A Treatment of Partiality: its Application to the B Method

Lilian Burdy

Matra Transport International
48-56 rue Barbès 92120 Montrouge
tel : +33 (0) 1 49 65 76 37 fax : +33 (0) 1 49 65 70 55
e-mail : burdy@matra-transport.fr

Lab. CEDRIC
Conservatoire National des Arts et Métiers
292 rue St Martin 75141 Paris Cedex 3
e-mail : burdy@cnam.fr

1 Introduction

This paper presents the solution that has been developed at Matra Transport International to treat the ill-definedness problem occurring in the B method. The B method is a formal method that permits to develop software from the specification to the code generation by successive refinement steps. The validity of each refinement step is ensured by the proof of automatically produced lemmas.
We have constructed a logic that guarantees that the proof of the refinement lemmas is sufficient to ensure the validity of all the sources (from the specification to the code) of a B development. The main problem comes from the fact that the B language contains partial functions. This implies that one could construct a model with an undefined value (the term 1/0 could be interpreted as undefined), thus our interpretation is three-valued (a formula is either true, false or ill-defined). Different three-v alued interpretations and different consequence relations exist for three-valued logics. Thus, we have to make some choices.
After these steps, we have a logic for the B method but the deduction system sound and complete for the consequence relation that we have chosen, does not satisfy us. We consider that it would be ineffective if it were implemented on an automatic prover. For the B method to be used in an industrial context, it is necessary to develop efficient tools. Thus, we look for a different way to prove our lemmas.
The main contribution is the theorem 2. This theorem indicates that deduction systems that are not sound for our consequence relation can be used to prove our lemmas, if we associate to each proof another well-definedness proof. Since this result is true for many deduction systems, we have to choose, once again, one of these deduction systems.
After selecting the most effective deduction system, we conclude by presenting

the application of these theoretical results to the B method. We also show the impact of these new lemmas of well definedness on the development of an industrial project (an automatic pilot for subway) using the "well-defined B method".

2 Semantics

The B language can be defined as a language that extends the first order logic (FOL) language where all functions and predicates are total. In B, terms, as well as formulae, can be either well defined or ill-defined.
A term is well defined if the functions that it contains are called in their domain, otherwise it is ill-defined. A formula that contains ill-defined terms or predicates (ill-defined predicates are predicates with some parameters that do not belong to the domain of the predicate) can be ill-defined or not.
We define a three-valued semantics for our language. We extend the classical interpretation of terms by an interpretation $\mathcal{J}_v$ where the ill-defined terms are mapped to a particular element called $\perp_A$.
There exist different interpretations in the literature, mainly Kleene's ([Kle52]) and Mc Carthy's ([McC63]) interpretations. We choose Kleene's interpretation for the formulae of our language (it is recalled in appendix A). This choice gives us a definition of formula ill-definedness. We note it $\mathcal{I}_v$. $\mathcal{I}_v$ interprets formula as true, false or ill-defined (t,f or $\perp$).
In a three-valued logic, unlike in a two-valued logic, different validity notions may be defined. The validity notion depends on the way ill-definedness is treated. The different consequence relations (and consequently validity notions) for three-valued logics are described in [GL90] (we recall their definitions in appendix B). We choose the WS consequence relation; this consequence relation is particularly described in [Owe93].

3 A Sound Deduction System

We are now looking for a deduction system which is sound in the WS consequence relation. We choose to express it in a sequent calculus form. The first result is that the sequent calculus is not sound in WS. So we have to find another deduction system; to construct it, we define two operators called Δ Operators: the idea is to protect unsound sequent calculus rule with new antecedents of well-definedness.

