An analysis of total correctness refinement models for partial relation semantics I

MOSHE DEUTSCH, MARTIN C. HENSON, *Department of Computer Science, University of Essex, Wivenhoe Park, Colchester, Essex CO4 3SQ, UK*
E-mail: {mdeuts, hensm}@essex.ac.uk

STEVE REEVES, *Department of Computer Science, University of Waikato, Private Bag 3105, Hamilton, New Zealand*
E-mail: stever@cs.waikato.ac.nz

Abstract

This is the first of a series of papers devoted to the thorough investigation of (total correctness) refinement based on an underlying partial relational model. In this paper we restrict attention to operation refinement. We explore four theories of refinement based on an underlying partial relation model for specifications, and we show that they are all equivalent. This, in particular, sheds some light on the relational completion operator (lifted-totalisation) due to Woodcock which underlies data refinement in, for example, the specification language Z. It further leads to two simple alternative models which are also equivalent to the others.

Keywords: operation refinement, specification language, specification logic

1 Introduction

In this paper we provide a thorough investigation of total correctness[1] operation refinement (that is the degenerate case of data refinement in which simulations are identity functions) where the underlying semantics of specifications is given in terms of partial relations. A major example of such a semantics is the specification language Z, in which a specification denotes a set of bindings which can be construed to be a partial relation between input sub-bindings and output sub-bindings (see [18], [11], [6] and [8] for accounts of Z logic and semantics along these lines). Woodcock, in the book [18] (chapter 16), describes a general account of data refinement in which a relational operator (which we will call the *Woodcock-completion*) plays the crucial underlying role.[2] This paper can be construed, at one level, to be providing a detailed exposition and mathematical account of that relational completion operation by providing a systematic and comprehensive comparison with other possible approaches. In addition, since the theme of *combining* formal methods is now an important re-

[1] Now that we have established that we are dealing with total correctness, we will drop this qualification: unless we specify otherwise, we are interested only in total correctness refinement.

[2] [5] (chapter 8) also introduces a similar idea.

search area, this paper can be seen as contributing both results and methodological techniques which are of use when combining systems with different semantic bases.
In order that our general analysis of partial relations can be extended to Z in particular (which will permit in due course an investigation of the *schema calculus*) we will base our mathematical investigation on the logic $\mathcal{Z_C}$ described in [11][3]. This is an extension of higher order logic with schema types and filtered bindings which can be extended conservatively to cover the key features of Z. We will need, in fact, a theory extension of $\mathcal{Z_C}$ which includes additional constants and axioms in order to provide an account of the Woodcock-completion. The base logic and the theory extension, are presented for completeness in appendix A.
Our investigation is essentially proof-theoretic. We have found that the connections and relationships between various theories of refinement can be most clearly demonstrated in this mathematical style. Even where the underlying notions of refinement are model-theoretic, we will nevertheless express them as theories: they are characterised by sets of introduction and elimination rules. This permits a particularly straightforward strategy for demonstrating that theories are equivalent, or related, which is quite general. This strategy is explained in section 4 below.
The paper is structured in the following way. We first revise Z logic, setting some notation and outlining this important application area for the partial relation semantics. We then move on, in section 3, to provide four theories of refinement, which are *prima facie* distinct, based on various alternative semantics for specifications: the first is based on sets of implementations, the second on properties of preconditions and postconditions, the third on a relational completion operator, and the last on weakest preconditions. In section 4 we go on to demonstrate that all four of these theories are equivalent. We are led to introduce further models in sections 5 and 6 which are motivated by our earlier results. In addition to our conclusions, acknowledgements and references, we conclude with an appendix setting out the mathematical basis for the entire investigation.

2 Preliminaries

In this first section we will revise a little Z logic, settling some notational conventions in the process.[4]
In [11] Z schemas, and operation schemas in particular, were formalised as sets of bindings. This captures the informal account to be found in the literature (*e.g.* [7], [18]). In this paper we will use the meta-variable U (with decorations) to range over operation schema. As an example, consider the operation schema:

[3]Unlike our earlier papers, [11] attempts to formalise Z as it is informally understood.

[4]Further and additional detail, including a summary of the ideas introduced here, can be found in appendix A and in [11]. The appendix covers, in addition to a description of the core logic $\mathcal{Z_C}$ and the theory $\mathcal{Z}_\mathcal{C}^\perp$, a section demonstrating that $\mathcal{Z}_\mathcal{C}^\perp$ is conservative over $\mathcal{Z_C}$. Our objective is to make the appendix here sufficiently complete so that it can serve to support the other papers in this series in the future. All the mathematics in the main body of this paper takes place in the extended theory $\mathcal{Z}_\mathcal{C}^\perp$ which is described in the appendix. In certain circumstances (*e.g.* sections 3.2 and 3.4 below) some theories can be formalised in the core logic $\mathcal{Z_C}$; since this is technically significant, we will indicate the fact explicitly.

$$\begin{array}{|l}
\multicolumn{1}{l}{\textit{Ex}_0} \\
\mathtt{x}, \mathtt{x}' : \mathbb{N} \\
\hline
\mathtt{x} = 0 \\
\mathtt{x}' < 10 \\
\hline
\end{array}$$

Ex_0 has the the type $\mathbb{P}[\mathtt{x} : \mathbb{N}, \mathtt{x}' : \mathbb{N}]$, and is understood to be a set of bindings of schema type $[\mathtt{x} : \mathbb{N}, \mathtt{x}' : \mathbb{N}]$. Recall that unprimed labels (such as $\mathtt{x}$) are understood to be observations of the state before the operation, whereas primed labels (such as $\mathtt{x}'$) are observations of the state afterwards. Each operation schema U will have a type of the form $\mathbb{P}\ T$ where T is a schema type. The type T can, additionally, always be partitioned as the (compatible) union of its input (or before) type T^{in}, and its output (or after) type $T^{out'}$. That is, $T = T^{in} \curlyvee T^{out'}$. For the schema Ex_0 we have $T^{in} = [\mathtt{x} : \mathbb{N}]$ and $T^{out'} = [\mathtt{x}' : \mathbb{N}]$. In this paper, since we are only dealing with operation refinement, we can assume that all operation schemas have the type $\mathbb{P}\ T$ where $T = T^{in} \curlyvee T^{out'}$. With this in place we can omit the type superscripts in most places in the sequel.

Definition 2.1 (semantics for atomic schemas)

$$[T \mid P] =_{df} \{z \in T \mid z.P\}$$

Note that this definition[5] draws bindings from the *natural carrier* of the type T. As a consequence, writing $t(\perp)$ for any binding which projects an instance of the constant $\perp$ (that is: $t.\mathtt{x} = \perp$ for some observation $\mathtt{x}$), we have:

Lemma 2.2

$$\frac{t(\perp) \in U}{\mathit{false}}$$

$\blacksquare$

The bindings $\langle\!| \mathtt{x} \Rrightarrow 0, \mathtt{x}' \Rrightarrow n |\!\rangle$, where $n < 10$, are all elements of Ex_0. In fact there are no other elements in this case. We can introduce the idea of the *precondition* of the schema (domain of the relation the schema denotes) to express the partiality involved.

Definition 2.3 Let $T^{in} \preceq V$.

$$\mathit{Pre}\ U\ x^V =_{df} \exists z \in U \bullet x =_{T^{in}} z$$

Proposition 2.4 The following introduction and elimination rules are immediately derivable for preconditions:

$$\frac{t_0 \in U \quad t_0 =_{T^{in}} t_1}{\mathit{Pre}\ U\ t_1} \qquad \frac{\mathit{Pre}\ U\ t \quad y \in U, y =_{T^{in}} t \vdash P}{P}$$

where y is fresh. $\blacksquare$

[5] In the technical development it is usual to write the operation schema horizontally, with a vertical line separating the declarations from the predicate.

Clearly, the precondition of Ex_0 is not (and for operation schemas in general, will not be) the whole of $[\mathtt{x} : \mathbb{N}]$ (in general, T^{in}). In this sense operation schemas denote partial relations.
Naturally, an immediate question will arise: what does it mean for one operation schema to refine another? More generally, we are asking: what does it mean for one partial relation to refine another?[6]

3 A basic analysis of refinement

We begin by introducing four distinct notions of refinement based on four distinct answers to the questions above and then we go on to compare them. This serves to illuminate them all, particularly the notion based on the Woodcock-completion which is the *de facto* standard for Z.

3.1 F-refinement

To a logician, a specification resembles a theory; so a natural question is: what are the models of the theory? A computer scientist may ask a closely related question: when is a program an implementation of the specification? We will, in this section, consider deterministic programs and model them as (total) functions. We do this, via a standard expedient of introducing a special value $\perp$, which might represent unwelcome behaviour. Such behaviour might be non-termination but it need not, and nothing we shall do with this value commits us to that interpretation. We shall pronounce $\perp$ the *abortive* value. In [18] the authors refer to it as *undefined* which is, we feel, unfortunate in the context of partial relations; we will have much more to say about this below in section 4.4. In order to introduce this value into the analysis, the technical development below takes place in the extended theory $\mathcal{Z}_C^{\perp}$.
From the logical perspective, we are interested in all the models of a theory, so given a putative model g and a theory U, we would be inclined to write:

$$g \models U$$

to represent the statement that g is a model of U. Within our application area in computer science we might prefer to read this as a relation of *implementation.* To signal this interpretation we shall in fact write this judgement as:

$$g \Subset U$$

to be pronounced "g implements (is an implementation of) U".
Now an operation schema has been modelled as a partial relation, so we need to consider how an implementation behaves outside the precondition of the schema (domain of the underlying relation). There are degrees of freedom, but we shall, in this paper, permit what we call *chaotic models.*[7] More exactly, we understand silence in the specification to be permission for an implementation to behave in any arbitrary manner,

[6] If we were not interested in total correctness, we could simply take the subset relation on the partial relations as refinement, and we would then obtain a very well-behaved theory which also makes the schema calculus fully monotonic with respect to refinement but necessarily treats preconditions as triggers (firing conditions).

[7] There is an obvious alternative, based on *abortive models* or what in [10] is effectively the *partial model*; we will say a little more in section 8 and it will be investigated in our future work.

including the abortive behaviour $\perp$. In other respects we will expect a model (an implementation) to produce a result which is in the relation whenever it is supplied with an input inside the precondition. This leads to the following definition of the modelling (implementing) relation.

Definition 3.1

$$g \Subset_f U =_{df} (\forall z \in T_\perp^{in} \bullet Pre\ U\ z \Rightarrow z \star (g\ z)' \in U) \wedge g \in T_\perp^{in} \rightarrow T_\perp^{out'}$$

Then we can prove the following.

