1 Verification Condition Generator

theory *VCG = AbsSem*:For technical reasons Isabelle's locales do not (yet?) support recursive definitions. Hence, we define the VCG outside and refer to this defintion inside the locale VCG, which we define below. This external definition looks quite ugly as a rat tail of parameters is in the signature. The definition as it shoul be is derived as lemma **vcgdef** below.

1.1 Inductive Safety Formulas

consts

isafe:: (*'pos list* × *'program* × (*'pos* ⇒ *'form option*) × *'pos* × *'form* × (*'form list* ⇒ *'form*) × (*'form* ⇒ *'form* ⇒ *'form*) × (*'program* ⇒ *'pos* ⇒ *'form*) × (*'program* ⇒ *'pos* ⇒ (*'pos* × *'form*) *list*) × (*'program* ⇒ *'pos* ⇒ *'pos* ⇒ *'form* ⇒ *'form*)) ⇒ *'form*

recdef

isafe measure (λ *a*. *length* (*fst a*))
isafe (*S*,*prg*,*anF*,*pc*,*FalseF'*,*Conj'*,*Impl'*,*appF'*,*succsF'*,*wpF'*) =
(*if* ¬ (*pc mem S*)
then FalseF'
else Conj' ([*appF' prg pc*] @
(*case* (*anF pc*) *of*
None ⇒ (*map* (λ (*pc'*,*B*). *Impl' B*
(*wpF' prg pc pc'* (*isafe* ((*filter* (λ *a*. *a* ≠ *pc*) *S*),
prg,
anF,
pc',
FalseF',
Conj',
Impl',
appF',
succsF',
wpF')
)
)
)
(*succsF' prg pc*))
| *Some a* ⇒ [*a*])))
(**hints** *recdef-simp*: *filterreduction*)

1.2 External VCG

The vcG functions takes the auxiliary functions from the framework as additional arguments**consts**

vcG:: (*'form list* ⇒ *'form*) ⇒ (*'form* ⇒ *'form* ⇒ *'form*) ⇒ *'form* ⇒ (*'program* ⇒ *'pos*) ⇒ (*'program* ⇒ *'form*) ⇒ (*'program* ⇒ *'pos* ⇒ *'form*) ⇒ (*'program* ⇒ *'pos* ⇒ (*'pos* × *'form*) *list*) ⇒ (*'program* ⇒ *'pos* ⇒ *'pos* ⇒ *'form* ⇒ *'form*) ⇒ (*'program* ⇒ *'pos list*) ⇒ (*'program* ⇒ *'pos list*) ⇒ (*'program* ⇒ (*'pos* ⇒ *'form option*)) ⇒ *'program* ⇒ *'form*

defs *vcG-def*:
vcG Conj Impl FalseF ipc initF safeF succsF wpF domC domA an p ≡
(*let an′* = (λ *i*. *isafe*(*domC p*,*p*,*an p*,*i*,*FalseF*,*Conj*,*Impl*,*safeF*,*succsF*,*wpF*))
in (*Conj* ((*Impl* (*initF p*) (*an′* (*ipc p*)))#(*map* (λ *i*. *Conj* (*map* (λ (*j*,*B*). *Impl* (*Conj* [(*an′ i*), *B*])
((*wpF p i j*) (*an′ j*)))
(*succsF p i*)))
(*domA p*)))))

1.3 Pre Safe States

Here we define the set of pre safe states. These are all initial states and states that are reachable from those by only traversing inductively safe intermediate states

1.4 Generic Verification Conditons

locale *VCG* = *AbsSem* +

fixes *vcg*:: *'prog* ⇒ *'form*
defines *vcg* ≡ λ Π. *vcG Conj Impl FalseF ipc initF safeF succsF wpF domC domA anF* Π

fixes *isafe′*::(*'pos list* × *'prog* × *'pos*) ⇒ *'form*
defines *isafe′* ≡ λ (*S*,Π,*pc*). *isafe*(*S*,Π,(*anF* Π),*pc*,*FalseF*,*Conj*,*Impl*,*safeF*,*succsF*,*wpF*)
fixes *isafeF*:: *'prog* ⇒ *'pos* ⇒ *'form*
defines *isafeF* ≡ (λ Π *pc*. *isafe′* (*domC* Π,Π,*pc*))

This lemma shows that vcg could be defined as follows, if the locale mechanism allowed us recursive definitions**lemma** (**in** *VCG*) *vcgdef*:
⟦ *wf* Π ⟧ ⟹
vcg Π = *Conj* ((*Impl* (*initF* Π) (*isafeF* Π (*ipc* Π)))#(*map* (λ *pc*. *Conj* (*map* (λ (*pc′*,*B*). *Impl* (*Conj* [(*isafeF* Π *pc*), *B*])
((*wpF* Π *pc pc′*) (*isafeF* Π *pc′*)))
(*succsF* Π *pc*)))
(*domA* Π)))

end