3.1 WS Logic

The axiom rule is not sound in WS consequence relation. The rule is only valid when it is applied to a well-defined predicate, a predicate that is true or false for all interpretations.

$$\frac{}{\Gamma,\ P \vdash P}\ \textit{axiom rule}$$

One way to have an axiom rule in WS is to guard it by an antecedent of well-definedness. So one can construct an operator Δ_p that respects the next proposition:

Proposition 1. *For each formula* φ*,*

- $\mathcal{I}_v(\Delta_p\varphi) = \mathrm{t}$ *if and only if* $\mathcal{I}_v(\varphi) \neq \perp$
- $\mathcal{I}_v(\Delta_p\varphi) = \mathrm{f}$ *if and only if* $\mathcal{I}_v(\varphi) = \perp$
- $\mathcal{I}_v(\Delta_p\varphi) \neq \perp$

With such an operator, we can construct a sound deduction system for WS.

Definition 1. *The deduction system* $\vdash_{WS}$*, is defined in [Owe93] and called WS-Logic. We only recall (figure 1) the rules of the sequent calculus that have to be changed to be sound (one could refer to [Owe93] for the complete sequent calculus).*

$$\frac{\Gamma \vdash_{WS} \Delta_p\varphi}{\Gamma,\ \varphi \vdash_{WS} \varphi}\ \mathit{well\ \ def}$$

$$\frac{\Gamma \vdash_{WS} \Delta_p\varphi}{\Gamma \vdash_{WS} \varphi \vee \neg\varphi}\ \mathit{excluded\ \ middle}$$

Fig. 1. WS-Logic

Theorem 1. $\Gamma \models_{WS} \varphi$ *if and only if* $\Gamma \vdash_{WS} \varphi$

The proof can be found in [EO93].

3.2 $\boldsymbol{\Delta}$ Pseudo-Operators

The $\boldsymbol{\Delta_\mathrm{p}}$ Pseudo-Operator We do not want to modify our language, so we do not add a new operator to our language as it is done in [Che86],[GL90] and [Hoo79]; but we define Δ_p as a pseudo-operator which assigns a formula of well-definedness to each formula. It should be taken as a macro that can always be expanded.
To do this another pseudo-operator Δ_e is defined, which has the same properties as Δ_p but on terms:

Proposition 2. *For each valuation* v*,*

- $\mathcal{I}_v(\Delta_e e) = \mathrm{t}$ *if and only if* $\mathcal{J}(e) \neq \perp_A$
- $\mathcal{I}_v(\Delta_e e) = \mathrm{f}$ *if and only if* $\mathcal{J}(e) = \perp_A$
- $\mathcal{I}_v(\Delta_e e) \neq \perp$

This new pseudo-operator is not sufficient to assign a formula of well-definedness to $p(e_1,...,e_n)$, thus we have to associate a predicate of well-definedness δ_p to each predicate p of P. This predicate represents the domain of P, it should be defined such that the next proposition holds:

Proposition 3. *For each valuation v, for all predicate $p \in P$ with $arity_p(p) = n$ and terms $e_1,...,e_n$ of T_Σ*

$$\mathcal{I}_v(\delta_p(e_1,...,e_n)) = \begin{cases} \text{t if and only if } \bigwedge_{i=1}^{n} \mathcal{J}(e_i) \neq \bot_A \Rightarrow \mathcal{I}_v(p(e_1,...,e_n)) \neq \bot \\ \text{f if and only if } \bigwedge_{i=1}^{n} \mathcal{J}(e_i) \neq \bot_A \Rightarrow \mathcal{I}_v(p(e_1,...,e_n)) = \bot \end{cases}$$

Remark 1. δ_p and p have the same arity.

With these new operators, we can define the pseudo-operator Δ_p

Definition 2. *$\Delta_\mathrm{p} :\ F_\Sigma \rightarrow F_\Sigma$ is defined as*