Proposition 3.2 The following introduction and elimination rules are derivable:

$$\frac{z \in T_\perp^{in}, Pre\ U\ z \vdash z \star (g\ z)' \in U \quad g \in T_\perp^{in} \rightarrow T_\perp^{out'}}{g \Subset_f U}\ (\Subset_f^+)$$

where z is a fresh variable.

$$\frac{g \Subset_f U \quad Pre\ U\ t \quad t \in T_\perp^{in}}{t \star (g\ t)' \in U}\ (\Subset_{f_0}^-) \qquad \frac{g \Subset_f U}{g \in T_\perp^{in} \rightarrow T_\perp^{out'}}\ (\Subset_{f_1}^-)$$

■

This is sufficient technical development to allow us to explore refinement. We can answer the question: when is U_0 a refinement of U_1? A reasonable answer is: when any implementation of U_0 is also an implementation of U_1. After all, we wish to be able to replace any specification U_1 by its refinement U_0, and if all potential implementations of the latter are implementations of this former we are quite safe. Thus we are led to:

Definition 3.3

$$\widehat{U} =_{df} \{z \mid z \Subset_f U\}$$

Then we have F-refinement ("F" for function).

Definition 3.4

$$U_0 \sqsupseteq_f U_1 =_{df} \widehat{U_0} \subseteq \widehat{U_1}$$

Obvious introduction and elimination rules for F-refinement follow from this definition.

3.2 S-refinement

In this section we introduce a purely proof theoretic characterisation of refinement, which is closely connected to refinement as introduced by Spivey in, for example, [16] and as discussed in [13] and [15]. In those contexts we do not so much have an alternative notion of refinement as two sufficient conditions (essentially the premises of the introduction rule in proposition 3.5 below). By adding the two elimination rules we add necessary conditions, and thus formalise an independent theory. There is also a connection with theorem 3.1.2 of [12] (page 77), although that analysis concerns

the two-predicate *designs* of the refinement calculus (syntactic preconditions) rather than the single predicate specifications of a language like Z (logical preconditions).
This notion is based on two basic observations regarding the properties one expects in a refinement: firstly, that a refinement may involve the reduction of non-determinism, secondly that a refinement may involve the expansion of the domain of definition. Put another way, we have a refinement providing that *postconditions do not weaken* (we do not permit an increase in non-determinism in a refinement) and that *preconditions do not strengthen* (we do not permit requirements in the domain of definition to disappear in a refinement).
This notion can be captured by forcing the refinement relation to hold *exactly* when these conditions apply. S-refinement, named for Mike Spivey, is written $U_0 \sqsupseteq_s U_1$ and is given by the definition that leads directly to the following rules:

Proposition 3.5 Let z, z_0, z_1 be fresh variables.

$$\frac{Pre\ \ U_1\ z \vdash Pre\ \ U_0\ z \quad Pre\ \ U_1\ z_0, z_0 \star z_1' \in U_0 \vdash z_0 \star z_1' \in U_1}{U_0 \sqsupseteq_s U_1}\ (\sqsupseteq_s^+)$$

$$\frac{U_0 \sqsupseteq_s U_1 \quad Pre\ \ U_1\ t}{Pre\ \ U_0\ t}\ (\sqsupseteq_{s_0}^-)$$

$$\frac{U_0 \sqsupseteq_s U_1 \quad Pre\ \ U_1\ t_0 \quad t_0 \star t_1' \in U_0}{t_0 \star t_1' \in U_1}\ (\sqsupseteq_{s_1}^-)$$

■

This theory does not depend on, and makes no reference to, the $\perp$ value. It can be formalised in the core theory $\mathcal{Z}_\mathcal{C}$.

3.3 W$_\bullet$-refinement

Our third notion of refinement is taken directly from the literature [18]. It is based on a relational completion operator due to Woodcock. For notational convenience we will write $T^\star$ for the set $T_\perp^{in} \star T_\perp^{out'}$.
The *lifted totalisation* of a set of bindings (Woodcock-completion) can be defined as follows:

Definition 3.6

$$\overset{\bullet}{U} =_{df} \{z_0 \star z_1' \in T^\star \mid Pre\ \ U\ z_0 \Rightarrow z_0 \star z_1' \in U\}$$

Proposition 3.7 The following introduction and elimination rules are derivable for lifted totalised sets:

$$\frac{t_0 \star t_1' \in T^\star \quad Pre\ \ U\ t_0 \vdash t_0 \star t_1' \in U}{t_0 \star t_1' \in \overset{\bullet}{U}}\ (\bullet^+)$$

and:

$$\frac{t_0 \star t_1' \in \overset{\bullet}{U} \quad Pre\ \ U\ t_0}{t_0 \star t_1' \in U}\ (\bullet_0^-) \qquad \frac{t_0 \star t_1' \in \overset{\bullet}{U}}{t_0 \star t_1' \in T^\star}\ (\bullet_1^-)$$

■

Note that it is, sometimes, useful to use the following version of $(\bullet_o^-)$ rule, which is based upon disjunction elimination, rather than implication elimination.

Proposition 3.8

$$\dfrac{t_0 \star t_1' \in \overset{\bullet}{U} \qquad \neg\, Pre\; U\; t_0 \vdash P \qquad t_0 \star t_1' \in U \vdash P}{P}\;(\bullet_o^-)$$

∎

Lemma 3.9 The following are derivable:

$$\dfrac{}{U \subseteq \overset{\bullet}{U}}\;(i) \qquad \dfrac{}{\bot \in \overset{\bullet}{U}}\;(ii) \qquad \dfrac{\neg Pre\;\; U\;\; t_0 \qquad t_0 \in T_\bot^{in} \qquad t_1' \in T_\bot^{out'}}{t_0 \star t_1' \in \overset{\bullet}{U}}\;(iii)$$

Proof. (i) is trivial. For (ii), consider the following derivation:

$$\dfrac{\dfrac{}{\bot \in T^\star} \qquad \dfrac{\dfrac{}{Pre\;\; U\;\; \bot}\,(1) \qquad \dfrac{\dfrac{\dfrac{}{y \in U}\,(2) \qquad \dfrac{}{y =_{T^{in}} \bot}\,(2)}{false}\,(2.2)}{\bot \in U}}{\bot \in U}\,(2)}{\bot \in \overset{\bullet}{U}}\,(1)$$

For (iii), consider the following derivation:

$$\dfrac{\dfrac{t_0 \in T_\bot^{in} \qquad t_1' \in T_\bot^{out'}}{t_0 \star t_1' \in T^\star} \qquad \dfrac{\dfrac{\neg Pre\; U\; t_0 \qquad \dfrac{}{Pre\; U\; t_0}\,(1)}{false}}{t_0 \star t_1' \in U}}{t_0 \star t_1' \in \overset{\bullet}{U}}\,(1)$$

∎

Lemmas 3.9(i), (ii) and (iii) demonstrate that definition 3.6 is consistent with the intentions described in [18] chapter 16: the underlying partial relation is contained in the completion, the abortive element is present in the relation, and more generally, each value outside the precondition maps to every value in the co-domain of the relation. Furthermore, the following rules, which are derived from lemma 3.9(iii), embody the *non-strict* lifting with respect to the abortive element and the fact that everything outside the precondition is mapped onto the abortive value (as well as everything else in the co-domain of the relation).

Corollary 3.10

$$\dfrac{t' \in T_\bot^{out'}}{\bot \star t' \in \overset{\bullet}{U}}\;(i) \qquad \dfrac{\neg\, Pre\; U\; t \qquad t \in T_\bot^{in}}{t \star \bot' \in \overset{\bullet}{U}}\;(ii)$$

∎

$W_\bullet$-refinement, written $U_0 \sqsupseteq_{w_\bullet} U_1$, and named for Jim Woodcock, is defined as follows.

Definition 3.11

$$U_0 \sqsupseteq_{w_\bullet} U_1 =_{df} \overset{\bullet}{U_0} \subseteq \overset{\bullet}{U_1}$$

Obvious introduction and elimination rules follow from this.

3.4 WP-refinement

Our final theory of refinement is that based on a *weakest-precondition* interpretation.[8] In order to formalise this we begin by introducing a notion of *postcondition* to complement the precondition we introduced earlier.

Definition 3.12

$$Post\ U\ z_0 =_{df} \{z_1' \mid z_0 \star z_1' \in U\}$$

Note that this introduces a set, rather than a predicate.
With this in place we can introduce the weakest precondition interpretation of an operation schema. Again, the specified postcondition (C in the definition below) is expressed as a set rather than as a predicate.

Definition 3.13

$$wp\ U\ C =_{df} \{z \mid Pre\ U\ z \wedge Post\ U\ z \subseteq C\}$$

The reason why we choose to work with sets, rather than predicates, is simply that it casts the technical material in a similar style to the models we introduced earlier for F-refinement and W$_\bullet$-refinement, which also construct sets from the underlying partial relations.

Proposition 3.14 The following introduction and elimination rules for the weakest precondition of U are derivable:

$$\frac{Pre\ U\ t \quad z' \in Post\ U\ t \vdash z' \in C}{t \in wp\ U\ C}$$

where z is a fresh variable.

$$\frac{t \in wp\ U\ C}{Pre\ U\ t} \qquad \frac{t_0 \in wp\ U\ C \quad t_1' \in Post\ U\ t_0}{t_1' \in C}$$

■

[8] A weakest precondition semantics for Z is provided in [3], a paper based on the semantics of Z to be found in the (then) draft Z standard [1] (now superseded by [8]). It would be very interesting to investigate the relationship between the two approaches, but that is beyond the scope of this paper. In passing we note that they provide an interpretation over the syntax of Z (atomic operation schema expressions) whereas we opt for one over the partial relational semantics (sets of bindings). Generality (an interpretation over *all* schema expressions) is obtained in two significantly different ways: in ours it follows since all schema expressions denote sets of bindings through the semantics; [3] on the other hand relies on the fact that all schema expressions may be written in the form of an atomic schema. That, in turn, relies on the standard equational logic of schemas. We make some further remarks about our semantic approach, and the equational logic, in section 8. These considerations will become even more significant when, in future work, we examine monotonicity issues and what would be necessary to obtain fully monotonic schema calculi for Z.

We can now define WP-refinement.

Definition 3.15

$$U_0 \sqsupseteq_{wp} U_1 =_{df} \forall C^{\mathbb{P}\, T^{out'}} \bullet wp\ U_1\ C \subseteq wp\ U_0\ C$$

Proposition 3.16 The following introduction and elimination rules for WP-refinement are derivable:

$$\frac{z \in wp\ U_1\ C \vdash z \in wp\ U_0\ C}{U_0 \sqsupseteq_{wp} U_1}\ (\sqsupseteq^{+}_{wp})$$

where z and C are fresh variables.

$$\frac{U_0 \sqsupseteq_{wp} U_1 \quad t \in wp\ U_1\ C}{t \in wp\ U_0\ C}\ (\sqsupseteq^{-}_{wp})$$

■

Lemma 3.17

$$\frac{Pre\ U\ t}{t \in wp\ U\ (Post\ U\ t)}$$

PROOF.

$$\frac{Pre\ U\ t \quad \dfrac{}{z' \in Post\ U\ t}\ (1)}{t \in wp\ U\ (Post\ U\ t)}\ (1)$$

Since the reverse direction always holds, we have established that $t \in wp\ U\ (Post\ U\ t)$ and $Pre\ U\ t$ are equivalent. ■

These interpretations will be reminiscent from accounts such as [14], although it is unusual to provide a proof-theoretic presentation in terms of introduction and elimination rules. In addition, in [14], the preconditions and postconditions of specification statements are syntactic (explicitly given), rather than logical (implicitly given) as they are in Z. Note that our interpretation does not involve the $\perp$ values, and could therefore be formalised in $\mathcal{Z}_{\mathcal{C}}$.

