Step-index like semantics for our assertions

Work in progress

SOPHIA D, HOLISTIC GROUP, and HELP FROM ALEX S. AND KRYSIA B., zzz

How we define the semantics for assertions through step-indexing or similar ideas. The ensuing logic is not classical. We propose some alternatives, and discuss their ramifications. In this version we do not include the **obeys** -predicate.

ACM Reference Format:
Sophia D, Holistic Group, and help from Alex S. and Krysia B.. 2025. Step-index like semantics for our assertions: Work in progress. 1, 1 (June 2025), 9 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

1.1 Preliminaries

We have the following entities:

- expressions e with the expected syntax, and
- assertions A,
- modules `M` which hold the definitions of classes, methods and functions, and the function $\mathcal{B}ody(\texttt{M}, \texttt{id})$ which returns the body of a function or a predicate `id`, as defined in `M`.
- states σ which contain a stack of frames. where each frame has the receiver, the method id, the arguments, the local variables, and the code being executed

The above are more or less as described in the July documents. For clarity we define assertions A (we skip some parts of the syntax for the time being)

DEFINITION 1 (ASSERTIONS). *have the following syntactic form*
A `:;= true | false | e |` $A \rightarrow A$ `|` $A \vee A$ `|` $\neg A$ `|PredIde`.

1.2 Examples

In this paper we will consider functions, predicates and states, described below.
The function `acyclic` returns `true` if the list is acyclic, and is defined defined as

```
acyclic(x) ≡
if x:List then (if x.next==null then true else acyclic(x.next)) else false.
```

The function `length` returns the length of a list, and is defined defined as

```
length(x) ≡
if x:List then (if x.next==null then 0 else 1+length(x.next)) else 0
```

We also define the predicates `Finite` and `Finite′` similar to the function `acylcic`, as follows:

```
Finite(x) ≡ x.next ≠ null → Finite(x.next)).
Finite′(x) ≡ x.next = null ∨ Finite′(x.next).
```

Authors' address: Sophia D; Holistic Group; help from Alex S. and Krysia B.zzz.

XXXX-XXXX/2025/6-ART $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

And we define the contradictory predicate `Cntrdct`,
`Cntrdct(x)` ≡ ¬`Cntrdct(x)`.

, Moreover, we assume a state σ_{ac} such that σ_{ac}(`z.next.next`)=`null`[1] And we assume another state, σ_{cyc}, where `z` is a cyclic `List` object, and $\sigma_{cyc}(\sigma_{cyc}(\texttt{z}), \texttt{next}) = \sigma_{cyc}(\texttt{z})$.

1.3 What we wiil do

We now define the judgments

- $\texttt{M}, \sigma, e \rightsquigarrow_k v$ which says that e evaluates to v in up to k steps
- $\texttt{M}, \sigma \models_k A$ which says that validity of A in $\texttt{M}, \sigma$ can be established in up to k steps.
- $\texttt{M}, \sigma \models A$ which says that A is valid in σ.
- $\texttt{M}, \sigma \rightsquigarrow_k \sigma'$ which says that state σ evaluates to new state σ' in up to k steps

We will not give complete definitions here, but show the most salient parts below.

2 EVALUATION OF EXPRESSIONS

DEFINITION 2. *Given a module* `M`*, a state* σ*, and a natural number* k *we define the judgment* $\texttt{M}, \sigma, e \rightsquigarrow_k v$ *by cases on* `e` *as follows:*

- $\texttt{M}, \sigma, \texttt{x} \rightsquigarrow_k v$ *iff* $\texttt{x} \in Dom(\sigma)$ *and* $\sigma(\texttt{x})$=v*, for any value of* k.
- $\texttt{M}, \sigma, \texttt{e.f} \rightsquigarrow_k v$ *iff* $\texttt{M}, \sigma, \texttt{e} \rightsquigarrow_k \iota$ *for some* ι*, and* $\sigma(\iota, \texttt{f})$=$v$.
- $\texttt{M}, \sigma, \texttt{e1 == e2} \rightsquigarrow_k \texttt{true}$ *iff* $\texttt{M}, \sigma, \texttt{e1} \rightsquigarrow_k v$, *and* $\texttt{M}, \sigma, \texttt{e2} \rightsquigarrow_k v$ *for some* v.
- $\texttt{M}, \sigma, \texttt{e1 == e2} \rightsquigarrow_k \texttt{false}$ *iff* $\texttt{M}, \sigma, \texttt{e1} \rightsquigarrow_k v1$, *and* $\texttt{M}, \sigma, \texttt{e2} \rightsquigarrow_k v2$, *and* $v1 \neq v2$.
- $\texttt{M}, \sigma, \texttt{if e0 then e1 else e2} \rightsquigarrow_k v$ *iff* $\texttt{M}, \sigma \rightsquigarrow_{k-1} \texttt{true}$ *and* $\texttt{M}, \sigma, \texttt{e1} \rightsquigarrow_{k-1} v$ *or* $\texttt{M}, \sigma \rightsquigarrow_{k-1} \texttt{false}$ *and* $\texttt{M}, \sigma, \texttt{e2} \rightsquigarrow_{k-1} v$, *and* $k \geq 1$.
- $\texttt{M}, \sigma, \texttt{fnId(e)} \rightsquigarrow_k v$ *iff* $\texttt{M}, \sigma, \texttt{e} \rightsquigarrow_{k-1} v'$ *and* $\mathcal{B}ody(\texttt{M}, \texttt{fnId})$=$\texttt{e}'$ *and* $\texttt{M}, \sigma[\texttt{z} \mapsto v'], \texttt{e}'[\texttt{z}/\texttt{x}] \rightsquigarrow_{k-1} v$ *and* `z` *free in* `e` *and* σ, *and* $k \geq 1$.