$$\begin{array}{ll} \Delta_\mathrm{p}(p(e_1,...,e_n)) = \bigwedge_{i=1}^{n} \Delta_\mathrm{e}(e_i) \wedge \delta_p(e_1,...,e_n) & \\ \Delta_\mathrm{p} True & = True \\ \Delta_\mathrm{p} False & = True \\ \Delta_\mathrm{p}(\varphi \wedge \psi) & = (\Delta_\mathrm{p}\varphi \wedge \Delta_\mathrm{p}\psi) \vee (\Delta_\mathrm{p}\varphi \wedge \neg\varphi) \vee (\Delta_\mathrm{p}\psi \wedge \neg\psi) \\ \Delta_\mathrm{p}(\varphi \vee \psi) & = (\Delta_\mathrm{p}\varphi \wedge \Delta_\mathrm{p}\psi) \vee (\Delta_\mathrm{p}\varphi \wedge \varphi) \vee (\Delta_\mathrm{p}\psi \wedge \psi) \\ \Delta_\mathrm{p}(\varphi \Rightarrow \psi) & = (\Delta_\mathrm{p}\varphi \wedge \Delta_\mathrm{p}\psi) \vee (\Delta_\mathrm{p}\varphi \wedge \neg\varphi) \vee (\Delta_\mathrm{p}\psi \wedge \psi) \\ \Delta_\mathrm{p}(\neg\varphi) & = \Delta_\mathrm{p}\varphi \\ \Delta_\mathrm{p}(\forall x.\varphi) & = \forall x.\Delta_\mathrm{p}\varphi \vee \exists x.(\Delta_\mathrm{p}\varphi \wedge \neg\varphi) \\ \Delta_\mathrm{p}(\exists x.\varphi) & = \forall x.\Delta_\mathrm{p}\varphi \vee \exists x.(\Delta_\mathrm{p}\varphi \wedge \varphi) \end{array}$$

This definition satisfies the proposition 1.

The $\boldsymbol{\Delta_\mathrm{e}}$ Pseudo-Operator. We define the Δ_e pseudo-operator on terms ($\Delta_\mathrm{e} : T_\Sigma \rightarrow F_\Sigma$). It assigns a formula of well-definedness to each term. As we have done with Δ_p, we assign a predicate of well-definedness δ_f to each function f of F. This predicate represents the domain of f, it should be defined such that this proposition holds:

Proposition 4. *For each valuation v, for all function $f \in F$ with $arity_f(f) = n$ and terms $e_1,...,e_n$ of T_Σ*

$$\mathcal{I}_v(\delta_f(e_1,...,e_n)) = \begin{cases} \text{t if and only if } \bigwedge_{i=1}^{n} \mathcal{J}(e_i) \neq \bot_A \Rightarrow \mathcal{J}(f(e_1,...,e_n)) \neq \bot_A \\ \text{f if and only if } \bigwedge_{i=1}^{n} \mathcal{J}(e_i) \neq \bot_A \Rightarrow \mathcal{J}(f(e_1,...,e_n)) = \bot_A \end{cases}$$

Remark 2. δ_f and f have the same arity.

So we can define Δ_e pseudo-operator.

Definition 3. *$\Delta_e : T_\Sigma \rightarrow F_\Sigma$ is inductively defined on T_Σ:*
$\Delta_e v = True$
$$\Delta_e f(e_1, ..., e_n) = \bigwedge_{i=1}^{n} \Delta_e e_i \wedge \delta_f(e_1, ..., e_n)$$

This definition satisfies the proposition 2.

3.3 Well Definedness formulae

We have seen that our language has to be completed with well-definedness predicates (δ_p and δ_f) associated to each predicate and function respectively. These predicates should respect the proposition 3 and 4 respectively.

Example 1. Let $\delta_{pred}(x)$ be $\neg(x = zero)$, $\delta_{succ}(x)$ be $True$, δ_{zero} be $True$, δ_{one} be $True$ and $\delta_{equal}(x, y)$ be $True$ then
$\Delta_e(succ(zero)) \Leftrightarrow True$,
$\Delta_e(pred(zero)) \Leftrightarrow False$,
$\Delta_p(pred(zero) = pred(zero)) \Leftrightarrow False$,
$\Delta_p(x = one \Rightarrow pred(x) = zero) \Leftrightarrow True$,
$\Delta_p(pred(zero) = zero \Rightarrow succ(zero) = zero) \Leftrightarrow False$,
$\Delta_p(pred(zero) = zero \Rightarrow succ(zero) = one) \Leftrightarrow True$,
$\Delta_p(succ(zero) = zero \Rightarrow pred(zero) = zero) \Leftrightarrow True$.

