ac' = \emptyset\right) \\
\end{align*}
\]
\{Law G.1.4 and Law 5.7.1\}
\[
\begin{align*}
\neg P[\emptyset/ac'][s/in\alpha] \\
\quad \vdash \\
\quad \quad (\exists ac_0 \bullet P[ac_0/ac'][s/in\alpha] \land acdash2ac(ac_0) \subseteq ac' \land ac' \neq \emptyset) \\
\quad \quad \lor \\
\quad \quad (\exists ac_0 \bullet P[ac_0/ac'][s/in\alpha] \land ac_0 = \emptyset \land ac' = \emptyset) \quad \{\text{One-point rule}\}
\end{align*}
\]

\[
\begin{align*}
\neg P[\emptyset/ac'][s/in\alpha] \\
\quad \vdash \\
\quad \quad (\exists ac_0 \bullet P[ac_0/ac'][s/in\alpha] \land acdash2ac(ac_0) \subseteq ac' \land ac' \neq \emptyset) \\
\quad \quad \lor \\
\quad \quad (P[\emptyset/ac'][s/in\alpha] \land ac' = \emptyset) \\
\quad \quad \quad \{\text{Definition of design}\}
\end{align*}
\]

\[
\begin{align*}
(\text{ok} \land \neg P[\emptyset/ac'][s/in\alpha]) \\
\implies \\
\quad \quad (\exists ac_0 \bullet P[ac_0/ac'][s/in\alpha] \land acdash2ac(ac_0) \subseteq ac' \land ac' \neq \emptyset) \\
\quad \quad \lor \\
\quad \quad (P[\emptyset/ac'][s/in\alpha] \land ac' = \emptyset) \\
\quad \quad \quad \quad \{\text{Predicate calculus}\}
\end{align*}
\]

\[
\begin{align*}
\text{ok} \implies \\
\quad (\exists ac_0 \bullet P[ac_0/ac'][s/in\alpha] \land acdash2ac(ac_0) \subseteq ac' \land ac' \neq \emptyset) \\
\quad \lor \\
\quad (P[\emptyset/ac'][s/in\alpha] \land ac' = \emptyset) \\
\quad \quad \quad \quad \quad \quad \quad \{\text{Predicate calculus: absorption law}\}
\end{align*}
\]

\[
\begin{align*}
\text{ok} \implies \\
\quad (\exists ac_0 \bullet P[ac_0/ac'][s/in\alpha] \land acdash2ac(ac_0) \subseteq ac' \land ac' \neq \emptyset) \\
\quad \lor \\
\quad P[\emptyset/ac'][s/in\alpha] \\
\quad \quad \quad \quad \quad \quad \quad \{\text{Predicate calculus and definition of design}\}
\end{align*}
\]

\[
\begin{align*}
\neg P[\emptyset/ac'][s/in\alpha] \\
\quad \vdash \\
\quad \quad (\exists ac_0 \bullet P[ac_0/ac'][s/in\alpha] \land acdash2ac(ac_0) \subseteq ac' \land ac' \neq \emptyset)
\end{align*}
\]