We also define, more generally:

- $\lfloor e \rfloor_{\texttt{M},\sigma}$=$v$ *iff* $\exists k. \forall j \geq k. \texttt{M}, \sigma, \texttt{e} \rightsquigarrow_j v$.

Note that expressions have no side-effects. The evaluation $\rightsquigarrow_k$ is monotonic, in the sense that if execution of up to k steps returns a result, then any larger number of steps returns the same result.

LEMMA 3 (EXPRESSION EVALUATION IS MONOTONIC). *For any module* `M`*, any state* σ*, expression* `e`*, any natural numbers* k*,* j *and values* v *and* v' *we have:*

- $\texttt{M}, \sigma, \texttt{e} \rightsquigarrow_k v \longrightarrow \texttt{M}, \sigma, \texttt{e} \rightsquigarrow_{k+j} v$.
- $\texttt{M}, \sigma, \texttt{e} \rightsquigarrow_j v \wedge \texttt{M}, \sigma, \texttt{e} \rightsquigarrow_k v' \longrightarrow v$=$v'$

PROOF. The lemma is easy to prove. Just a sketch below

- By well-founded ordering introduced on the structure of `e` and value of k[2]. Only the base case, where `e`=`v`, and the function calls are a bit interesting cases.
- Similar argument.

□

[1]This needs to be written better.
[2]Here to find and use the right term from the step-index lit. - but it is well-founded ordering thing.

In some cases, there exists no k and v such that $\mathtt{M}, \sigma, \mathtt{x} \rightsquigarrow_k v$; therefore the expression $\lfloor e \rfloor_{\mathtt{M},\sigma}$ is not always defined. For example, the terms $\lfloor \mathtt{acyclic(z)} \rfloor_{\mathtt{M},\sigma_{cyc}}$ and $\lfloor \mathtt{lenght(z)} \rfloor_{\mathtt{M},\sigma_{cyc}}$ are undefined.[3] And the term $\lfloor \mathtt{if\ acyclic(z)\ then\ false\ else\ true} \rfloor_{\mathtt{M},\sigma_{cyc}}$, which corresponds to the negation of `acyclic` is also undefined.

3 CODE EXECUTION

We now define what it means to execute code.[4]

DEFINITION 4. *Given a module* $\mathtt{M}$ *a state* σ *and a natural number* k *we define the judgment* $\mathtt{M}, \sigma \rightsquigarrow_k \sigma'$ *by cases on* $\sigma.\mathtt{code}$ *as follows:*

- $\mathtt{M}, \sigma \rightsquigarrow_k \sigma$ *for all* k
- *For all* k,
  if $\sigma[\mathtt{x}, \mathtt{f} \mapsto \sigma(\mathtt{y}), \mathtt{code} \mapsto \mathtt{stmts}] \rightsquigarrow_k \sigma'$, *then* $\sigma[\mathtt{code} \mapsto \mathtt{x.f=y;\ stmts}] \rightsquigarrow_{k+1} \sigma'$
- *... skipping the rest*

LEMMA 5 (CODE EXECUTION IS MONOTONIC). *For any module* $\mathtt{M}$*, any states* σ*,* σ'*, and* σ''*, and any natural numbers* k*,* j *we have:*
$\mathtt{M}, \sigma \rightsquigarrow_k \sigma'$ *and* $\mathtt{M}, \sigma \rightsquigarrow_j \sigma''$ *and* $j \geq k$ *imply that* $\mathtt{M}, \sigma' \rightsquigarrow_{j-k} \sigma''$.

