Veriopt Theories

April 17, 2024

Contents

theory *SemanticsSnippets*
  imports
    Semantics.IRStepObj Semantics.Form Proofs.Stuttering Snippets.Snipping
begin

declare [[*show-types=false*]]

notation (*latex*)
  kind (-⟪-⟫)

notation (*latex*)
  stamp-expr (⇑ -)

notation (*latex*)
  val-to-bool (-$_{bool}$)

syntax (*spaced-type-def* **output**)
  -constrain :: *logic* => *type* => *logic* (- :: - [*4*, *0*] *3*)

is-BinaryArithmeticNode :: *IRNode* ⇒ *bool*

inputs-of :: *IRNode* ⇒ *nat list*
inputs-of (*ConstantNode const*) = []
inputs-of (*ParameterNode index*) = []
inputs-of (*ValuePhiNode nid0.0 values merge*) = *merge* · *values*
inputs-of (*AddNode x y*) = [*x*, *y*]
inputs-of (*IfNode condition trueSuccessor falseSuccessor*) = [*condition*]

typedef *IRGraph* = {*g* :: *ID* ⇀ *IRNode* . *finite* (*dom g*)}

fun *ids-fake* :: $(ID \rightharpoonup IRNode) \Rightarrow ID\ set$ **where**
  $ids\text{-}fake\ g = \{nid \in dom\ g\ .\ g\ nid \neq (Some\ NoNode)\}$

fun *kind-fake* :: $(ID \rightharpoonup IRNode) \Rightarrow (ID \Rightarrow IRNode)$ **where**
  $kind\text{-}fake\ g = (\lambda nid.\ (case\ g\ nid\ of\ None \Rightarrow NoNode \mid Some\ v \Rightarrow v))$

$ids\text{-}fake :: (nat \Rightarrow IRNode\ option) \Rightarrow nat\ set$
$ids\text{-}fake\ g = \{nid \in dom\ g \mid g\ nid \neq Some\ NoNode\}$

$kind\text{-}fake :: (nat \Rightarrow IRNode\ option) \Rightarrow nat \Rightarrow IRNode$
$kind\text{-}fake\ g = (\lambda nid.\ \mathsf{case}\ g\ nid\ \mathsf{of}\ None \Rightarrow NoNode \mid Some\ v \Rightarrow v)$

$inputs :: IRGraph \Rightarrow nat \Rightarrow nat\ set$
$inputs\ g\ nid = set\ (inputs\text{-}of\ g\langle\!\langle nid \rangle\!\rangle)$

$succ :: IRGraph \Rightarrow nat \Rightarrow nat\ set$
$succ\ g\ nid = set\ (successors\text{-}of\ g\langle\!\langle nid \rangle\!\rangle)$

$input\text{-}edges :: IRGraph \Rightarrow (nat \times nat)\ set$
$input\text{-}edges\ g = (\bigcup_{i \in ids\ g} \{(i, j) \mid j \in inputs\ g\ i\})$

$usages :: IRGraph \Rightarrow nat \Rightarrow nat\ set$
$usages\ g\ nid = \{i \in ids\ g \mid nid \in inputs\ g\ i\}$

$successor\text{-}edges :: IRGraph \Rightarrow (nat \times nat)\ set$
$successor\text{-}edges\ g = (\bigcup_{i \in ids\ g} \{(i, j) \mid j \in succ\ g\ i\})$

$predecessors :: IRGraph \Rightarrow nat \Rightarrow nat\ set$
$predecessors\ g\ nid = \{i \in ids\ g \mid nid \in succ\ g\ i\}$

$wf\text{-}start\ g =$
$(0 \in ids\ g \wedge is\text{-}StartNode\ g\langle\!\langle 0 \rangle\!\rangle)$

$wf\text{-}closed\ g =$
$(\forall\, n \in ids\ g.$
  $inputs\ g\ n \subseteq ids\ g \wedge$
  $succ\ g\ n \subseteq ids\ g \wedge g\langle\!\langle n \rangle\!\rangle \neq NoNode)$

```
wf-phis g =
(∀ n∈ids g.
    is-PhiNode g⟪n⟫ ⟶
    |ir-values g⟪n⟫| =
    |ir-ends g⟪ir-merge g⟪n⟫⟫|)
```

```
wf-ends g =
(∀ n∈ids g.
    is-AbstractEndNode g⟪n⟫ ⟶
    0 < |usages g n|)
```

```
wf-graph :: IRGraph ⇒ bool
wf-graph g = (wf-start g ∧ wf-closed g ∧ wf-phis g ∧ wf-ends g)
```

type-synonym *Signature = string*

type-synonym *Program = Signature ⇀ IRGraph*

print-antiquotations

```
type-synonym Heap = string ⇒ objref ⇒ Value
type-synonym Free = nat
type-synonym DynamicHeap = Heap × Free


h-load-field :: string ⇒ objref ⇒ DynamicHeap ⇒ Value
h-load-field f r (h, n) = h f r


h-store-field :: string ⇒ objref ⇒ Value ⇒ DynamicHeap ⇒ DynamicHeap
h-store-field f r v (h, n) = (h(f := (h f)(r := v)), n)


h-new-inst :: DynamicHeap ⇒ (DynamicHeap × Value)
h-new-inst (h, n) className = (h-store-field ''class'' (Some n) (ObjStr
className) (h, n + 1), ObjRef (Some n))
```

step:seq step:if step:end step:newinst step:load step:store
step:load-static step:store-static

top:lift top:invoke top:return top:return-void top:unwind

$$\frac{g,\ p \vdash (nid,\ m,\ h) \rightarrow (nid',\ m,\ h)}{g\ m\ p\ h \vdash nid \rightsquigarrow nid'}$$

$$\frac{g,\ p \vdash (nid,\ m,\ h) \rightarrow (nid'',\ m,\ h) \qquad g\ m\ p\ h \vdash nid'' \rightsquigarrow nid'}{g\ m\ p\ h \vdash nid \rightsquigarrow nid'}$$

notation (*latex* **output**)
  filtered-inputs (*inputs*$^{\text{-}\langle\!\langle\text{-}\rangle\!\rangle}$ _)
notation (*latex* **output**)
  filtered-successors (*succ*$^{\text{-}\langle\!\langle\text{-}\rangle\!\rangle}$ _)
notation (*latex* **output**)
  filtered-usages (*usages*$^{\text{-}\langle\!\langle\text{-}\rangle\!\rangle}$ _)

$$inputs^{g\langle\!\langle nid\rangle\!\rangle}{}_{f}$$

notation (*latex* **output**)
  Pure.dummy-pattern (−)

notation (*latex* **output**)
  IntVal (*IntVal* (*2* -))

end