3.4 Conclusion

As we saw in the previous paragraphs the axiom and excluded middle rules are not sound in the WS-semantic, so they have to be guarded by the well-definedness of the disappearing formula. All the other rules of the classical calculus sequent are retained. Intuitively, a proof in WS-Logic is quite similar to a proof in first order logic, except at leaves, where well-definedness has to be proved. Its main advantage is that only the well-definedness of active formulae has to be proved. $\vdash_{WS}$ could be the deduction system of an automatic prover but it would be ineffective due to the invalidity of the axiom rule.

4 Two Proofs For One

Since the $WS-Logic$ cannot be the deduction system of an efficient automatic prover, we have looked for another way to prove our lemmas. Our intuition was that a first proof of well definedness would allow us to use a deduction system that no longer cares about ill-definedness. Our aim was to construct an implementable deduction system in which making two proofs, one of well-definedness and one of validity, would be equivalent to make one proof of validity in $\vdash_{WS}$.
So with the Δ pseudo-operators, we can in a first time check if a predicate is or is not well-defined. And then, if it is well defined, we can prove it. So we have

to make two proofs rather than one but in a simplified deduction system.
We now present the theorem which permits to make, as we wanted, two validity proofs in a consequence relation to ensure the validity in another one.

Theorem 2. *Given a set Γ of formulae and a formula φ,*
$\models_{WW} \Delta_p(\Gamma \Rightarrow \varphi)$ *and* $\Gamma \models_{WW} \varphi$ *if and only if*
$\models_{SW} \Delta_p(\Gamma \Rightarrow \varphi)$ *and* $\Gamma \models_{SW} \varphi$ *if and only if*
$\models_{SS} \Delta_p(\Gamma \Rightarrow \varphi)$ *and* $\Gamma \models_{SS} \varphi$ *if and only if*
$\Gamma \models_{WS} \varphi$

This theorem means that if a sequent is valid in WS then it is also valid in other consequence relations and its well-definedness is also valid in these other consequence relations. And vice versa, if a sequent and its well-definedness are valid in a consequence relation then the sequent is valid in WS. The three WW, SW and SS consequence relations have this property. As we can associate to each one a sound and valid deduction system, we have now to choose among the three deduction systems. We wa nt to find a more practical deduction system than WS is, even if two proofs are necessary instead of one.

5 The Other Deduction Systems

We can construct other deduction systems, one by validity notion. And we know that if we make two proofs of a sequent in this system, this sequent is valid in our sense.
We present the deduction systems SW, we only give the rules of the sequent calculus that are not still valid and that we have to guard with an antecedent of well-definedness. We give neither the complete deduction system nor the completeness proof but we cite the reference where they can be found. Every time, the soundness proof is obvious. The two remaining deduction systems SS and WW are presented in appendix C.

5.1 $\vdash_{SW}$

Definition 4. *We define the deduction system $\vdash_{SW}$ (figure 2) which is the sequent calculus where cut, $\forall - left$ and $\exists - right$ rules are guarded by the proof of the well-definedness of introduced formulae and terms.*

Theorem 3. $\Gamma \models_{SW} \varphi$ *if and only if* $\Gamma \vdash_{SW} \varphi$

The proof can be found in [GL90].

5.2 Three logics

We get three deduction systems and we know that making two proofs in each of this system is equivalent making one proof in $\vdash_{WS}$.

$$\frac{\Gamma \vdash_{SW} \Delta_p\varphi \quad \Gamma \vdash_{SW} \varphi \quad \Gamma,\ \varphi \vdash_{SW} \psi}{\Gamma \vdash_{SW} \psi}\ cut$$

$$\frac{\Gamma, \varphi[x \leftarrow t] \vdash_{SW} \psi \quad \Gamma \vdash_{SW} \Delta_p\varphi[x \leftarrow t]}{\Gamma,\ \forall x.\varphi \vdash_{SW} \psi}\ \forall\text{ -left}$$

$$\frac{\Gamma \vdash_{SW} \varphi[x \leftarrow t] \quad \Gamma \vdash_{SW} \Delta_p\varphi[x \leftarrow t]}{\Gamma \vdash_{SW} \exists x\ \varphi}\ \exists\text{ -right}$$