PROOF. TODO. □

4 SEMANTICS OF ASSERTIONS THROUGH STEP-INDEXING

Here step-indexing is important and interesting. The basic structure of the definition has to follow the LICS style, and add a case for predicates.[5] There is a large design space:

(1) Do we define implication a) in terms of the "next" world, as in Kripke semantics, or b) in terms of the previous world, as in the LICS paper?
(2) For the base case, i.e. for $\mathtt{M}, \sigma \models_0 A$, do we have
  (a) $\mathtt{M}, \sigma \models_0 A$ always (as in LICS), or
  (b) $\mathtt{M}, \sigma \models_0 A$ never (this is inductive, and firs the execution of expressions),
  (c) $\mathtt{M}, \sigma \models_0 A$ iff A is an expression e and $\mathtt{M}, \sigma, e \rightsquigarrow_0 \mathtt{true}$.
(3) For implication, we have
  a) $\mathtt{M}, \sigma \models_k A \to A'$ iff $\mathtt{M}, \sigma \models_{k-1} A$ implies that $\mathtt{M}, \sigma \models_{k-1} A'$, or
  b) $\mathtt{M}, \sigma \models_k A \to A'$ iff $\forall j < k.\ \mathtt{M}, \sigma \models_j A$ implies that $\mathtt{M}, \sigma \models_j A'$
(4) For disjunction do we have
  a) $\mathtt{M}, \sigma \models_k A \vee A'$ iff $\mathtt{M}, \sigma \models_k A$ or $\mathtt{M}, \sigma \models_k A'$, or
  b) $\mathtt{M}, \sigma \models_k A \vee A'$ iff $\forall j < k.\ \mathtt{M}, \sigma \models_j A$ or $\mathtt{M}, \sigma \models_j A'$
(5) For validity, do we have
  a) $\mathtt{M}, \sigma \models A'$ iff $\forall k \in \mathbb{N}.\ \mathtt{M}, \sigma \models_k A$, or
  b) $\mathtt{M}, \sigma \models A'$ iff $\exists k.\ \forall j \geq k.\ \mathtt{M}, \sigma \models_j A$

Wrt (1), I cannot see how the definition in terms of a future world could work for u, because most of predicates (e.g. `Finite`) have an inductive (least fixpoint reading). Therefore I discard this avenue. Similarly, So, there are 18 alternatives.

[3] In Summers-Drossopoulu-ECOOP11 we had special error values for such cases.
[4] Or perhaps we can be parametric with that. This sections is not that interesting for the time being.
[5] Note the in Kripke predicate logics the predicates have an interpretation, and no bodies.

In the below, I am choosing 2c)-3a)-4b)-5b). I believe that the only other sensible alternative is to chose 2c)-3a)-4a)-5b. [6] [7]

4.1 Step-indexing with 2c)-3a)-4b)-5b

DEFINITION 6. *Given a module* M *a state* σ *and a natural number* k *we define the judgment* $\mathtt{M}, \sigma \models_k A$ *by cases on* A *and* k *as follows:*
For all k *we have*

- $\mathtt{M}, \sigma \models_k e$ *iff* $\mathtt{M}, \sigma, e \rightsquigarrow_k \mathtt{true}$.[8]

For $k \geq 1$ *we define*

- $\mathtt{M}, \sigma \models_k A \to A'$ *iff* $\mathtt{M}, \sigma \models_{k-1} A$ *implies that* $\mathtt{M}, \sigma \models_{k-1} A'$.
- $\mathtt{M}, \sigma \models_k A \vee A'$ *iff* $\forall j \leq k.\ \mathtt{M}, \sigma \models_k A$ *or* $\forall j \leq k.\ \mathtt{M}, \sigma \models_k A'$.
- $\mathtt{M}, \sigma \models_k \mathtt{PredId(e)}$ *iff* $\exists v.\ \mathtt{M}, \sigma, \mathtt{e} \rightsquigarrow_{k-1} v$ *and* $\mathcal{B}ody(\mathtt{M}, \mathtt{PredId}) = \mathtt{A}$ *and* $\mathtt{M}, \sigma[\mathtt{z} \mapsto v'] \models_{k-1} \mathtt{A[z/x]}$ *and* z *free in* A *and* σ.

As well, we define

- $\mathtt{M}, \sigma \not\models_k A$ *iff* $\mathtt{M}, \sigma \models_k A$ *does not hold.*

And also

- $\mathtt{M}, \sigma \models A$ *iff* $\exists k. \forall j \geq k.\ \mathtt{M}, \sigma \models_j A$.