3.5 Review

What is the relationship between these four notions of refinement? In particular, can an exploration of that question shed any light on why the Woodcock-completion has been defined in just the way it has? What, in particular, is the role of the value $\perp$? Why is the lifting process non-strict with respect to the abortive value? We will begin with the first of these questions.

4 Four equivalent theories

In this section we demonstrate that our four theories of refinement are all equivalent. In doing this we will see clearly the critical role that the $\perp$ value plays.
We shall be showing that all judgements of refinement in one theory are contained among the refinements sanctioned by another. Such results will always be established proof-theoretically. Specifically we will show that the refinement relation of a theory

T_0 satisfies the elimination rule (or rules) for refinement of another theory T_1. Since the elimination rules and introduction rules of a theory enjoy the usual symmetry property, this is sufficient to show that all T_0-refinements are also T_1 refinements.[9]

4.1 F-refinement and W$_\bullet$-refinement are equivalent (in $\mathcal{Z}_{\mathcal{C}}^{\perp}$ + AC)

4.1.1 R-refinement

We begin this analysis by defining, by way of an intermediate stage, the set of total functions compatible with an operation schema. This forms a bridge between F-refinement and W$_\bullet$-refinement.

Definition 4.1

$$\overline{U} =_{df} \{ z \in T_{\perp}^{in} \rightarrow T_{\perp}^{out'} \mid z \subseteq \overset{\bullet}{U} \}$$

Then we have:

Definition 4.2

$$g \in_r U =_{df} g \in \overline{U}$$

And then R-refinement is simply: $U_0 \sqsupseteq_r U_1 =_{df} \overline{U_0} \subseteq \overline{U_1}$ with the usual introduction and elimination rules.

4.1.2 R-refinement and W$_\bullet$-refinement are equivalent

We begin by showing that R-refinement satisfies the W$_\bullet$-refinement elimination rule.

Proposition 4.3 The following rule is derivable:

$$\frac{U_0 \sqsupseteq_r U_1 \quad t \in \overset{\bullet}{U_0}}{t \in \overset{\bullet}{U_1}}$$

PROOF. The proof requires the axiom of choice (see the step labelled (AC) below).

$$\frac{\dfrac{t \in \overset{\bullet}{U_0}}{\exists\, g \in T_{\perp}^{in} \rightarrow T_{\perp}^{out'} \bullet t \in g \wedge g \subseteq \overset{\bullet}{U_0}}\,(AC) \qquad \begin{array}{c}\delta \\ \vdots \\ t \in \overset{\bullet}{U_1}\end{array}}{t \in \overset{\bullet}{U_1}}\,(1)$$

[9]An alternative strategy would be to show that a similar property holds for the introduction rule. In the refinement theories we consider there is only ever a single introduction rule, and this suggests that this might be a more efficient approach. However, there are as many premises to the introduction rule as there are distinct elimination rules, so in the end the amount of work involved is essentially the same. Moreover, by considering the elimination rules separately, we can in some cases (see for example sections 4.2 and 4.4 below) isolate particular properties and reasons underlying equivalence (or non-equivalence in other circumstances) which highlight particular issues of interest.

Where δ is:

$$\dfrac{\dfrac{\dfrac{U_0 \sqsupseteq_r U_1 \qquad \dfrac{\dfrac{}{y \in T_\perp^{in} \to T_\perp^{out'}}\,(1) \qquad \dfrac{}{y \subseteq \overset{\bullet}{U_0}}\,(1)}{y \in \overline{U_0}}}{y \in \overline{U_1}}}{y \subseteq \overset{\bullet}{U_1}} \qquad \dfrac{}{t \in y}\,(1)}{t \in \overset{\bullet}{U_1}}$$

■

From this we have:

Theorem 4.4

$$\dfrac{U_0 \sqsupseteq_r U_1}{U_0 \sqsupseteq_{w_\bullet} U_1}$$

Proof. This follows immediately, by $(\sqsupseteq_{w_\bullet}^{+})$, from proposition 4.3.[10] ■

We now show that W$_\bullet$-refinement satisfies the R-refinement elimination rule.

Proposition 4.5

$$\dfrac{U_0 \sqsupseteq_{w_\bullet} U_1 \quad g \in \overline{U_0}}{g \in \overline{U_1}}$$

Proof.

$$\dfrac{\dfrac{g \in \overline{U_0}}{g \in T_\perp^{in} \to T_\perp^{out'}} \qquad \dfrac{\dfrac{U_0 \sqsupseteq_{w_\bullet} U_1 \qquad \dfrac{\dfrac{g \in \overline{U_0}}{g \subseteq \overset{\bullet}{U_0}} \quad \dfrac{}{t \in g}\,(1)}{t \in \overset{\bullet}{U_0}}}{t \in \overset{\bullet}{U_1}}\,(1)}{g \subseteq \overset{\bullet}{U_1}}}{g \in \overline{U_1}}$$

■

Theorem 4.6

$$\dfrac{U_0 \sqsupseteq_{w_\bullet} U_1}{U_0 \sqsupseteq_r U_1}$$

■

Theorems 4.4 and 4.6 together demonstrate that W$_\bullet$-refinement and R-refinement are equivalent.

[10]The proofs of such theorems are always automatic by the structural symmetry between introduction and elimination rules. We shall not give them in future.

4.1.3 R-refinement and F-refinement are equivalent

In this case we show that the notions of *implementation* (rather than refinement) are equivalent by the same strategy involving elimination rules. We first establish that F-implementation implies R-implementation:

Proposition 4.7 The following rules are derivable:

$$\dfrac{g \in_f U}{g \subseteq \overset{\bullet}{U}} \qquad \dfrac{g \in_f U}{g \in T^{in}_{\perp} \to T^{out'}_{\perp}}$$

Proof.

$$\dfrac{\dfrac{g \subseteq \dfrac{\dfrac{\dfrac{g \in_f U}{g \in T^{in}_{\perp} \to T^{out'}_{\perp}} \quad \dfrac{}{z_0 \star z_1' \in g}(1)}{z_0 \star z_1' \in T^{\star}} \quad \dfrac{\dfrac{g \in_f U \quad \dfrac{}{Pre\ \ U\ \ z_0}(2) \quad \dfrac{\dfrac{}{Pre\ \ U\ \ z_0}(2)}{z_0 \in T^{in}_{\perp}}}{z_0 \star (g\ z_0)' \in U} \quad \begin{matrix}\delta \\ \vdots\end{matrix}}{z_0 \star z_1' \in U}}{z_0 \star z_1' \in \overset{\bullet}{U}}(2)}{}}{g \subseteq \overset{\bullet}{U}}(1)$$

where δ is:

$$\dfrac{\dfrac{g \in_f U}{g \in T^{in}_{\perp} \to T^{out'}_{\perp}} \quad \dfrac{}{z_0 \star z_1' \in g}(1)}{z_1' = (g\ \ z_0)'}$$

The second rule is immediate. ■

Theorem 4.8

$$\dfrac{g \in_f U}{g \in_r U}$$

■

Now we show that R-implementation implies F-implementation.

Proposition 4.9

$$\dfrac{g \in_r U \quad Pre\ \ U\ \ t \quad t \in T^{in}_{\perp}}{t \star (g\ \ t)' \in U} \qquad \dfrac{g \in_r U}{g \in T^{in}_{\perp} \to T^{out'}_{\perp}}$$

Proof.

$$\dfrac{\dfrac{\dfrac{g \in_r U}{g \subseteq \overset{\bullet}{U}} \quad \dfrac{\dfrac{g \in_r U}{g \in T^{in}_{\perp} \to T^{out'}_{\perp}} \quad t \in T^{in}_{\perp}}{t \star (g\ \ t)' \in g}}{t \star (g\ \ t)' \in \overset{\bullet}{U}} \qquad Pre\ \ U\ \ t}{t \star (g\ \ t)' \in U}$$

The second rule is immediate. ■

Theorem 4.10

$$\frac{g \Subset_r U}{g \Subset_f U}$$

∎

Then, from 4.8 and 4.10, we see that the two notions of implementation are equivalent. Hence, so are the two notions of refinement.

4.2 W$_\bullet$-refinement and S-refinement are equivalent

We begin by showing that W$_\bullet$-refinement satisfies the two S-refinement elimination rules. Firstly the rule for preconditions.

Proposition 4.11 The following rule is derivable:

$$\frac{U_0 \sqsupseteq_{w_\bullet} U_1 \quad Pre\ U_1\ t}{Pre\ U_0\ t}$$

Proof. Consider the following derivation:

$$\dfrac{\dfrac{U_0 \sqsupseteq_{w_\bullet} U_1 \qquad \dfrac{\dfrac{}{\neg Pre\ U_0\ t}\,(1) \quad \dfrac{Pre\ U_1\ t}{t \in T_\perp^{in}}}{t\star \perp' \in \overset{\bullet}{U_0}}\,(3.10(ii))}{t\star \perp' \in \overset{\bullet}{U_1}} \qquad Pre\ U_1\ t}{\dfrac{\dfrac{t\star \perp' \in U_1}{false}\,(2.2)}{Pre\ U_0\ t}\,(1)}$$

∎

Turning now to the second elimination rule in S-refinement.

Proposition 4.12 The following rule is derivable:

$$\frac{U_0 \sqsupseteq_{w_\bullet} U_1 \quad Pre\ U_1\ t_0 \quad t_0 \star t_1' \in U_0}{t_0 \star t_1' \in U_1}$$

Proof.

$$\dfrac{\dfrac{U_0 \sqsupseteq_{w_\bullet} U_1 \quad \dfrac{t_0 \star t_1' \in U_0}{t_0 \star t_1' \in \overset{\bullet}{U_0}}\,(3.9(i))}{t_0 \star t_1' \in \overset{\bullet}{U_1}} \qquad Pre\ U_1\ t_0}{t_0 \star t_1' \in U_1}$$

∎

Theorem 4.13

$$\frac{U_0 \sqsupseteq_{w_\bullet} U_1}{U_0 \sqsupseteq_s U_1}$$

∎

We now show that S-refinement satisfies the $W_\bullet$-elimination rule.