Fig. 2. SW Logic

Proposition 5. *Giving a set Γ of formulae and a formula φ,*
$\vdash_{WW} \Delta_p(\Gamma \Rightarrow \varphi)$ *and* $\Gamma \vdash_{WW} \varphi$ *if and only if*
$\vdash_{SW} \Delta_p(\Gamma \Rightarrow \varphi)$ *and* $\Gamma \vdash_{SW} \varphi$ *if and only if*
$\vdash_{SS} \Delta_p(\Gamma \Rightarrow \varphi)$ *and* $\Gamma \vdash_{SS} \varphi$ *if and only if*
$\Gamma \models_{WS} \varphi$

Now we have presented the three deduction systems and we have to choose one.

5.3 Why the SW Deduction System

To choose a deduction system, one can take the papers that present the SW, SS, WS and WW consequence relations and the sound deduction systems associated.
In all the papers related to this subject, the main criterion to elect a consequence relation is that one wants a deduction system that is as close as possible to the classical sequent calculus. We know that each consequence relation renders some of the classical sequent calculus rules unsound. Hence we have to choose a deduction system in terms of which rules are preserved and which ones are to be guarded by a well-definedness antecedent.
We should also reason at two levels: the deduction system should be effective when it is implemented in an automatic prover, and it should also be sensible and practical for an interactive proof usage.
Since the axiom rule is not sound in WS, any automatic prover that implements the WS-Logic would be ineffective. It is the reason why we eliminated it in paragraph 3.4.
Three other deduction systems remain. We eliminate the SS-Logic since it does not validate the deduction rule. We think that this rule is very useful in an interactive proof and that it would be quite constraining to prove a sublemma of well-definedness every time the rule is applied.
We do not elect WW since it neither validates the Modus Ponens rule, nor the rule which permits to use false hypothesis. These rules are also natural in an interactive proof session.
In fact, we choose the SW deduction system because the rules, which have to be guarded by an antecedent of well-definedness, are the rules which introduce new

terms or formulae in the proof. It seems very natural to start with checking the well-definedness of the proof inputs (initial formula and introduced terms and formulae) prior to using them. This is achieved firstly by proving the delta lemmas, and then by proving the well-definedness of introduced terms or formulae through the antecedent of well-d efinedness of the guarded rules. This process can be related to the type-checking in first order logic with types. Moreover, the rules which introduce terms are useless in an automatic prover because they need to invent terms. So these are all the reasons why we choose the *SW* deduction system to make proof in the B method.

6 Application to the B method

In the B method, the lemmas to prove are generated from sources written in a language with partial functions (and total predicates). For each lemma generated, one could have generated a delta-lemma. But to factorise the lemmas, delta-lemmas are generated directly for the source such that delta-lemmas are minimal.
To construct these delta-lemmas, we do not use the Δ_p operator that we have defined but we define another operator which is more calculable. In fact it is a Mc Carthy Δ_p operator.