NOTE 7 (THE MEANING OF NEGATION). *We use* $\neg A$ *as shorthand for* $A \to \mathtt{false}$. *Therefore, we have that* $\mathtt{M}, \sigma \models_k \neg A$ *iff* $\mathtt{M}, \sigma \models_k A \to \mathtt{false}$, *which means that* $\mathtt{M}, \sigma \models_k \neg A$ *iff* $\mathtt{M}, \sigma \not\models_k A$.

Discussion

EXAMPLE 8 (ACYCLIC AND FINITE). *In the acyclic configuration* σ_{acyc}, *we can establish that* x *is finite after three steps. That is*

(A)	$\mathtt{M}, \sigma_{acyc} \models_0 \mathtt{x.next.next = null}$,	*by def 6, base case.*
(A')	$\mathtt{M}, \sigma_{acyc} \not\models_0 \mathtt{x.next = null}$,	*by def 6, base case.*
(B)	$\mathtt{M}, \sigma_{acyc} \not\models_0 \mathtt{Finite(x.next)}$,	*by def 6, base case.*
(C)	$\mathtt{M}, \sigma_{acyc} \not\models_0 \mathtt{Finite(x)}$,	*by def 6, base case.*
(D)	$\mathtt{M}, \sigma_{acyc} \models_1 \mathtt{Finite(x.next)}$,	*by A, and def 6 ind. case for predicates, and inductive case for implications*
(E)	$\mathtt{M}, \sigma_{acyc} \not\models_1 \mathtt{Finite(x)}$,	*by A' and C, and def 6 ind. case for predicates and inductive case for implications*
(F)	$\mathtt{M}, \sigma_{acyc} \models_2 \mathtt{Finite(x)}$,	*by A' and D, and def 6 ind. case for predicates and inductive case for implications*
(G)	$\mathtt{M}, \sigma_{acyc} \models_3 \mathtt{Finite(x)}$,	*by A' and F, and def 6 ind. case for predicates and inductive case for implications*
...	...	

Therefore, $\mathtt{M}, \sigma_{acyc} \models_k \mathtt{Finite(x)}$, *for all* $k \geq 2$., *which gives us that* $\mathtt{M}, \sigma_{acyc} \models \mathtt{Finite(x)}$.
In the cyclic configuration, as we saw earlier, $\lfloor \mathtt{acyclic(z)} \rfloor_{\sigma_{cyc}}$ *is undefined. Nevertheless, we* $\mathtt{M}, \sigma_{cyc} \models_k \neg \mathtt{Finite}(z)$ *for all* k. *We also have that* $\mathtt{M}, \sigma_{cyc} \models_k \neg \mathtt{length}(z) = 3$, *even though* length *and also* $\mathtt{length}(z) = 3$ *are ubdefined in the context of* σ_{cyc}. *This gives us also that* $\mathtt{M}, \sigma \models \neg \mathtt{Finite}(z)$, *and* $\mathtt{M}, \sigma_{cyc} \models_k \neg \mathtt{length}(z) = 3$.

[6] I do not like 2a) as it would give that $\mathtt{M}, \sigma \models_0 \neg A$ as well as $\mathtt{M}, \sigma \models_0 A$. Similarly, 2b) gives $\mathtt{M}, \sigma \models_0 \neg A$ as well as $\mathtt{M}, \sigma \models_0 \neg\neg A$

[7] In the LICS paper they start with $\mathtt{M}, \sigma \models_0 A$ always. But they do not have predicates, let alone recursive ones. But if we did that, then we would get, as shown below, that $\mathtt{M}, \sigma \models \mathtt{Acyclic(z)}$. And it makes sense to have a base case of invalid, as we have not yet dedicated enough fuel to investigate...

[8] This rule implies that $\mathtt{M}, \sigma \models_k \mathtt{true}$ always and $\mathtt{M}, \sigma \models_k \mathtt{false}$.

EXAMPLE 9 (CNTRDR). *We have* $\mathtt{M}, \sigma \not\models \mathtt{Cntrdct(z)}$ *as well as* $\mathtt{M}, \sigma \not\models \neg\mathtt{Cntrdct(z)}$. *Namely,*

(0) $\mathtt{M}, \sigma \not\models_0 \mathtt{Cntrdct(z)}$, *by def 6, base case.*
(1) $\mathtt{M}, \sigma \models_1 \mathtt{Cntrdct(z)}$, *by (0) and def 6, inductive case for predicates.*
(2) $\mathtt{M}, \sigma \models_2 \neg\mathtt{Cntrdct(z)}$, *by (1) and def 6, inductive case for predicates, negation, implication and* `false`
(3) $\mathtt{M}, \sigma \models_3 \mathtt{Cntrdct(z)}$, *by (2) and def 6, inductive step case for predicates, negation and implication.*

...