Proposition 4.14

$$\frac{U_0 \sqsupseteq_s U_1 \quad t_0 \star t_1' \in \overset{\bullet}{U_0}}{t_0 \star t_1' \in \overset{\bullet}{U_1}}$$

Proof.

$$\dfrac{\dfrac{t_0 \star t_1' \in \overset{\bullet}{U_0}}{t_0 \star t_1' \in T^\star} \quad \dfrac{U_0 \sqsupseteq_s U_1 \quad \overline{Pre\ U_1\ t_0}^{\,(1)} \quad \dfrac{t_0 \star t_1' \in \overset{\bullet}{U_0} \quad \dfrac{U_0 \sqsupseteq_s U_1 \quad \overline{Pre\ U_1\ t_0}^{\,(1)}}{Pre\ U_0\ t_0}}{t_0 \star t_1' \in U_0}}{t_0 \star t_1' \in U_1}}{t_0 \star t_1' \in \overset{\bullet}{U_1}}\ (1)$$

∎

Theorem 4.15

$$\frac{U_0 \sqsupseteq_s U_1}{U_0 \sqsupseteq_{w_\bullet} U_1}$$

∎

Theorems 4.13 and 4.15 together establish that the theories of S-refinement and $W_\bullet$-refinement are equivalent.

4.3 WP-refinement and S-refinement are equivalent

We begin by showing that WP-refinement satisfies the two S-refinement elimination rules. In these results we will often use the fact that $t_1' \in Post\ U\ t_0$ and $t_0 \star t_1' \in U$ are equivalent without further comment. First we have the rule for preconditions.

Proposition 4.16

$$\frac{U_0 \sqsupseteq_{wp} U_1 \quad Pre\ U_1\ t}{Pre\ U_0\ t}$$

Proof. Consider the following derivation:

$$\dfrac{\dfrac{U_0 \sqsupseteq_{wp} U_1 \quad \dfrac{Pre\ U_1\ t}{t \in wp\ U_1\ (Post\ U_1\ t)}\ (3.17)}{t \in wp\ U_0\ (Post\ U_1\ t)}}{Pre\ U_0\ t}$$

∎

Now the second elimination rule.

Proposition 4.17 The following rule is derivable:

$$\frac{U_0 \sqsupseteq_{wp} U_1 \quad Pre\ U_1\ t_0 \quad t_0 \star t_1' \in U_0}{t_0 \star t_1' \in U_1}$$

Proof. Consider the following derivation:

$$\dfrac{\dfrac{U_0 \sqsupseteq_{wp} U_1 \qquad \dfrac{Pre\ U_1\ t_0}{t_0 \in wp\ U_1\ (Post\ U_1\ t_0)}\,(3.17)}{t_0 \in wp\ U_0\ (Post\ U_1\ t_0)} \qquad t_0 \star t_1' \in U_0}{t_0 \star t_1' \in U_1}$$

■

Theorem 4.18

$$\dfrac{U_0 \sqsupseteq_{wp} U_1}{U_0 \sqsupseteq_s U_1}$$

■

We now show that every S-refinement is a WP-refinement.

Proposition 4.19

$$\dfrac{U_0 \sqsupseteq_s U_1 \quad t \in wp\ U_1\ C}{t \in wp\ U_0\ C}$$

Proof. Consider the following derivation:

$$\dfrac{\dfrac{U_0 \sqsupseteq_s U_1 \qquad \dfrac{t \in wp\ U_1\ C}{Pre\ U_1\ t}}{Pre\ U_0\ t} \qquad \dfrac{t \in wp\ U_1\ C \qquad \dfrac{U_0 \sqsupseteq_s U_1 \qquad \dfrac{t \in wp\ U_1\ C}{Pre\ U_1\ t} \qquad \dfrac{}{t \star t_0' \in U_0}\,(1)}{t \star t_0' \in U_1}}{t_0' \in C}}{t \in wp\ U_0\ C}\,(1)$$

■

Theorem 4.20

$$\dfrac{U_0 \sqsupseteq_s U_1}{U_0 \sqsupseteq_{wp} U_1}$$

■

Theorems 4.18 and 4.20 establish that WP-refinement and S-refinement are equivalent.

4.4 Review

The model of schemas introduced in $W_\bullet$-refinement not only totalises the schema as a set of bindings, it also introduces the $\perp$ values and extends the domains and co-domains accordingly. The totalisation then stipulates chaotic behaviour outside the precondition and additionally for the $\perp$ values.
Why is it necessary to include the new values? What are the consequences of totalisation *without* lifting?
In [18], the authors explicitly discuss these questions.[11] By way of explanation they consider the particular schema:

[11] They call the $\perp$ value "undefined", which is perhaps unfortunate since this is also used with reference to values outside the domain of definition of particular schemas. We will continue to refer to call it the "abortive" value.

κ

```
x, x' : ℕ
─────────
x' = 0
```

This denotes a total constant relation in the model. They then illustrate carefully that lifting ensures that schema composition is *strict* with respect to chaotic behaviour.[12] On the other hand, totalisation without lifting leads to non-strict recovery from chaos. First we introduce the chaotic specification.

Definition 4.21

$$Chaos =_{df} [T \mid false]$$

Now we describe the *non-lifted* totalisation, by ensuring that the values are drawn only from natural carrier set rather than the extension including the abortive value.

Definition 4.22

$$\overset{\diamond}{U} =_{df} \{z \in T \mid Pre\ U\ z \Rightarrow z \in U\}$$

Proposition 4.23 (Woodcock and Davies)

$$\begin{array}{llll} (i) & \overset{\bullet}{Chaos} \mathbin{⨾} \overset{\bullet}{\kappa} & =_{df} & \overset{\bullet}{Chaos} \\ (ii) & \overset{\diamond}{Chaos} \mathbin{⨾} \overset{\diamond}{\kappa} & =_{df} & \kappa \end{array}$$

Proof. See [18], page 238. ∎

It should be noted that the second of these results is contingent on the particular choice κ: it is not true in general. But this observation notwithstanding, the interpretation of the results requires care. *Chaos* is described as representing undefinedness, or a run-time error being encountered whatever the initial value. This is odd since, in particular, *Chaos* permits the input $\perp$ to result in a well-defined output (lifting is non-strict with respect to $\perp$). *Chaos*, as is indicated by our nomenclature, is a relation that permits a chaotic relationship between input and output. It is the relation $\{z_0 \star z_1' \in T^\star \mid z_1' = \perp'\}$ that would be closer to what they have in mind (since they refer to $\perp$ as the undefined value). We, for reasons discussed above, will call this *Abort*.

Now $W_\bullet$-refinement is defined as the subset relation on the Woodcock-completion. Since *Chaos* is the whole of $T^\star$, *every schema refines it.* In particular the identity relation refines it, and this is the identity for composition. Hence one has:[13]

$$\kappa = Identity \mathbin{⨾} \kappa \sqsupseteq Chaos \mathbin{⨾} \kappa$$

for *any* κ, and this would not appear to be recovery from run-time error, but a natural consequence of the general permissiveness inherent in *Chaos*, indeed, a natural consequence of the fact that the Woodcock-completion is non-strict with respect to the abortive value.

[12] Note, this is strictness with respect to chaos, not the abortive value.

[13] At least when κ is *e.g.* total, and so when composition can be guaranteed to be monotonic. We will explore this further in our companion papers.

Our analysis has, on the other hand, provided a very clear mathematical explanation for lifting: with non-lifted totalisation it is not possible to prove proposition 4.11 (which requires explicit use of $\perp$ value). Indeed, we can do better: the following is an explicit counterexample.

Definition 4.24

$$True =_{df} [T \mid true]$$

Proposition 4.25

$$\overset{\diamond}{True} = \overset{\diamond}{Chaos}$$

∎

It is an immediate consequence that the more permissive notion of refinement does not, for example, insist that preconditions do not strengthen.
We have, however, only begun to provide answers to the natural questions that arise. For example, although lifting appears to be necessary, why does it have to be non-strict with respect to $\perp$? Proposition 4.25 also raises a question: why is there a distinction between *implicit* (*Chaos*) and *explicit* (*True*) permission to behave? Note that in the Woodcock-completion, $\overset{\bullet}{True} \neq \overset{\bullet}{Chaos}$.

5 The non-lifted totalisation

In the previous section we noted the asymmetry between implicit and explicit chaos. Implicit chaos is more extensive, it permits abortive behaviour that explicit chaos does not allow. This asymmetry seems inevitable in order to obtain a reasonable theory of refinement.[14] This is, as we have shown, indeed the case, unless we re-examine the nature of preconditions.
It seemed only natural to identify the notion of the precondition of a schema with the domain of definition of the underlying relation. There is, however, an alternative approach. Instead of taking a value to be in the precondition when it *is* related to at least one element of the co-domain of the *underlying* relation, we could take the condition to be that a value *is not* related to at least one element of the co-domain of the *completed* relation. This anticipates the idea that a value which is not in the domain of definition of the underlying relation, will be related to *all* values in the co-domain *after* the relation is completed: it excludes from the precondition values in the underlying relation which are *already* related to all values in the co-domain.
Surprisingly, this leads to a theory which can be formalised entirely in $\mathcal{Z}_{\mathcal{C}}$ (it does not require lifting at all) and which is equivalent to the theories of the previous section. In this way we show that, for operation refinement at least, that *lifting* of relations is not necessary if one wishes to establish a relational completion semantics for refinement.

5.1 Preconditions revisited

In this section we refer to the standard definition of preconditions as Pre_0 in order to contrast it clearly with the new one.

[14] One might take S-refinement to establish *minimal* conditions.

Definition 5.1 Let $T^{in} \preceq V$.

$$Pre_1\ U\ z^V =_{df} \exists\, x_0', x_1' \in T^{out'} \bullet z \upharpoonright T^{in} \star x_0' \notin U \wedge z \upharpoonright T^{in} \star x_1' \in U$$

There is a similarity (but not quite an equivalence) here with the *total model* described in [10] (page 45). His interpretation of specifications as predicates in that model makes use of a similar concept of precondition to Pre_1, although this is not made explicit.

Proposition 5.2 The following introduction and elimination rules are derivable for preconditions:

$$\frac{t \star t_0' \notin U \quad t \star t_1' \in U \quad t_0' \in T^{out'}}{Pre_1\ U\ t}\ (Pre_1^+)$$

$$\frac{Pre_1\ U\ t \quad t \upharpoonright T^{in} \star y_0' \notin U, t \upharpoonright T^{in} \star y_1' \in U, y_0' \in T^{out'} \vdash P}{P}\ (Pre_1^-)$$

where y_0 and y_1 are fresh variables. ■

The new notion of preconditions implies the old one.