Definition 5. *$\Delta_{\mathrm{p}}^{MC}: F_\Sigma \to F_\Sigma$ is defined as*

$$\begin{array}{ll}
\Delta_{\mathrm{p}}^{MC}(p(e_1,...,e_n)) & = \bigwedge_{i=1}^{n} \Delta_{\mathrm{e}}(e_i) \wedge \delta_p(e_1,...,e_n) \\
\Delta_{\mathrm{p}}^{MC} True & = True \\
\Delta_{\mathrm{p}}^{MC} False & = True \\
\Delta_{\mathrm{p}}^{MC}(\varphi \wedge \psi) & = \Delta_{\mathrm{p}}^{MC}\varphi \wedge (\varphi \Rightarrow \Delta_{\mathrm{p}}^{MC}\psi) \\
\Delta_{\mathrm{p}}^{MC}(\varphi \vee \psi) & = \Delta_{\mathrm{p}}^{MC}\varphi \wedge (\neg\varphi \Rightarrow \Delta_{\mathrm{p}}^{MC}\psi) \\
\Delta_{\mathrm{p}}^{MC}(\varphi \Rightarrow \psi) & = \Delta_{\mathrm{p}}^{MC}\varphi \wedge (\varphi \Rightarrow \Delta_{\mathrm{p}}^{MC}\psi) \\
\Delta_{\mathrm{p}}^{MC}(\neg\varphi) & = \Delta_{\mathrm{p}}^{MC}\varphi \\
\Delta_{\mathrm{p}}^{MC}(\forall x.\varphi) & = \forall x.\Delta_{\mathrm{p}}^{MC}\varphi \\
\Delta_{\mathrm{p}}^{MC}(\exists x.\varphi) & = \forall x.\Delta_{\mathrm{p}}^{MC}\varphi
\end{array}$$

This operator introduces asymmetries since its value for $\varphi \wedge \psi$ can be different from its value for $\psi \wedge \varphi$. But, in every case, if its value on a formula is true, then this formula is well-defined in the sense of Kleene.

Proposition 6. *For each formula φ, $\Delta_{\mathrm{p}}^{MC}\varphi \Rightarrow \Delta_p\varphi$*

This proposition establishes that the Mc Carthy Δ_p is more restrictive than Kleene's one. So to check the well-definedness at the proof inputs, one can implement this operator which is more calculable. And this is sufficient to ensure the well-definedness with the Kleene sense. The main consequence on inputs is that one has to take care about well-definedness. But it is more natural to write $x \neq 0 \wedge 1/x \leq 2/x$ rather than $1/x \leq 2/x \wedge x \neq 0$.
One can find more details on the implementation of this solution in the B method in [BBM98].

7 Conclusion

We have defined a logic that allows to validate any development using the B method. We have chosen, to construct this logic, the consequence relation WS. We have defined a proof method that associates a proof of well-definedness to the "classical" proof of a lemma. We consider that the deduction system SW that we have chosen is the most adapted to an industrial context.
One could believe that proving, in addition to each lemma, a lemma of well definedness doubles the proof cost of the project. Our experience with an industrial software product shows clearly that there is no real overhead. The delta-lemma proof costs less than five percent of the lemma proof. In fact, a prototype that implements this method of validation has been partially realised and tested on an industrial project. On this project, the well-definedness lemmas found errors in the B sources, mainly in the specification parts. The much benefit on the proof of the main lemmas is that the user rules are simpler to write since the developers do not have to consider the well-definedness unless their rules introduce new terms. Another benefit that we expect of a total implementation of this method, is the factorisation of the well-definedness proof at the inputs of the proof rather than at the leaves as we said in chapter 5.
After this successful experimentation, a delta-lemma generator will be soon integrated to the commercial version of the B toolset developed by STERIA.

References

[Abr96] Abrial J.R., The B-Book (1996)

[BBM98] Behm P., Burdy L., Meynadier J.M. Well Defined B; LNCS 1393 (1998)

[BCJ84] Barringer H., Cheng J.H., Jones C.B., A Logic Covering Undefinedness in Program Proofs; Acta Informatica 21:251-269 (1984)

[Bla86] Blamey S., Partial Logic; Handbook of Philosophical Logic 3:1-70 (1986)

[BKT88] Blikle A., Konikowska B., Tarlecki A., A Three Valued Logic for Software Specification and Validation; LNCS 328 (1988)

[Che86] Cheng J.H., A Logic for Partial Functions; Ph.D. Thesis, Report UMCS-86-7-1, Dept of CS, University of Manchester (1986)

[CJ90] Cheng J.H., Jones C.B., On the Usability of Logics Which Handle Partial Functions; Proceedings of the Third Refinement Workshop 51-69 (1990)

[EO93] Elvang-Gøransson M., Owe O., A Simple Sequent Calculus for Partial Functions; TCS 114:317-330 (1993)