So we obtain that

(F1) $\mathtt{M}, \sigma \models_k \mathtt{Cntrdct(z)}$ *for all odd k, and* $\mathtt{M}, \sigma \models_k \neg\mathtt{Cntrdct(z)}$ *for all even k.*

Therefore,

(F2) $\mathtt{M}, \sigma \not\models \mathtt{Cntrdct(z)}$ *als well as* $\mathtt{M}, \sigma \not\models \neg\mathtt{Cntrdct(z)}$.

And also:

(F3) $\mathtt{M}, \sigma \not\models \mathtt{Cntrdct(z)} \vee \neg\mathtt{Cntrdct(z)}$.

EXAMPLE 10 (`Acyclic'`). *In the acyclic configuration* σ_{acyc} *we cannot estabish the predicate* `Acyclic'`. *This is because of our choice of 4b. In more detail.*

(A) $\mathtt{M}, \sigma_{acyc} \not\models_k \mathtt{x.next} = \mathtt{null}$, *for all k, by def 6, base case.*
(B) $\mathtt{M}, \sigma_{acyc} \not\models_0 \mathtt{Finite'(x)}$, *by def 6, base case.*
(C) $\mathtt{M}, \sigma_{acyc} \not\models_1 \mathtt{Finite'(x)}$, *by A, B, and def 6 ind. case for predicates, and inductive case for disjunction*
(D) $\mathtt{M}, \sigma_{acyc} \not\models_2 \mathtt{Finite'(x)}$, *by A, C, and def 6 ind. case for predicates, and inductive case for disjunction*

Therefore $\forall k.\mathtt{M}, \sigma_{acyc} \not\models_k \mathtt{Finite'(x)}$, *which gives that* $\mathtt{M}, \sigma_{acyc} \not\models \mathtt{Finite'(x)}$

OBSERVATION 11 (LACK OF MONOTONICITY). *In contrast with LICS'09 (and all other step-index works I know), the judgment* $\mathtt{M}, \sigma \models_k A$ *is not monotonic with k. Namely, it is possible to have that* $\mathtt{M}, \sigma \models_k A$, *as well as* $\mathtt{M}, \sigma \not\models_{k+j} A$.
An example can be seen in (F1).

OBSERVATION 12 (LACK OF MODUS PONENS). *We do not have modus ponens. A counter-example appears in (F2) and in (F3).*

Question Do we have modus ponens "in the steps", ie it does not hold for all k that either $\mathtt{M}, \sigma \models_k A$ *or* $\mathtt{M}, \sigma \models_k \neg(A)$*?*

OBSERVATION 13 (DISTRIBUTIVITY OF DISJUNCTION). *In our system, disjunction distributes, ie we have*

LEMMA 14 (DISJUNCTION IS DISTRIBUTIVE). *For all A.*

- $\mathtt{M}, \sigma \models_k A \vee A'$ *implies* $\mathtt{M}, \sigma \models_k A$ *or* $\mathtt{M}, \sigma \models_k A'$ *for all k.*
- $\mathtt{M}, \sigma \models A \vee A'$ *implies* $\mathtt{M}, \sigma \models A$ *or* $\mathtt{M}, \sigma \models A'$

Note that if we had chosen 4a) rather than 4b), the above would not hold, Namely, we call $\models_{alt}$ *the inference system 2c)-3a)-4b)-5b), and obtain:*

(F4) $\mathtt{M}, \sigma \models_{alt,k} \mathtt{Cntrdct(z)} \vee \neg\mathtt{Cntrdct(z)}$ *for all k.*

Which gives that,

(F5) $\mathtt{M}, \sigma \models_{alt} \mathtt{Cntrdct(z)} \vee \neg\mathtt{Cntrdct(z)}$.

And also:

(F6) $\mathtt{M}, \sigma \not\models_{alt} \mathtt{Cntrdct(z)}$ *and* $\mathtt{M}, \sigma \not\models_{alt} \neg\mathtt{Cntrdct(z)}$.

CONJECTURE 15 (CONSISTENCY). *For all* M, *A, A' and* σ.