Lemma 5.3 The following rule is derivable:

$$\frac{Pre_1\ U\ t}{Pre_0\ U\ t}$$

■

5.2 $W_\diamond$-refinement

The *totalisation* (non-lifted) of a set of bindings can be defined as follows:

Definition 5.4

$$\overset{\diamond}{U} =_{df} \{z_0 \star z_1' \in T \mid Pre_1\ U\ z_0 \Rightarrow z_0 \star z_1' \in U\}$$

Proposition 5.5 The following rules are derivable:

$$\frac{t_0 \star t_1' \in T \quad Pre_1\ U\ t_0 \vdash t_0 \star t_1' \in U}{t_0 \star t_1' \in \overset{\diamond}{U}}\ (\diamond^+)$$

$$\frac{t_0 \star t_1' \in \overset{\diamond}{U} \quad Pre_1\ U\ t_0}{t_0 \star t_1' \in U}\ (\diamond_0^-) \qquad \frac{t_0 \star t_1' \in \overset{\diamond}{U}}{t_0 \star t_1' \in T}\ (\diamond_1^-)$$

■

Notice that the values in this completion range over the natural carrier set of the type T.

Lemma 5.6

$$\frac{}{U \subseteq \overset{\diamond}{U}}\ (i) \qquad \frac{}{\overset{\diamond}{U} \subseteq \overset{\bullet}{U}}\ (ii) \qquad \frac{\neg\, Pre_0\ U\ t_0 \quad t_0 \in T^{in} \quad t_1' \in T^{out'}}{t_0 \star t_1' \in \overset{\diamond}{U}}\ (iii)$$

■

Proof. For (ii), consider the following derivation:

$$\dfrac{\dfrac{\dfrac{t_0 \star t_1' \in \overset{\diamond}{U}}{t_0 \star t_1' \in T}}{t_0 \star t_1' \in T^\star} \qquad \dfrac{\dfrac{\dfrac{}{t_0 \star t_1' \notin U}\,(2) \qquad \dfrac{t_0 \star t_1' \in \overset{\diamond}{U} \qquad \begin{matrix}\delta\\ \vdots\\ Pre_1\ U\ t_0\end{matrix}}{t_0 \star t_1' \in U}}{\dfrac{false}{t_0 \star t_1' \in U}\,(2)}}{t_0 \star t_1' \in \overset{\bullet}{U}}\,(1)$$

where δ is:

$$\dfrac{\dfrac{}{Pre_0\ U\ t_0}\,(1) \qquad \dfrac{\dfrac{}{t_0 \star t_1' \notin U}\,(2) \qquad \dfrac{}{t_0 \star y' \in U}\,(3) \qquad \dfrac{\dfrac{t_0 \star t_1' \in \overset{\diamond}{U}}{t_0 \star t_1' \in T}}{t_1' \in T^{out'}}}{Pre_1\ U\ t_0}}{Pre_1\ U\ t_0}\,(3)$$

∎

W$_\diamond$-refinement is then defined as follows:

Definition 5.7

$$U_0 \sqsupseteq_{w_\diamond} U_1 =_{df} \overset{\diamond}{U_0} \subseteq \overset{\diamond}{U_1}$$

Obvious introduction and elimination rules are derivable.

5.3 W$_\diamond$-refinement and S$_1$-refinement are equivalent

As we have seen, the abortive value was critical in showing that W$_\bullet$-refinement and S-refinement are equivalent. Naturally we should assure ourselves that W$_\diamond$-refinement and a modified version of S-refinement are equivalent.
Let S$_1$-refinement be S-refinement, in which all instances of Pre_0 are replaced by Pre_1. We will content ourselves by showing that W$_\diamond$-refinement satisfies the S$_1$-refinement elimination rule concerning preconditions. The remaining elimination rule, and indeed the other direction of the equivalence proof are not significantly different from the proofs we provided earlier in section 4.2.

Proposition 5.8 The following rule is derivable:

$$\dfrac{U_0 \sqsupseteq_{w_\diamond} U_1 \qquad Pre_1\ U_1\ t}{Pre_1\ U_0\ t}$$

PROOF. Consider the following derivation:

$$\dfrac{\dfrac{\mathit{Pre}_1\ U_1\ t \qquad \dfrac{\dfrac{\dfrac{\overset{\delta}{\vdots}\ \ t \star y_0' \in U_1 \qquad \dfrac{}{t \star y_0' \notin U_1}\,(2)}{\mathit{false}}}{t \star y_1' \notin U_1} \qquad \dfrac{}{t \star y_1' \in U_1}\,(2)}{\mathit{false}}}{\mathit{false}}\,(2)}{\mathit{Pre}_1\ U_0\ t}\,(1)$$

where δ is:

$$\dfrac{\dfrac{U_0 \sqsupseteq_{w_\diamond} U_1 \qquad \dfrac{\dfrac{}{\neg\ \mathit{Pre}_1\ U_0\ t}\,(1) \qquad \dfrac{\mathit{Pre}_1\ U_1\ t}{t \in T^{in}} \qquad \dfrac{}{y_0' \in T^{out'}}\,(2)}{t \star y_0' \in \overset{\diamond}{U_0}}\,(5.6(iii))}{t \star y_0' \in \overset{\diamond}{U_1}} \qquad \mathit{Pre}_1\ U_1\ t}{t \star y_0' \in U_1}$$

∎

5.4 W$_\diamond$-refinement and W$_\bullet$-refinement are equivalent (in $\mathcal{Z_C}$)

We begin by showing that W$_\diamond$-refinement satisfies the W$_\bullet$-refinement elimination rule, for bindings, which range over the natural carrier set.

Proposition 5.9 Let $t_0 \star t_1'$ be a binding with the property that:

$$t_0 \star t_1' \in T$$

Then the following rule is derivable:

$$\dfrac{U_0 \sqsupseteq_{w_\diamond} U_1 \qquad t_0 \star t_1' \in \overset{\bullet}{U_0}}{t_0 \star t_1' \in \overset{\bullet}{U_1}}$$

PROOF. Consider the following derivation:

$$\dfrac{\dfrac{U_0 \sqsupseteq_{w_\diamond} U_1 \qquad \dfrac{t_0 \star t_1' \in T \qquad \dfrac{t_0 \star t_1' \in \overset{\bullet}{U_0} \qquad \dfrac{\dfrac{}{\mathit{Pre}_1\ U_0\ t_0}\,(1)}{\mathit{Pre}_0\ U_0\ t_0}\,(5.3)}{t_0 \star t_1' \in U_0}}{t_0 \star t_1' \in \overset{\diamond}{U_0}}\,(1)}{t_0 \star t_1' \in \overset{\diamond}{U_1}}}{t_0 \star t_1' \in \overset{\bullet}{U_1}}\,(5.6(ii))$$

∎

Theorem 5.10 When W$_\bullet$-refinement is understood to range over the natural carriers, then, we have:

$$\frac{U_0 \sqsupseteq_{w_\diamond} U_1}{U_0 \sqsupseteq_{w_\bullet} U_1}$$

■

Likewise, we show that W$_\bullet$-refinement satisfies the W$_\diamond$-refinement elimination rule.

Proposition 5.11 The following rule is derivable:

$$\frac{U_0 \sqsupseteq_{w_\bullet} U_1 \quad t_0 \star t_1' \in \overset{\diamond}{U_0}}{t_0 \star t_1' \in \overset{\diamond}{U_1}}$$

Proof. Consider the following derivation:

$$\dfrac{\dfrac{t_0 \star t_1' \in \overset{\diamond}{U_0}}{t_0 \star t_1' \in T} \qquad \dfrac{\dfrac{U_0 \sqsupseteq_{w_\bullet} U_1 \quad \dfrac{t_0 \star t_1' \in \overset{\diamond}{U_0}}{t_0 \star t_1' \in \overset{\bullet}{U_0}}\;(5.6(ii))}{t_0 \star t_1' \in \overset{\bullet}{U_1}} \qquad \dfrac{\dfrac{}{Pre_1\ U_1\ t_0}\;(1)}{Pre_0\ U_1\ t_0}\;(5.3)}{t_0 \star t_1' \in U_1}\;(1)}{t_0 \star t_1' \in \overset{\diamond}{U_1}}$$

■

In this case, a term cannot satisfy the premise $t_0 \star t_1' \in \overset{\diamond}{U_0}$ without belonging to the natural carrier, so in this direction no extra condition is necessary.

Theorem 5.12

$$\frac{U_0 \sqsupseteq_{w_\bullet} U_1}{U_0 \sqsupseteq_{w_\diamond} U_1}$$

■

For terms ranging over the natural carriers, then, the non-lifted and lifted models are equivalent. Since W$_\diamond$-refinement can be formalised in $\mathcal{Z}_\mathcal{C}$, and since the carrier sets of $\mathcal{Z}_\mathcal{C}$ are the natural carriers of $\mathcal{Z}_\mathcal{C}^\perp$, we may conclude that we can formalise a form of W-refinement in $\mathcal{Z}_\mathcal{C}$ without moving to the extended theory and introducing lifting. The penalty for this is the need for a novel notion of precondition.

6 The strict-lifted totalisation

A second question arising from our review was this: why is it necessary to permit recovery from the abortive value in completing the relation? In order to investigate this we consider a final relational completion in which the relation is lifted and totalised, but is strict with respect to abortive behaviour: $\perp$ maps only to $\perp$.

Definition 6.1

$$\overset{\ominus}{U} =_{df} \{z_0 \star z_1' \in T^\star \mid \mathit{Pre}\ U\ z_0 \Rightarrow z_0 \star z_1' \in U \wedge z_0 = \perp \Rightarrow z_1' = \perp'\}$$

We obtain obvious introduction and elimination rules, which in this case we will not state explicitly. In addition we have, what are by now, fairly standard properties:

Lemma 6.2

$$\frac{}{U \subseteq \overset{\ominus}{U}}\ (i) \qquad \frac{}{\overset{\ominus}{U} \subseteq \overset{\bullet}{U}}\ (ii) \qquad \frac{}{\perp \in \overset{\ominus}{U}}\ (iii)$$

$$\frac{\neg\ \mathit{Pre}\ U\ t \quad t \in T_\perp^{in}}{t \star \perp' \in \overset{\ominus}{U}}\ (iv) \qquad \frac{\neg \mathit{Pre}\ U\ t_0 \quad t_0 \in T^{in} \quad t_1' \in T_\perp^{out'}}{t_0 \star t_1' \in \overset{\ominus}{U}}\ (v)$$

Notice that in (v) t_0 ranges over the natural carrier set, rather than the extended carrier. ■

We now introduce W$_\ominus$-refinement.

Definition 6.3

$$U_0 \sqsupseteq_{w_\ominus} U_1 =_{df} \overset{\ominus}{U_0} \subseteq \overset{\ominus}{U_1}$$

Again, we will not state the obvious rules.

7 W$_\ominus$-refinement and W$_\bullet$-refinement are equivalent

In the usual manner, we show that W$_\ominus$-refinement satisfies the elimination rule of W$_\bullet$-refinement.

