12-07-19

EX1

Alphabet:

C(x), BC(x), S(x), P(x), agreement(x,y), discount(x,y,z), contract(x,y,z)

Axioms:

Forall x. BC(x) implies C(x) \\ ISA

Forall x,y,z. contract(x,y,z) implies C(x) and P(y) and S(z) \\ typing

Forall x,y,z,z’. contract(x,y,z) and contract(x,y,z’) implies z=z’ \\ key

Forall x,y. specializedin(x,y) implies P(x) and S(y) \\ typing

Forall x. P(x) implies 1<=#{y|specializedin(x,y) } \\ multiplicity

Forall x,y. agreement(x,y) implies BC(x) and P(y) \\ typing

Forall x. BC(x) implies #{y|agreement(x,y)}<=1 \\ multiplicity

Forall x,y,z. discount(x,y,z) implies agreement(x,y) \\ typing

Forall x,y. agreement(x,y) implies 1<=#{z|discount(x,y,z)}<=1 \\ multiplicity

EX2

1. The instantiation is not complete because of presence of ISA, we need to apply the following procedure to obtain a complete instantiation given a UML diagram T

Iold=0, Inew=I

While(Inew and Iold are different) do

For each (forall x. A(x) implies B(x) ) do

For each a in A^Inew do

B^Inew = B^Inew union {a}

Similar for each subset constrains forall x,y. P(x,y) implies R(x,y)

I=Inew

Return I

I = (Obj^I, BC^I, C^I, S^I, P^I, agreement^I, contract^I, specializedin^I, agreement^I, discount^I)

I0:

BC^I = {b1,b2}

C^I = {c1,c2}

S^I = {s1,s2,s3,s4}

P^I = {p1,p2}

agreement^I = {}

contract^I = { (c1,p1,s1), (c1,p2,s2), (c2,p1,s1), (b1,p1,s4), (b2,p2,s5) }

specializedin^I ={ (p1,s1), (p1,s2), (p1,s3), (p2,s4), (p2,s5) }

agreement^I/discount^I ={(b1,p1,30)}

I1:

BC^I = {b1,b2}

C^I = {c1,c2, b1,b2}

S^I = {s1,s2,s3,s4}

P^I = {p1,p2}

agreement^I = {}

contract^I = { (c1,p1,s1), (c1,p2,s2), (c2,p1,s1), (b1,p1,s4), (b2,p2,s5) }

specializedin^I ={ (p1,s1), (p1,s2), (p1,s3), (p2,s4), (p2,s5) }

agreement^I/discount^I ={(b1,p1,30)}

I2 = I1 the instantiation is complete

Now we need to check if it is correct, so I2 |= T, this means that we want to verify if all axioms in T are evaluate true in this interpretation I2.

The instantiation is correct.

1. q(x) <- P(x) and Exists y,y’. S(y) and S(y’) and specializein(x,y) and specializein(x,y’) and y noteq y’

q(x) : {p1,p2}

1. q(x) <- BC(x) and forall y. contract(x,y) an

EX3

Model checking a closed mu calculus formula phi over T=<S,Ra,Pi> (S set of states, Ra set of transitions and Pi a mapping function from a set of proposition P to a subset of S) means that we want to check if the initial state of T is in the extension of phi over T. With model checking we return a subset of S in which each element satisfies phi. To compute it we apply the labelling algorithm, that consist in labelling the states of T with predicates that are true in them. The extensions of least fixpoint and greatest fixpoint are computed using Tarski-Knaster approximates theorem.

νX.muY.((a and <next> X) or ([next] not b and <next> Y)

We are going to find the greatest fixpoint GFP because of presence vX

[X0] = {1,2,3,4}

[X1] = [muY.((a and <next> X0) or ([next] not b and <next> Y)]

We are going to find the least fixpoint LFP because of presence muY

[Y00] = {}

[Y01] = [(a and <next> X0) or ([next] not b and <next> Y00)] = ([a] intersect PreE(next,[X0])) union (PreA(next,[not b]) intersect PreE(next,[Y00])) = ({2} intersection {1,2,3,4}) union (PreA(next,{1,2,3}) intersect PreE(next,[Y00])) = ({2} intersection {1,2,3,4}) union ({1,2,4} intersect {} ) = {2}

[Y02] =[(a and <next> X0) or ([next] not b and <next> Y01)] = ([a] intersect PreE(next,[X0])) union (PreA(next,[not b]) intersect PreE(next,[Y01])) = ({2} intersection {1,2,3,4}) union (PreA(next,{1,2,3}) intersect PreE(next,[Y01])) = ({2} intersection {1,2,3,4}) union ({1,2,4} intersect {1} ) = {1,2}

[Y03] =[(a and <next> X0) or ([next] not b and <next> Y02)] = ([a] intersect PreE(next,[X0])) union (PreA(next,[not b]) intersect PreE(next,[Y02])) = ({2} intersection {1,2,3,4}) union (PreA(next,{1,2,3}) intersect PreE(next,[Y02])) = ({2} intersection {1,2,3,4}) union ({1,2,4} intersect {1,3,4} ) = {1,2,3,4}