- *It is never the case that* $\mathtt{M}, \sigma \models A$ *and* $\mathtt{M}, \sigma \models \neg A$.
- $\mathtt{M}, \sigma \models A \rightarrow A'$ *and* $\mathtt{M}, \sigma \models A$ *entails* $\mathtt{M}, \sigma \models A'$
- $\mathtt{M}, \sigma \models \neg A \vee A'$ *entails* $\mathtt{M}, \sigma \models A \rightarrow A'$
- $\mathtt{M}, \sigma \models A \vee A'$ *implies* $\mathtt{M}, \sigma \models A$ *or* $\mathtt{M}, \sigma \models A'$

PROOF. This is a sketch

- We will show that for all k it is impossible to have $\mathtt{M}, \sigma \models_k A$ and $\mathtt{M}, \sigma \models_k \neg A$.
- Expecting this to work by unfolding the definitions.
- Expecting this to work by unfolding the definitions. This property is less crucial tan the other two, but holds in intuitionistic logic.
- This property is less crucial tan the other two, but holds in intuitionistic logic.

□

4.2 Considering holistic aspects

In Definition 6 we have not considered the holistic aspects of assertions. We will therefore extend the definition as below

DEFINITION 16. *Given a module* $\mathtt{M}$*, a state* σ*, and a natural number* k*, we define the judgment* $\mathtt{M}, \sigma \models_k A$ *by cases on* A *and* k *as follows, where* $k > 0$ *and arbitrary:*

- *all cases from Definition 6, and in addition:*
- $\mathtt{M}, \sigma \models_0 \mathtt{x}$ ***obeys*** $\mathtt{S}$ *always.*
- $\mathtt{M}, \sigma \models_k \mathtt{x}$ ***obeys*** $\mathtt{S}$ *iff*
  for all σ'*, for all assertions* A *from* $\mathtt{S}$*, if* $\mathtt{M}, \sigma \rightsquigarrow_{k-1} \sigma'$ *then* $\mathtt{M}, \sigma' \models_{k-1} A[\mathtt{this} \mapsto \mathtt{x}]$.
- $\mathtt{M}, \sigma \models_k \Diamond A$ *iff* $\exists \sigma' . \mathtt{M}, \sigma \rightsquigarrow_{k-1} \sigma'$ *and* $\mathtt{M}, \sigma' \models_{k-1} A$.
- $\mathtt{M}, \sigma \models_k \nabla A$ *iff ... like, and a bit more complex than above, doable.*
- $\mathtt{M}, \sigma \models_k A$ in $\mathtt{S}$ *iff* $\mathtt{M}, \sigma|_S \models_k A$ *where* $\lfloor \mathtt{S} \rfloor_{\mathtt{M},\sigma} = S$ [9]

NOTE 17. *that we treat predicates differently to specifications. Namely, for all predicates* $\mathtt{P}$*, we have that* $\mathtt{M}, \sigma \not\models_0 \mathtt{P(x)}$*, while for any specification,* $\mathtt{S}$*, we have that* $\mathtt{M}, \sigma \models_0 \mathtt{x}$ ***obeys*** $\mathtt{S}$*. That is, we define validity of predicates inductively, and validity of assertions co-inductively. We take the smallest fixpoints for predicates and take the largest fixpoints for assertions.*

---

5 SEMANTICS OF ASSERTIONS THROUGH UNFOLDING

Here an alternative way of defining validity of assertions – less well thought at the moment. We define unfolding of assertions in the obvious way, and then validity of an assertion through validity of finite unfolding:

DEFINITION 18. *We define the relation* $\mathtt{M}, A \rightsquigarrow_k A'$ *as follows, where* $k \in \mathbb{N}$*:*

- $\mathtt{M}, x \rightsquigarrow_k x$*, and* $val \rightsquigarrow_k val$ *for any k, and for all* $val \in \{\mathtt{null}, \mathtt{true}, \mathtt{false}\}$.
- $\mathtt{M}, e.f \rightsquigarrow_k e'.f$ *if* $e \rightsquigarrow_k e'$*, for any k.*
- $\mathtt{M}, func(e) \rightsquigarrow_{k+1} e'$ *if* $e \rightsquigarrow_k e''$*, and* $FBody(fun)[x \mapsto e''] \rightsquigarrow_k e'$
- $\mathtt{M}, A \rightarrow A' \rightsquigarrow_{k+1} A'' \rightarrow A'''$ *iff* $\mathtt{M}, A \rightsquigarrow_k A''$ *and* $\mathtt{M}, A' \rightsquigarrow_k A'''$
- $\mathtt{M}, A \vee A' \rightsquigarrow_{k+1} A'' \vee A'''$ *iff* $\mathtt{M}, A \rightsquigarrow_k A''$ *and* $\mathtt{M}, A' \rightsquigarrow_k A'''$
- $\mathtt{PredId(e)} \rightsquigarrow_{k+1} A'$ *iff* $e \rightsquigarrow_k e'$*, and* $FBody(\mathtt{M}, \mathtt{PredId})[x \mapsto e'] \rightsquigarrow_k A'$

The $\rightsquigarrow_k$ relation is deterministic[10], i.e. $A \rightsquigarrow_k A'$ and $A \rightsquigarrow_k A''$ implies that A'=A''.