Proposition 7.1 The following rule is derivable:

$$\frac{U_0 \sqsupseteq_{w_\ominus} U_1 \quad t_0 \star t_1' \in \overset{\bullet}{U_0}}{t_0 \star t_1' \in \overset{\bullet}{U_1}}$$

Proof. Consider the following derivation:

$$\dfrac{\dfrac{t_0 \star t_1' \in \overset{\bullet}{U_0}}{t_0 \star t_1' \in T^\star} \qquad \dfrac{\dfrac{\dfrac{\begin{array}{c}\delta_0 \\ \vdots \\ U_0 \sqsupseteq_{w_\ominus} U_1 \quad t_0 \star t_1' \in \overset{\ominus}{U_0}\end{array}}{t_0 \star t_1' \in \overset{\ominus}{U_1}} \qquad \dfrac{}{\mathit{Pre}\ U_1\ t_0}\,(1)}{t_0 \star t_1' \in U_1}}{}}{t_0 \star t_1' \in \overset{\bullet}{U_1}}\,(1)$$

where δ_0 is:

$$\dfrac{t_0 \star t_1' \in \overset{\bullet}{U_0} \qquad \dfrac{\dfrac{\dfrac{\dfrac{\begin{matrix}\delta_1\\ \vdots\\ U_0 \sqsupseteq_{w_\ominus} U_1 \quad t_0\star \bot' \in \overset{\ominus}{U_0}\end{matrix}}{t_0\star \bot' \in \overset{\ominus}{U_1}} \qquad \dfrac{}{\mathit{Pre}\ U_1\ t_0}\,(1)}{t_0\star \bot' \in U_1}}{\mathit{false}}\,(2.2)}{t_0 \star t_1' \in \overset{\ominus}{U_0}} \qquad \dfrac{\dfrac{}{t_0 \star t_1' \in U_0}\,(2)}{t_0 \star t_1' \in \overset{\ominus}{U_0}}\,(6.2(i))}{t_0 \star t_1' \in \overset{\ominus}{U_0}}\,(2)\,(3.8)$$

and where δ_1 is:

$$\dfrac{\dfrac{}{\neg\ \mathit{Pre}\ U_0\ t_0}\,(2) \qquad \dfrac{\dfrac{t_0 \star t_1' \in \overset{\bullet}{U_0}}{t_0 \star t_1' \in T^{\star}}}{t_0 \in T_{\bot}^{in}}}{t_0\star \bot' \in \overset{\ominus}{U_0}}\,(6.2(iv))$$

Notice the use of the second version of rlue $(\bullet_{o}^{-})$ (proposition 3.8) in δ_0.

∎

Theorem 7.2

$$\dfrac{U_0 \sqsupseteq_{w_\ominus} U_1}{U_0 \sqsupseteq_{w_\bullet} U_1}$$

∎

Likewise, we prove that $W_\bullet$-refinement implies $W_\ominus$-refinement by proving that $W_\bullet$-refinement satisfies the elimination rule of $W_\ominus$-refinement.

Proposition 7.3 The following rule is derivable:

$$\dfrac{U_0 \sqsupseteq_{w_\bullet} U_1 \quad t_0 \star t_1' \in \overset{\ominus}{U_0}}{t_0 \star t_1' \in \overset{\ominus}{U_1}}$$

Proof. Consider the following derivation:

$$\dfrac{\dfrac{t_0 \star t_1' \in \overset{\ominus}{U_0}}{t_0 \star t_1' \in T^{\star}} \qquad \dfrac{\begin{matrix}\delta\\ \vdots\\ t_0 \star t_1' \in \overset{\bullet}{U_1}\end{matrix} \quad \dfrac{}{\mathit{Pre}\ U_1\ t_0}\,(1)}{t_0 \star t_1' \in U_1} \qquad \dfrac{t_0 \star t_1' \in \overset{\ominus}{U_0} \quad \dfrac{}{t_0 = \bot}\,(1)}{t_1' = \bot'}}{t_0 \star t_1' \in \overset{\ominus}{U_1}}\,(1)$$

where δ is:

$$\dfrac{U_0 \sqsupseteq_{w_\bullet} U_1 \qquad \dfrac{\dfrac{t_0 \star t_1' \in \overset{\ominus}{U}_0 \qquad \dfrac{U_0 \sqsupseteq_{w_\bullet} U_1 \quad \dfrac{}{\mathit{Pre}\ U_1\ t_0}\,(1)}{\mathit{Pre}\ U_0\ t_0}\,(4{\cdot}11)}{t_0 \star t_1' \in U_0}}{t_0 \star t_1' \in \overset{\bullet}{U}_0}\,(3{\cdot}9(i))}{t_0 \star t_1' \in \overset{\bullet}{U}_1}$$

■

Theorem 7.4

$$\dfrac{U_0 \sqsupseteq_{w_\bullet} U_1}{U_0 \sqsupseteq_{w_\ominus} U_1}$$

■

This concludes the analysis.

8 Conclusions and future work

Refinement in the context of *partial* relational models of specifications is an important sub-area of refinement in general, not least because the standard interpretation of Z specifications is a partial relational interpretation. There are, as we have demonstrated here, a number of, at least at first sight, distinct notions of refinement in this context. For Z, the standard account is that due to Woodcock and is based on an interesting notion of relational completion which lifts and totalises partial relations. We have shown that this notion is by no means arbitrary, but that there are a number of alternative notions which are equivalent. We have been able to *begin* to explain some of the mathematical reasons why it is defined in precisely the way it is, particularly why it is necessary to introduce the abortive value $\perp$. Contrary to the argument in [18] the critical reason concerns the behaviour of preconditions, rather than the behaviour of refinement under composition. On the other hand a novel notion of what it means for a value to be in the precondition of a specification leads to a Woodcock-like theory in which the relations need not be lifted. There is, in addition, the possibility of insisting that the abortive value is handled in a strict fashion, also leading to an equivalent theory.

It would have been possible to develop the technical material in this paper purely in terms of abstract partial relations, rather than those induced by Z specifications. Formalisation in $\mathcal{Z}_{\mathcal{C}}$, however, makes dealing with Z no more complicated than dealing with abstract partial relations, and it has a further advantage which will become clear in the companion papers to follow: we may smoothly extend our analysis to explore refinement in the context of the *schema calculus*. The specification language Z has the advantage of permitting large scale specifications to be constructed in a modular and systematic fashion using connectives reminiscent of quantificational logic. The development of Z logic in, for example, [11] shows how this algebra of schema expressions can be realised as a calculus by showing how to interpret the connectives and quantifiers as operations on partial relations which have, in general, non-equal domains. In order to understand how Z, in particular, handles refinement in the generalised context of the schema calculus, a separate investigation is planned as a sequel to this paper. In particular, monotonicity issues have to be fully explored. It is well-known that the specification forming operations have poor monotonicity properties with respect to refinement in the relational completion semantics, but a thorough investigation, with a careful consideration of all the options, has not to date been undertaken. What we can say already, given the results in this paper, is that there is no likelihood for improved monotonicity properties for Z if its underlying semantics is to remain based on the partial relation model: every refinement theory we have constructed on this basis is equivalent. It is perhaps worth pointing out that although we only gave an explicit semantics for atomic schemas (see definition 2.1) the meta-variables U range over arbitrary sets of bindings of type $T^{in} \curlyvee T^{out'}$. In other words we can plug in the entire semantics of the schema calculus (in the way described in, for example, [11]) beneath the account of operation refinement without affecting the analysis in any way.[15] The consequence of this is, then, that all six theories we have described must suffer from the same defects regarding monotonicity properties. Improving upon this position will be one of the themes underlying our further investigation and we will see how the one way forward is to abandon the standard semantics and its equational logic, in favour of an inequational logic of refinement.

There is a well-known alternative to the notion of refinement which we have investigated in this paper. This is the *behavioural* approach outlined, for example, in [6]. In this model, preconditions are understood to be firing conditions (or triggers) and may, therefore, not be weakened. A parallel investigation to the one we have described in this paper (with extensions to cover the schema calculus) is possible, and will also be reported in a future paper in this series.

We made comment, in section 3.2, regarding the relationship between S-refinement and a theorem concerning two-predicate designs appearing in [12]. The semantics of these designs involve a *termination* value (it is written *ok'*). This suggests an interesting generalisation of the work reported here in which two-predicate designs are represented by total relations involving both signals for termination and non-termination. This model will also be reported in future work.

This paper has concentrated on operation refinement, that is the degenerate case of data refinement in which all simulations are identity functions. This was sufficient for our purposes here, but the deeper notion is data refinement, and there are further aspects to the questions we have raised regarding the Woodcock-completion which do not emerge until data refinement is considered. This is why we emphasised the word "begin" earlier in this concluding section. Operation refinement is sufficiently interesting mathematically to motivate the range of possibilities we have explored here, but there are divergences and surprises to uncover when non-trivial simulations are permitted. This topic will also be covered in a companion paper, making links and comparison between our proof-theoretic approach and the largely semantic approach generally taken in [5].

Our analysis of weakest precondition semantics is here restricted to the atomic schema

[15] However, one can reasonably argue that, by insulating the schema logic (based on the partial relation interpretation) from the refinement logic (based on any one of the six theories we have described) Z effectively reduces its refinement logic to the atomic schema case. This is because the schema logic permits the derivation of an equational logic which makes all schema expressions equal to an atomic schema.

case, and to operation refinement. Some interesting work, developing one or other of these dimensions, has been undertaken, for example by Woodcock and Cavalcanti [2] [4], by Groves [9], and in [5]. Formalisation and investigation within our theoretical setting, of an extension to what we have shown in this paper and with our proof-theoretic methodology, is both simple and revealing. It will form an additional phase of our exploration of refinement and we plan to cover the topic in another paper in this series.

9 Acknowledgements

We would like to thank the EPSRC (grant reference: GR/L57913), the Royal Society of Great Britain and the New Zealand Foundation for Research, Science and Technology (grant reference: UOWX0011) for financially supporting this research. Moshe Deutsch is supported by the British Council through an ORS award.
This work has been influenced in its development by too many people to name explicitly. However, special thanks for particularly important discussions and comments go to Rob Arthan, Lindsay Groves, Greg Reeve, Ray Turner and Jim Woodcock.

References

[1] S. Brien and J. Nicholls. Z base standard version 1.0. Technical report, University of Oxford, 1992.

[2] A. Cavalcanti. *A Refinement Calculus for Z.* PhD thesis, University of Oxford, 1997.

[3] A. Cavalcanti and J. C. P. Woodcock. A weakest precondition semantics for Z. *The Computer Journal*, 41(1):1–15, 1998.

[4] A. Cavalcanti and J. C. P. Woodcock. ZRC – a refinement calculus for Z. *Formal Aspects of Computing*, 10(3):267–289, 1998.

[5] W. P. de Roever and K. Engelhardt. *Data refinement: model-oriented proof methods and their comparison.* Prentice Hall International, 1998.