[Y04] =[(a and <next> X0) or ([next] not b and <next> Y03)] = ([a] intersect PreE(next,[X0])) union (PreA(next,[not b]) intersect PreE(next,[Y03])) = ({2} intersection {1,2,3,4}) union (PreA(next,{1,2,3}) intersect PreE(next,[Y03])) = ({2} intersection {1,2,3,4}) union ({1,2,4} intersect {1,2,3,4} ) = {1,2,3,4}

[Y04] = [Y03] - - > found LFP

[X1] = {1,2,3,4}

[X1] = [X2] - - > found GFP

Now we are going to do model checking with CTL formula. Given a Kripke model (KM) = <S,I,R,AP,L> (S is set of states, I set of initial state, R se of transitions, AP set of atomic proposition and L labelling function) and a CTL formula phi, we want to check if KM,s |= phi where s is state of S. With model checking we return a subset of S in which each element satisfied phi, and to compute it we need to exploit the syntactic structure of CTL formula, in particular, we translate each sub formula of CTL into mu calculus formula and then apply the labelling algorithm to find their extensions.

EG(AFa and(EFb or AG not b))

alpha = AG not b = vX not b and [next] X

beta = EF b = muY b or <next> X

gamma = beta or alpha

delta = AF a = mu X a or [next] X

sigma = delta and gamma

tau = EG sigma

[alpha]= [vX not b and [next] X]

We are going to find the greatest fixpoint GFP because of the presence of vX

[X0] = {1,2,3,4}

[X1] = [not b and [next] X0] = [not b] intersect PreA(next,[X0]) = {1,2,3} intersect {1,2,3,4} = {1,2,3}

[X2] = [not b and [next] X1] = [not b] intersect PreA(next,[X1]) = {1,2,3} intersect {1,2,4} = {1,2}

[X3] = [not b and [next] X2] = [not b] intersect PreA(next,[X2]) = {1,2,3} intersect {1,4} = {1}

[X4] =[not b and [next] X3] = [not b] intersect PreA(next,[X3]) = {1,2,3} intersect {4} = {}

[X5] =[not b and [next] X4] = [not b] intersect PreA(next,[X4]) = {1,2,3} intersect {4} = {}

[X5] = [X4]- - > found GFP

[alpha] = {}

[beta] = [muY b or <next> X]

We are going to find the least fixpoint LFP because of the presence of muX

[X0] = {}

[X1] = [b or <next> X0] = [b] union PreE(next,[X0]) = {4} union {} = {4}

[X2] = [b or <next> X1] = [b] union PreE(next,[X1]) = {4} union {3} = {3,4}

[X3] = [b or <next> X2] = [b] union PreE(next,[X2]) = {4} union {2,3} = {2,3,4}

[X4] = [b or <next> X3] = [b] union PreE(next,[X3]) = {4} union {1,2,3} = {1,2,3,4}

[X5] = [b or <next> X4] = [b] union PreE(next,[X4]) = {4} union {1,2,3,4} = {1,2,3,4}

[X5] = [X4] - - > found LFP

[beta] = {1,2,3,4}

[gamma] = [beta or alpha] = [beta] union [alpha] = {1,2,3,4} union{} = {1,2,3,4}

[delta] =[ mu X a or [next] X]

We are going to find the least fixpoint LFP because of the presence of muX

[X0] = {}

[X1] = [a or [next] X0]] = [a] union PreA(next,[X0]) = {2} union {} = {2}

[X2] = [a or [next] X1]] = [a] union PreA(next,[X1]) = {2} union {1} = {1,2}

[X3] = [a or [next] X2]] = [a] union PreA(next,[X2]) = {2} union {1,4} = {1,2,4}

[X4] = [a or [next] X3]] = [a] union PreA(next,[X3]) = {2} union {1,3,4} = {1,2,3,4}

[X5] = [a or [next] X4]] = [a] union PreA(next,[X3]) = {2} union {1,2,3,4} = {1,2,3,4}

[X5] = [X4] - -> found LFP

[delta] = {1,2,3,4}

[sigma] = [delta and gamma] = [delta] intersect [gamma] = {1,2,3,4} intersect {1,2,3,4} = {1,2,3,4}

[tau] = [EG sigma] = [vX sigma and <next> X]

We are going to find the greatest fixpoint GFP because of the presence of vX

[X0] = {1,2,3,4}

[X1] = [sigma and <next>X0]= [sigma] intersect PreE(next,[X0]) = {1,2,3,4} intersect {1,2,3,4} = {1,2,3,4}

[X1] = [X0] - - > found GFP

[tau] = {1,2,3,4}

It is 1 in [tau] ? YES, so CTL formula is true in this transition system

EX4

Check if the Hoare triple {P} while g do S {Q} is correct cannot be done in automatic way but we need inference rule.

Assuming to have a candidate invariant I, we show:

* P implies I
* {I and g} S {I} = {I and g} implies wp(S,I)
* {I and not g} implies Q

If I satisfied these 3 operation it is an invariant.