[9] We need to think a bit about $\lfloor S \rfloor_{\mathtt{M},\sigma}$. Also, should it be $\mathtt{M}, \sigma|_S \models_{k-1} A$, and does it make a difference?
[10] better word needed

DEFINITION 19. *We define three valued logic carrier set* {`true`, `false`, `b???`}*, and operations* $\vee$, $\rightarrow$ *as follows, where* `bval` *is an arbitrary value from* {`true`, `false`, `b???`}*:*

- *The operator* $\vee$
  - `true` $\vee$ `bval` $\triangleq$
  - `false` $\vee$ `bval` $\triangleq$ `bval`
  - `b???` $\vee$ `bval` $\triangleq$ `bval`
- *The operator* $\rightarrow$
  - `true` $\rightarrow$ `bval` $\triangleq$ `bval`
  - `false` $\rightarrow$ `bval` $\triangleq$ `true`
  - `b???` $\rightarrow$ `bval` $\triangleq$ `bval`

The operators $\wedge$ *and* $\neg$ *can be encoded.*

DEFINITION 20. *Given a state* σ *and an assertion* A *we define* $\langle A \rangle_\sigma$ *as follows*

- $\langle$`PredId(e)`$\rangle_\sigma \triangleq$ `b???`
- $\langle e \rangle_\sigma \triangleq$ `bval` *iff* $\emptyset, \sigma, e \rightsquigarrow_0$ `bval`.[11]
- $\langle A \rightarrow A' \rangle_\sigma \triangleq \langle A \rangle_\sigma \rightarrow \langle A' \rangle_\sigma$
- $\langle A \vee A' \rangle_\sigma \triangleq \langle A \rangle_\sigma \vee \langle A' \rangle_\sigma$

Given a module `M` *a state* σ *and a natural number* k *we define the judgment* $\langle$`PredId(e)`$\rangle_{\sigma,k}$ *as follows:*

- $\langle A \rangle_{\sigma,0} \triangleq \langle A' \rangle_\sigma$, *where* $A \rightsquigarrow_0 A'$.
- $\langle A \wedge A' \rangle_{\sigma,k+1} \triangleq \langle A' \wedge A'' \rangle_\sigma$, *where* $A \rightsquigarrow_k A'$ *and* $A \rightsquigarrow_{k+1} A''$
- $\langle A \rightarrow A' \rangle_{\sigma,k+1} \triangleq \langle A \rightarrow A' \rangle_{\sigma,k} \wedge \langle A'' \rightarrow A''' \rangle_\sigma$, *where* $A \rightsquigarrow_k A''$ *and* $A' \rightsquigarrow_{k+1} A'''$

As well, we define

- `M`$, \sigma \not\models_k A$ *iff* $\langle A' \rangle_{\sigma,j} \neq$ `true`.

And also

- `M`$, \sigma \models A$ *iff* $\exists k. \forall j \geq k. \langle A \rangle_{\sigma,j} =$ `true`.

5.1 Discussion

EXAMPLE 21 (ACYCLIC AND FINITE). *In the acyclic configuration* σ_{acyc}*, we can establish that* `x` *is finite after three steps. That is*

(A)	`Finite(x)` $\rightsquigarrow_2$ `x.next` $\neq$ `null` $\rightarrow$ (`x.next.next` $\neq$ `null` $\rightarrow$ `Finite(x.next.next)`	*by def 18, for all* $k \geq 2$
(B)	$\langle$`Finite(x)`$\rangle_{\mathrm{M},\sigma_{acyc},1} =$ `b???`	*by def 18, and def 20.*
(C)	$\langle$`Finite(x)`$\rangle_{\mathrm{M},\sigma_{acyc},2} =$ `true`	*by (A) and def 20.*
(D)	$\langle$`Finite(x)`$\rangle_{\mathrm{M},\sigma_{acyc},k} =$ `true`	*for all* $k \geq 2$*, by (A) and def 20.*
(E)	`M`$, \sigma_{acyc} \models$ `Finite(x)`,	*by (D) and def 20.*
...	...	

Similarly, we have

[11] Notice that we have the empty module $\emptyset$, and so cannot unfold functions in this judgment.