[6] J. Derrick and E. Boiten. *Refinement in Z and Object-Z: Foundations and Advanced Applications.* Formal Approaches to Computing and Information Technology – FACIT. Springer, May 2001.

[7] A. Diller. *Z: An introduction to formal methods.* J. Wiley and Sons, 2^{nd} edition, 1994.

[8] I. Toyn (ed.). *Z Notation: Final committee draft, CD 13568.2.* Z Standards Panel, 1999. ftp://ftp.york.ac.uk/hise_reports/cadiz/ZSTAN/fcd.ps.gz.

[9] L. Groves. *Evolutionary software development in the refinement calculus.* PhD thesis, Victoria University, 2000.

[10] J. Grundy. *A method of program refinement.* PhD thesis, University of Cambridge, 1993.

[11] M. C. Henson and S. Reeves. Investigating Z. *Logic and Computation*, 10(1):43–73, 2000.

[12] C.A.R Hoare and J. He. *Unifying Theories of Programming.* Prentice Hall International, 1998.

[13] S. King. Z and the Refinement Calculus. In D. Bjørner, C. A. R. Hoare, and H. Langmaack, editors, *VDM '90 VDM and Z – Formal Methods in Software Development*, volume 428 of *Lecture Notes in Computer Science*, pages 164–188. Springer-Verlag, April 1990.

[14] C. C. Morgan. *Programming from Specifications.* Prentice Hall International, 2^{nd} edition, 1994.

[15] B. Potter, J. Sinclair, and D. Till. *An introduction to formal specification and Z.* Prentice Hall, 2^{nd} edition, 1996.

[16] J. M. Spivey. *The Z notation: A reference manual.* Prentice Hall, 2^{nd} edition, 1992.

[17] J. C. P. Woodcock and S. Brien. $\mathcal{W}$: A logic for Z. In *Proceedings of ZUM '91, 6th Conf. on Z.* Springer Verlag, 1992.

[18] J. C. P. Woodcock and J. Davies. *Using Z: Specification, Refinement and Proof.* Prentice Hall, 1996.

A The specification logic $\mathcal{Z_C}$ and the theory $\mathcal{Z}_\mathcal{C}^\perp$

A.1 $\mathcal{Z_C}$

A.1.1 The language of $\mathcal{Z_C}$

$\mathcal{Z_C}$ is an extension of higher order logic with *schema types.* We begin with the types:

$$T ::= \mathbb{B} \;\Big|\; \mathbb{N} \Big|\; \mathbb{P}\,T \;\Big|\; T \times T \;\Big|\; [\cdots \mathsf{z}^T \cdots]$$

We will often permit the meta-variable D to range over sequences of labels such as $\cdots \mathsf{z}_i^{T_i} \cdots$ (the order is not important), also writing $\alpha[D]$, when D is $\cdots \mathsf{z}_i^{T_i} \cdots$, for the alphabet set (in the meta-language) of labels $\{\cdots \mathsf{z}_i \cdots\}$. No label may occur more than once in such a type.
Types of the form $[D]$ are called *schema types.* The symbols $\preceq$, $\curlywedge$, $\curlyvee$ and $-$ denote the *schema subtype* relation, and the operations of *schema type intersection*, (compatible) *schema type union* and *schema type subtraction.*
All categories of the language of $\mathcal{Z_C}$ must be well-formed with respect to these types.
Next we have the category of *terms*: We assume the existence of a denumerable set of variables for each type T. We use t as a meta-variable over terms of arbitrary type.
The syntax of terms is then:

$$\begin{array}{lll}
t^T & ::= & x^T \;\Big|\; t^{[\cdots \mathsf{z}^T \cdots]}.l \;\Big|\; t^{T \times T_1}.1 \;\Big|\; t^{T_0 \times T}.2 \\
t^{[D]} & ::= & t^{T_2} \upharpoonright [D] \\
t^{[\cdots \mathsf{z}^T \cdots]} & ::= & \langle\!| \cdots \; \mathsf{z} \Rrightarrow t^T \; \cdots |\!\rangle \\
t^{T_0 \times T_1} & ::= & (t^{T_0}, t^{T_1}) \\
t^{\mathbb{P}\,T} & ::= & \{z^T \mid P\} \\
t^{\mathbb{B}} & ::= & \mathsf{True} \;\mid\; \mathsf{False} \\
t^{\mathbb{N}} & ::= & 0 \;\mid\; 1 \;\mid\; 2 \;\mid\; \cdots
\end{array}$$

where $[D] \preceq T_2$.
For clarity of presentation we permit the meta-variable C (*etc.*) to range over terms of power type, and S (*etc.*) to range over sets of schema type. The latter are *schemas.* We let U (*etc.*) range over *operation* schemas. These are schemas involving both *before* and *after* observations. We can always write the type of such operation schemas as $\mathbb{P}(T^{in} \curlyvee T^{out'})$ where T^{in} is the type of the input sub-binding and $T^{out'}$ is the type of the output sub-binding. We also permit *binding concatenation*, written $t_0 \star t_1$ when the alphabets of t_0 and t_1 are disjoint. This is, in fact, exclusively used for partitioning bindings in operation schemas into before and after components, so the terms involved are necessarily disjoint. We lift this operation to sets (of appropriate type):

$$C_0 \star C_1 =_{df} \{z_0 \star z_1 \mid z_0 \in C_0 \wedge z_1 \in C_1\}$$

The same restriction obviously applies here: the types of the sets involved must be disjoint.
We employ the notation $b.P$ and $b.t$ (generalising binding selection) which is adapted from [17]. Suppose that $\{\mathsf{z}_0 \cdots \mathsf{z}_n\}$ is the alphabet set of t, then $t.P$ is $P[\mathsf{z}_0/t.\mathsf{z}_0] \cdots [\mathsf{z}_n/t.\mathsf{z}_n]$.
The formulæ of $\mathcal{Z_C}$ delineate a typed bounded predicate logic.

$$P \;::=\; \mathit{false} \;\mid\; t^T = t^T \;\mid\; t^T \in C^{\mathbb{P}\,T} \;\mid\; \neg P \;\mid\; P \vee P \;\mid\; \exists\, z^T \in C^{\mathbb{P}\,T} \bullet P$$

The logic of $\mathcal{Z_C}$ is classical, so the remaining logical operations are available by definition. We also, as usual, abbreviate $\neg\,(t^T \in C^{\mathbb{P}\,T})$ to $t^T \notin C^{\mathbb{P}\,T}$.
A crucial observation is *unicity of types*: every term and set of $\mathcal{Z_C}$ has a unique type. We can make great use of this observation. It enables us to remove type decoration in most circumstances.

A.1.2 The logic of $\mathcal{Z_C}$

The judgements of the logic have the form $\Gamma \vdash P$ where Γ is a set of formulæ.

The logic is presented as a natural deduction system *in sequent form*. We shall omit all data (entailment symbol, contexts, type *etc.*) which remain unchanged by a rule. In the rule $(\exists^-)$, the variable y may not occur in C, P_0, P_1 nor any other assumption.

$$\frac{P_0}{P_0 \vee P_1}\ (\vee_0^+) \qquad \frac{P_1}{P_0 \vee P_1}\ (\vee_1^+) \qquad \frac{P_0 \vee P_1 \quad P_0 \vdash P_2 \quad P_1 \vdash P_2}{P_2}\ (\vee^-)$$

$$\frac{P \vdash \mathit{false}}{\neg P}\ (\neg^+) \qquad \frac{P \quad \neg P}{\mathit{false}}\ (\mathit{false}^+) \qquad \frac{\neg\neg P}{P}\ (\neg^-) \qquad \frac{\mathit{false}}{P}\ (\mathit{false}^-)$$

$$\frac{P[z/t] \quad t \in C}{\exists\, z \in C \bullet P}\ (\exists^+) \qquad \frac{\exists\, z \in C \bullet P_0 \quad y \in C, P_0[z/y] \vdash P_1}{P_1}\ (\exists^-)$$

$$\frac{}{\Gamma, P \vdash P}\ (\mathit{ass}) \qquad \frac{}{t = t}\ (\mathit{ref}) \qquad \frac{t = t' \quad P[z/t]}{P[z/t']}\ (\mathit{sub})$$

$$\frac{}{\langle\!|\, \cdots \mathbf{z}_i \Rrightarrow t_i \cdots |\!\rangle . \mathbf{z}_i = t_i}\ (\Rrightarrow_0^=) \qquad \frac{}{\langle\!|\, \cdots \mathbf{z}_i \Rrightarrow t.\mathbf{z}_i \cdots |\!\rangle = t^{[\cdots \mathbf{z}_i \in T_i \cdots]}}\ (\Rrightarrow_1^=)$$

$$\frac{}{(t_0, t_1).1 = t_0}\ (()_0^=) \qquad \frac{}{(t_0, t_1).2 = t_1}\ (()_1^=) \qquad \frac{}{(t.1, t.2) = t}\ (()_2^=)$$

$$\frac{P[z/t]}{t \in \{z \mid P\}}\ (\{\}^+) \qquad \frac{t \in \{z \mid P\}}{P[z/t]}\ (\{\}^-)$$

$$\frac{t_0 \equiv t_1}{t_0 = t_1}\ (\mathit{ext}) \qquad \frac{t^T.\mathbf{z}_i = t_i \quad [\cdots \mathbf{z}_i : T_i \cdots] \preceq T}{(t \upharpoonright [\cdots \mathbf{z}_i \in T_i \cdots]).\mathbf{z}_i = t_i}\ (\upharpoonright^=)$$

where

$$t_0 \equiv t_1 =_{df} \forall\, z \in t_0 \bullet z \in t_1 \wedge \forall\, z \in t_1 \bullet z \in t_0$$

The transitivity of equality and numerous *equality congruence* rules for the various term forming operations are all derivable in view of rule (*sub*). In particular, we can prove that set-equality in $\mathcal{Z_C}$ is extensional.
The following weakening rule is admissible and is incorporated within the system.

$$\frac{\Gamma \vdash P_1}{\Gamma, P_0 \vdash P_1}\ (\mathit{wk})$$

The carrier sets (with the expected derived rules) are easily defined this system:

$$T = \{z^T \mid \mathit{True}\}$$

The ambiguity presents no problem, since only types appear as superscripts and only sets appear after the membership relation.

A.1.3 Restricted equality

In many contexts we need to compare bindings over a common restricted type.

Definition A.1 Let $T \preceq T_0$ and $T \preceq T_1$.

$$t_0^{T_0} =_T t_1^{T_1} =_{df} t_0 \upharpoonright T = t_1 \upharpoonright T$$

A restricted form of membership is very useful for establishing the schema calculus. In this case the types are uniquely determined by the syntax and do not appear in the notation.

Definition A.2 Let $T_0 \preceq T_1$.

$$t^{T_1}\ \dot{\in}\ C^{\mathbb{P}\, T_0} =_{df} t \upharpoonright T_0 \in C$$

A.1.4 Functions

We will need total functions over types. These are easily introduced.