[GL90] Gavilanes-Franco A., Lucio-Carrasco F. A First Order Logic for Partial Functions; TCS 74 37-69 (1990)

[Hol91] Holden M., Weak Logic Theory; TCS 79:295-321 (1991)

[Hoo79] Hoogewijs A., On a formalization of the non-definedness notion; Z. Logik Grundlagen Math. 25:213-221 (1979)

[Hoo87] Hoogewijs A., A Partial Predicate Logic in Computer Science; Acta Informatica 24:381-393 (1987)

[Jon95] Jones C.B. Partial functions and logics: a warning. Information Processing Letters 54, 65-67 (1995)

[Kle52] Kleene S.C., Introduction to Metamathematics (1992)

[McC63] Mc Carthy J., A Basis for a Mathematical Theory of Computation; Computer Programming and Formal Systems 33-70 (1963)
[Owe85] Owe O., An approach to program reasoning based on first order logic for partial functions,
Research Report No. 89 University of Oslo (1985)
[Owe93] Owe O., Partial Logics Reconsidered: a Conservative Approach,
Formal Aspects of Comput. 5:208-223 (1993)
[Wan61] Wang H. The calculus of partial predicates and its extension to set theory. Logik. Grundlagen Math. 7:283-288 (1961)

A Kleene Semantics

Note 1. $v \Lleftarrow \{x \mapsto a\}$ is a valuation such that $v \Lleftarrow \{x \mapsto a\}(y) = a$ if $x = y$ and $v(y)$ else.

$$
\begin{aligned}
\mathcal{I}_v(p(e_1, ..., e_n)) &= \begin{cases} p_A(\mathcal{J}_v(e_1), ..., \mathcal{J}_v(e_n)) \textit{ if for all } i \in 1..n,\ \mathcal{J}_v(e_i) \neq \perp_A \\ \perp \textit{ otherwise} \end{cases} \\
\mathcal{I}_v(\varphi \wedge \psi) &= \begin{cases} \mathrm{t} \textit{ if } \mathcal{I}_v(\varphi) = \mathrm{t} \textit{ and } \mathcal{I}_v(\psi) = \mathrm{t} \\ \mathrm{f} \textit{ if } \mathcal{I}_v(\varphi) = \mathrm{f} \textit{ or } \mathcal{I}_v(\psi) = \mathrm{f} \\ \perp \textit{ otherwise} \end{cases} \\
\mathcal{I}_v(\varphi \vee \psi) &= \begin{cases} \mathrm{t} \textit{ if } \mathcal{I}_v(\varphi) = \mathrm{t} \textit{ or } \mathcal{I}_v(\psi) = \mathrm{t} \\ \mathrm{f} \textit{ if } \mathcal{I}_v(\varphi) = \mathrm{f} \textit{ and } \mathcal{I}_v(\psi) = \mathrm{f} \\ \perp \textit{ otherwise} \end{cases} \\
\mathcal{I}_v(\varphi \Rightarrow \psi) &= \begin{cases} \mathrm{t} \textit{ if } \mathcal{I}_v(\varphi) = \mathrm{f} \textit{ or } \mathcal{I}_v(\psi) = \mathrm{t} \\ \mathrm{f} \textit{ if } \mathcal{I}_v(\varphi) = \mathrm{t} \textit{ and } \mathcal{I}_v(\psi) = \mathrm{f} \\ \perp \textit{ otherwise} \end{cases} \\
\mathcal{I}_v(\neg\varphi) &= \begin{cases} \mathrm{t} \textit{ if } \mathcal{I}_v(\varphi) = \mathrm{f} \\ \mathrm{f} \textit{ if } \mathcal{I}_v(\varphi) = \mathrm{t} \\ \perp \textit{ otherwise} \end{cases} \\
\mathcal{I}_v(\forall x.\varphi) &= \begin{cases} \mathrm{t} \textit{ if for all } a \in A,\ \mathcal{I}_{v \Lleftarrow \{x \mapsto a\}}(\varphi) = \mathrm{t} \\ \mathrm{f} \textit{ if there exists an } a \in A \textit{ such that } \mathcal{I}_{v \Lleftarrow \{x \mapsto a\}}(\varphi) = \mathrm{f} \\ \perp \textit{ otherwise} \end{cases} \\
\mathcal{I}_v(\exists x.\varphi) &= \begin{cases} \mathrm{t} \textit{ if there exists an } a \in A \textit{ such that } \mathcal{I}_{v \Lleftarrow \{x \mapsto a\}}(\varphi) = \mathrm{t} \\ \mathrm{f} \textit{ if for all } a \in A,\ \mathcal{I}_{v \Lleftarrow \{x \mapsto a\}}(\varphi) = \mathrm{f} \\ \perp \textit{ otherwise} \end{cases}
\end{aligned}
$$