(A)	$\texttt{Finite}'(\texttt{x}) \rightsquigarrow_k$ $\texttt{x.next} = \texttt{null} \vee$ $(\texttt{x.next.next} = \texttt{null} \vee \texttt{Finite}'(\texttt{x.next.next}))$	*by def 18, base case, if* $k \geq 2$
(B)	$\langle \texttt{Finite}'(\texttt{x}) \rangle_{\texttt{M},\sigma_{acyc},1} = \texttt{b???}$	*by def 18, and def 20.*
(C)	$\langle \texttt{Finite}'(\texttt{x}) \rangle_{\texttt{M},\sigma_{acyc},2} = \texttt{true}$	*by (A) and def 20.*
(D)	$\langle \texttt{Finite}'(\texttt{x}) \rangle_{\texttt{M},\sigma_{acyc},k} = \texttt{true}$	*for all* $k \geq 2$*, by (A) and def 20.*
(E)	$\texttt{M}, \sigma_{acyc} \models \texttt{Finite(x)},$	*by (D) and def 20.*
...	...	

A APPENDIX

A.1 Continuing the example

EXAMPLE 22 (CONTINUED). *The rest needs to be revisited and adapted.*

In the cyclic configuration, as we saw earlier, $\lfloor \texttt{acyclic(z)} \rfloor_{\sigma_{cyc}}$ *is undefined. Nevertheless, we* $\texttt{M}, \sigma_{cyc} \models_k \neg \texttt{Finite}(z)$ *for all k. We also have that* $\texttt{M}, \sigma_{cyc} \models_k \neg \texttt{length}(z) = 3$*, even though* `length` *and also* $\texttt{length}(z) = 3$ *are undefined in the context of* σ_{cyc}*. This gives us also that (*)* $\texttt{M}, \sigma \models \neg \texttt{Finite}(z)$*, and also* $\texttt{M}, \sigma_{cyc} \models_k \neg \texttt{length}(z) = 3$*.*

EXAMPLE 23 (CNTRDR). *We have* $\texttt{M}, \sigma \not\models \texttt{Cntrdct(z)}$ *as well as* $\texttt{M}, \sigma \not\models \neg \texttt{Cntrdct(z)}$*.*

OBSERVATION 24 (MONOTONICITY). *I think now we have that* $\texttt{M}, \sigma \models_k A$ *implies* $\texttt{M}, \sigma \not\models_{k+j} A$*. To think more.*

OBSERVATION 25 (LACK OF MODUS PONENS). *We do not have modus ponens. A counter-example appears in (*).*

OBSERVATION 26 (DISTRIBUTIVITY OF DISJUNCTION). *In our system, disjunction distributes, ie we have*

LEMMA 27 (DISJUNCTION IS DISTRIBUTIVE). *For all A.*

- $\texttt{M}, \sigma \models_k A \vee A'$ *implies* $\texttt{M}, \sigma \models_k A$ *or* $\texttt{M}, \sigma \models_k A'$ *for all k.*
- $\texttt{M}, \sigma \models A \vee A'$ *implies* $\texttt{M}, \sigma \models A$ *or* $\texttt{M}, \sigma \models A'$

TODO check!

CONJECTURE 28 (CONSISTENCY). *For all* M*, A, A′ and σ.*

- $\texttt{M}, \sigma \models A$ *ana* $\texttt{M}, \sigma \models A$ *never holds.*
- $\texttt{M}, \sigma \models A \rightarrow A'$ *and* $\texttt{M}, \sigma \models A$ *entails* $\texttt{M}, \sigma \models A'$
- $\texttt{M}, \sigma \models \neg A \vee A'$ *entails* $\texttt{M}, \sigma \models A \rightarrow A'$
- $\texttt{M}, \sigma \models A \vee A'$ *implies* $\texttt{M}, \sigma \models A$ *or* $\texttt{M}, \sigma \models A'$

TODO check!

A.2 Considering holistic aspects

In Definition 6 we have not considered the holistic aspects of assertions. We will therefore extend the definition as below

DEFINITION 29. *Given a module* M*, a state σ, and a natural number k, we define the judgment* $\texttt{M}, \sigma \models_k A$ *by cases on A and k as follows, where* $k > 0$ *and arbitrary:*

- ***x obeys*** $\texttt{S} \rightsquigarrow_{k+1} A'$ *iff for some A from* S*,* $A[\texttt{this} \mapsto \texttt{x}] \rightsquigarrow_k A'$*.*
- $\sigma \models_{naive}$ ***x obeys*** S *is true.*
- $\texttt{M}, \sigma \models A$ *iff* $\exists k. \forall j > k. \forall A'. A \rightsquigarrow_j A' \longrightarrow \texttt{M}, \sigma \models_{naive} A'$*.*

More thoughts needed here. Notice that in the non-holistic part we asked something weaker, ie we had that $\exists k.\forall j > k.\exists A'.A \rightsquigarrow_j A' \ \wedge \ \mathtt{M}, \sigma \models_{naive} A$.