Definition A.3

$$T_0 \to T_1 =_{df} \{g \in \mathbb{P}(T_0 \star T_1) \mid \mathit{unicity}(g) \wedge \mathit{total}(g)\}$$

Note that functions are modelled as subsets of $T_0 \star T_1$ rather than $T_0 \times T_1$, and that for notational convenience, we let g (*etc.*) range over terms of type $\mathbb{P}(T_0 \star T_1)$. In fact we only do this when such a term is a function.
When g is known to be an element of $T_0 \to T_1$ and $z_0 \in T_0$, we will write $g\ z_0$ (as usual) for the unique element $z_1 \in T_1$ such that $z_0 \star z_1 \in g$.

A.2 $\mathcal{Z}_\mathcal{C}^\perp$

In $\mathcal{Z}_\mathcal{C}^\perp$ we introduce new constants (to play the rôle of "undefined" values) at every type. Thus we postulate new constants $\perp^T$ in every type T. There are, additionally, a number of axioms which ensure that all the new $\perp^T$ values interact properly.

$$\frac{}{\langle\!| \, \mathbf{z}_0 \Rrightarrow \perp^{T_0} \cdots \mathbf{z}_n \Rrightarrow \perp^{T_n} |\!\rangle = \perp^{[\mathbf{z}_0 : T_0 \cdots \mathbf{z}_n : T_n]}}$$

$$\frac{}{(\perp^{T_0}, \perp^{T_1}) = \perp^{T_0 \times T_1}}$$

$$\frac{}{\{z^T \mid z = \perp^T\} = \perp^{\mathbb{P}\, T}}$$

For example:

$$\perp^{[\mathbf{z}_0 : T_0 \cdots \mathbf{z}_n : T_n]} .\mathbf{z}_i = \perp^{T_i} \ (0 \le i \le n)$$

Note that these are the *only* axioms concerning undefined terms, hence, the term forming constructions are *non-strict* with respect to the $\perp^T$ values.
In order to facilitate the proper interpretation of schemas we will need to introduce the *natural carrier sets* which are those sets of elements of types which *explicitly exclude* the $\perp^T$ values:

Definition A.4 The *natural carriers* for each type are defined by closing:

$$\mathbb{N} =_{df} \{z^{\mathbb{N}} \mid z \neq \perp^{\mathbb{N}}\}$$

and:

$$\mathbb{B} =_{df} \{z^{\mathbb{B}} \mid z \neq \perp^{\mathbb{B}}\}$$

under the operations of cartesian product, powerset and schema set.

In $\mathcal{Z_C}$ the schema calculus is introduced by means of definitions such as:

$$[S \mid P] =_{df} \{z^T \mid z \in S \wedge z.P\}$$

for atomic schemas, and:

$$S_0^{\mathbb{P}\, T_0} \vee S_1^{\mathbb{P}\, T_1} =_{df} \{z^{T_0 \curlyvee T_1} \mid z \mathrel{\dot\in} S_0 \vee z \mathrel{\dot\in} S_1\}$$

for schema disjunction.
In interpreting the schema calculus in $\mathcal{Z}_\mathcal{C}^\perp$ we systematically modify the defintions so that the values are drawn over the natural carriers rather than the type. For example:

$$[S \mid P] =_{df} \{z \in T \mid z \in S \wedge z.P\}$$

defines atomic schemas, and:

$$S_0^{\mathbb{P}\, T_0} \vee S_1^{\mathbb{P}\, T_1} =_{df} \{z \in (T_0 \curlyvee T_1) \mid z \mathrel{\dot\in} S_0 \vee z \mathrel{\dot\in} S_1\}$$

defines schema disjunction.
As a result the schema calculus is *hereditarily* $\perp$*-free*, since obviously:

Proposition A.5

$$t \in U^T \Rightarrow t \in T$$

∎

We will also need the *extended carriers*. These are defined for all types as follows:

Definition A.6

$$T_\perp =_{df} T \cup \{\perp^T\}$$

A.3 Conservativity

$\mathcal{Z}_\mathcal{C}^\perp$ is conservative over $\mathcal{Z}_\mathcal{C}$. The presence of *projection terms* ($t.1, t.2, t.\mathbf{z}$) in $\mathcal{Z}_\mathcal{C}^\perp$ makes the definition of a simple translation of $\mathcal{Z}_\mathcal{C}^\perp$ into $\mathcal{Z}_\mathcal{C}$ difficult. The appropriate approach is to show that there is a translation of $\mathcal{Z}_\mathcal{C}^{\perp -}$ into $\mathcal{Z}_\mathcal{C}$, where $\mathcal{Z}_\mathcal{C}^{\perp -}$ is roughly $\mathcal{Z}_\mathcal{C}^\perp$ with the projections removed. Having shown that $\mathcal{Z}_\mathcal{C}^{\perp -}$ is a conservative extension of $\mathcal{Z}_\mathcal{C}$ we can then show that $\mathcal{Z}_\mathcal{C}^\perp$ is a conservative (in fact a definitional) extension of $\mathcal{Z}_\mathcal{C}^{\perp -}$. This then completes the task.

A.3.1 The theory $\mathcal{Z}_\mathcal{C}^{\perp -}$

The language of $\mathcal{Z}_\mathcal{C}^{\perp -}$ is that of $\mathcal{Z}_\mathcal{C}^\perp$ with the projection terms removed. The $\mathcal{Z}_\mathcal{C}^{\perp -}$ logic is the logic of $\mathcal{Z}_\mathcal{C}^\perp$ with the equality rules for the projection terms removed and the following equality rules (which are derivable in $\mathcal{Z}_\mathcal{C}^\perp$ and $\mathcal{Z}_\mathcal{C}$) added.

$$\frac{t_0 = t_2 \quad t_1 = t_3}{(t_0, t_1) = (t_2, t_3)} \qquad \frac{\cdots\ t_i = v_i \cdots}{\langle\!| \cdots\ \mathbf{z}_i \Rrightarrow t_i \cdots |\!\rangle = \langle\!| \cdots\ \mathbf{z}_i \Rrightarrow v_i \cdots |\!\rangle}$$

A.3.2 $\mathcal{Z}_\mathcal{C}^{\perp -}$ is conservative over $\mathcal{Z}_\mathcal{C}$

Definition A.7 We define a mapping from the language of $\mathcal{Z}_\mathcal{C}^{\perp -}$ into that of $\mathcal{Z}_\mathcal{C}$. We omit the type superscripts which can easily be inferred.

$(\exists z \bullet P)^*$	$=$	$\exists z \bullet P^*$	
$(P_0 \vee P_1)^*$	$=$	$P_0^* \vee P_1^*$	
$(\neg P)^*$	$=$	$\neg P^*$	
false^*	$=$	false	
$(t \in \{z \mid P\})^*$	$=$	$P[z/t]^*$	
$((t_0, t_1) = (t_2, t_3))^*$	$=$	$(t_0 = t_2)^* \wedge (t_1 = t_3)^*$	
$(\langle\!| \cdots\ \mathbf{z}_i \Rrightarrow t_i\ \cdots |\!\rangle = \langle\!| \cdots\ \mathbf{z}_i \Rrightarrow v_i\ \cdots |\!\rangle)^*$	$=$	$\cdots \wedge\ (t_i = v_i)^* \wedge\ \cdots$	
$(\{z \mid P_0\} = \{z \mid P_1\})^*$	$=$	$(\forall z \bullet P_0 \Leftrightarrow P_1)^*$	
$(c = c)^*$	$=$	true	(c a constant)
$(\perp = t)^*$	$=$	$(t = \perp)^*$	($t \not\equiv \perp$)
$((t_0, t_1)^* = \perp)^*$	$=$	$(t_0 = \perp)^* \wedge (t_1 = \perp)^*$	
$(\langle\!| \cdots \mathbf{z}_i \Rrightarrow t_i \cdots |\!\rangle = \perp)^*$	$=$	$\cdots \wedge (t_i = \perp)^* \wedge\ \cdots$	
$(\{z \mid P\} = \perp)^*$	$=$	$(\forall z \bullet P \Rightarrow z = \perp)^*$	
$(c = \perp)^*$	$=$	false	(c a constant; $c \not\equiv \perp$)

Note that if P is in the language of $\mathcal{Z}_\mathcal{C}$ then $P^* = P$.

Proposition A.8 If $\Gamma \vdash_{\mathcal{Z}_\mathcal{C}^{\perp -}} P$ then $\Gamma^* \vdash_{\mathcal{Z}_\mathcal{C}} P^*$

PROOF. By induction on the structure of derivations. Since the translation is a direct structural recursion for the most part, the majority of the cases are immediate. We illustrate with one of the new axioms. Consider:

$$\frac{}{(\perp^{T_0}, \perp^{T_1}) = \perp^{T_0 \times T_1}}$$

Note simply that $((\perp^{T_0}, \perp^{T_1}) = \perp^{T_0 \times T_1})^* = (\perp^{T_0} = \perp^{T_0})^* \wedge (\perp^{T_1} = \perp^{T_1})^* = \mathit{true} \wedge \mathit{true} = \mathit{true}$. ∎

Corollary A.9 $\mathcal{Z}_\mathcal{C}^{\perp -}$ is conservative over $\mathcal{Z}_\mathcal{C}$. ∎

A.3.3 $\mathcal{Z}_{\mathcal{C}}^{\perp}$ is conservative over $\mathcal{Z}_{\mathcal{C}}^{\perp\,-}$

In fact $\mathcal{Z}_{\mathcal{C}}^{\perp}$ is a definitional extension of $\mathcal{Z}_{\mathcal{C}}^{\perp\,-}$.

Definition A.10 For all types, T_0 and T_1 we define:

$$\mathbf{1}^{\mathbb{P}((T_0 \times T_1) \times T_0)} =_{df} \{((z_0, z_1), z_2)^{(T_0 \times T_1) \times T_0} \mid z_0 = z_2\}$$

Note, that $\mathbf{1}$ satisfies totality and unicity so $\mathbf{1} \in (T_0 \times T_1) \to T_0$

Then we can define the first projection operator of $\mathcal{Z}_{\mathcal{C}}^{\perp}$(and of course, of $\mathcal{Z}_{\mathcal{C}}$).

Definition A.11

$$t^{T_0 \times T_1}.1 = \mathbf{1}^{\mathbb{P}((T_0 \times T_1) \times T_0)}(t)$$

Similar definitions can be established for the second projection and for binding projections. With these in place one easily derives the missing $\mathcal{Z}_{\mathcal{C}}^{\perp}$ equality rules for projections.

Proposition A.12 $\mathcal{Z}_{\mathcal{C}}^{\perp}$ is conservative over $\mathcal{Z}_{\mathcal{C}}^{\perp\,-}$ ∎

Received 18th January 2002, revised 10 February 2003.