B Consequence Relations

Definition 6. *The WW, WS, SS, SW semantics are defined in [GL90] :*
$\mathit{\Gamma} \models_{WW} \varphi$ *if and only if for each valuation* v, $\mathcal{I}_v(\mathit{\Gamma}) = \mathrm{f}$ *or* $\mathcal{I}_v(\varphi) \neq \mathrm{f}$
$\mathit{\Gamma} \models_{WS} \varphi$ *if and only if for each valuation* v, $\mathcal{I}_v(\mathit{\Gamma}) = \mathrm{f}$ *or* $\mathcal{I}_v(\varphi) = \mathrm{t}$
$\mathit{\Gamma} \models_{SS} \varphi$ *if and only if for each valuation* v, $\mathcal{I}_v(\mathit{\Gamma}) \neq \mathrm{t}$ *or* $\mathcal{I}_v(\varphi) = \mathrm{t}$
$\mathit{\Gamma} \models_{SW} \varphi$ *if and only if for each valuation* v, $\mathcal{I}_v(\mathit{\Gamma}) \neq \mathrm{t}$ *or* $\mathcal{I}_v(\varphi) \neq \mathrm{f}$

C Deduction Systems

C.1 $\vdash_{SS}$

$$\frac{\Gamma \vdash_{SS} \Delta_p \varphi}{\Gamma \vdash_{SS} \varphi \vee \neg\varphi} \; \textit{excluded middle}$$

$$\frac{\Gamma \vdash_{SS} \Delta_p \varphi \quad \Gamma,\ \varphi \vdash_{SS} \psi}{\Gamma \vdash_{SS} \varphi \Rightarrow \psi} \Rightarrow \textit{-right}$$

$$\frac{\Gamma, \varphi \vdash_{SS} False \quad \Gamma \vdash_{SS} \Delta_p \varphi}{\Gamma \vdash_{SS} \neg\varphi} \neg \textit{-right}$$

$$\frac{\Gamma, \varphi[x \leftarrow t] \vdash_{SS} \psi \quad \Gamma \vdash_{SS} \Delta_p \varphi[x \leftarrow t]}{\Gamma,\ \forall x.\varphi \vdash_{SS} \psi} \forall \textit{-left}$$

Fig. 3. SS Logic [Che86]

C.2 $\vdash_{WW}$

$$\frac{\Gamma \vdash_{WW} \Delta_p \varphi \quad \Gamma \vdash_{WW} \varphi \quad \Gamma,\ \psi \vdash_{WW} R}{\Gamma, \varphi \Rightarrow \psi \vdash_{WW} R} \Rightarrow \textit{-left}$$

$$\frac{\Gamma \vdash_{WW} \varphi \quad \Gamma \vdash_{WW} \Delta_p \varphi}{\Gamma, \neg\varphi \vdash_{WW} \psi} \neg \textit{-left}$$

$$\frac{\Gamma \vdash_{WW} \varphi[x \leftarrow t] \quad \Gamma \vdash_{WW} \Delta_p \varphi[x \leftarrow t]}{\Gamma \vdash_{WW} \exists x\ \varphi} \exists \textit{-right}$$

Fig. 4. WW Logic [Hol91]