If we know the candidate invariant these operation are automatic, otherwise it is difficult to manage with it. The problem of check if the Hoare triple is true or not is undecidable because there is no sound or complete technique to do this or to find all possible invariants. The only thing that we can do is to check if candidate is invariant or not.

Candidate invariant: {x>=0 and y=> 0 and x+y = 23}

{x= 23 and y= 0}while(x>0) do (x=x−1; y:= y+1){y= 23}

* P implies I

{x = 23 and y = 0 } implies {x>=0 and y=> 0 and x+y = 23}

OK, because if x=23 and y =0 then x is major or equal than 0 and also y and x+y = 23

* {I and not g} implies Q

{x>=0 and y=> 0 and x+y = 23 and x<=0} implies {y=23}

{x=0 and y>=0 and x+y = 23} implies {y=23}

OK

* wp(S,I)

{x>=1 and y>=-1 and x+y = 23} is wp

x=x-1

{x>=0 and y>=-1 and x+y+1 = 23}

y=y+1

{x>=0 and y>=0 and x+y = 23}

{I and g} implies wp(S,I)

{x>=0 and y>=0 and x+y = 23 and x>0} implies {x>=1 and y>=-1 and x+y = 23}

{y>=0 and x+y = 23 and x>0} implies {x>=1 and y>=-1 and x+y = 23}

{x>0 and y>=0 } implies {x>=1 and y>=-1}

the condition is satisfied because it always true

{x>=0 and y>=0 and x+y = 23} is invariant

EX5

Check if formula phi is valid means that we want to verify that forall interpretation I, I|=phi. Tableaux method is used for proving if a formula is satisfiable or not. This means that for validity we mush prove that if I|=phi, for all possible interpretation and this is a NP-complete problem.

Check if formula is satisfiable we need to find an interpretation I such that I|=phi. To test formula for validity we need to transform our problem into a satisfiable problem by negating the formula phi and check if it is satisfiable.

A formula is satisfiable if exists at least one open branch, but in our case of validity we star with not phi so if we found only closed branches our formula is valid

Immagine che contiene testo

Descrizione generata automaticamente

EX6

To do model checking with LTL formula we cannot transform it into mu calculus as CTL. We cannot even exploit in NFA or DFA properties because they are infinite and work with infinite states, while LTL is evaluated on infinite languages and has infinite traces. We can instead using NBA, because they accept to go to a finite state infinite often. To model check LTL formula over transition system T we need to prove L(T) subseteq L(phi) iff L(T) intersection L(not phi) = {}. If we translate T in automata At and notphi in Anotphi we check the nonemptiness so we have that L(At and Anotphi) = {}. In this way we can prove that phi is satisfied by T, so check if the new automata accept at least a word( a trace of transition system that start from initial state and go to the final and loops there) To prove it, we check that if At and Anotphi, init |=sigma = vXmuY(final and <next>X ) or <next>Y)

Immagine che contiene testo

Descrizione generata automaticamente

vXmuY(final and <next>X ) or <next>Y)

We are going to find the greatest fixpoint GFP because of presence of vX

[X0] = {(init,i),(0,i),(1,i),(2,i),(1,ii),(2,ii)}

[X1] = [muY (final and <next> X0) or <next>Y]

We are going to find the least fixpoint LFP because of presence of muY

[Y00] = {}

[Y01] = [(final and <next>X0) or <next>Y00] = [final] intersect PreE(next,[X0])) or PreE(next,[Y00]) = ({(1,ii),(2,ii)} intersect {(init,i),(0,i),(1,i),(2,i),(1,ii),(2,ii)}) union {} = {(1,ii),(2,ii)}

[Y02] = [(final and <next>X0) or <next>Y01] = [final] intersect PreE(next,[X0])) or PreE(next,[Y01]) = ({(1,ii),(2,ii)} intersect {(init,i),(0,i),(1,i),(2,i),(1,ii),(2,ii)}) union {(0,i),(1,i),(2,i),(2,ii)} = {(0,i),(1,i),(2,i),(2,ii)}

[Y03] = [(final and <next>X0) or <next>Y02] = [final] intersect PreE(next,[X0])) or PreE(next,[Y02]) = ({(1,ii),(2,ii)} intersect {(init,i),(0,i),(1,i),(2,i),(1,ii),(2,ii)}) union {(init,i),(0,i),(1,i),(2,i),(2,ii)} = {(init,i),(0,i),(1,i),(2,i),(2,ii)}

[Y04] = [(final and <next>X0) or <next>Y03] = [final] intersect PreE(next,[X0])) or PreE(next,[Y03]) = ({(1,ii),(2,ii)} intersect {(init,i),(0,i),(1,i),(2,i),(1,ii),(2,ii)}) union {(init,i),(0,i),(1,i),(2,i),(2,ii)} = {(init,i),(0,i),(1,i),(2,i),(2,ii)}

[Y04] = [Y03] - - > found LFP

[X1] ={(init,i),(0,i),(1,i),(2,i),(2,ii)}

[X1] = [X0] - - > found GFP

[sigma] = {(init,i),(0,i),(1,i),(2,i),(2,ii)}

It is (init,i) in [sigma]? YES, so the LTL formula is satisfied by this transition system