Limitation-II-Dependency VS Weight

Friday, June 24, 2022 12:36 PM

$E = \{ x^i, \omega, y^j) \mid x^i, y^j \in LVar$
$\omega = max \{ \#_\tau (\tau) \mid \forall c. Dep(x^i, y^j, \tau, c) \} \}$

example: input x:

```
y=0;
i=0;
w=0;
z <- q1;
if x = 0
{ y <- q2(z); i=x-1}
else
{skip};
while i \neq x
{w <- q3(y,w); if i neq x-2 then w=0 else skip ; i=i+1}
```

a trace τ_1. $\xi_1 \overset{=}{\in} \omega, 2$. $\quad \xi_2 \overset{=}{\in} \omega, 2$.
change $\xi_1 = \xi_1' \Rightarrow \xi_2' = \xi_1', 2$

then another trace: $\xi_1 = \omega, 2$, $\quad \xi_2 = \omega, 2$, $\quad \xi_3 = \omega, 2$.
change ↓ $\quad$ ‖ $\quad$ ↓change
$\xi_1' = \omega, 2, v'$. $\quad \xi_2 = \omega, 2$. $\quad \xi_3' = \omega^2, 2, v_3'$.

$\forall$ trace. s.t. ξ_1, τ, ξ
$Dep(\omega^2, \tau, \omega^2)$, # of occurrence time of ω ≥ # of event where value of ω^2 is changed.

then. count: $\forall$ trace τ s.t. $Dep(\omega^2, \tau, \omega^2)$.
$|Diff(\tau_1, \tau_2, \omega^2)|$

⇒ weight on edge (x^1, y^j, ω).
$\omega = max \{ |Diff(\tau_1, \tau_2, y^j)| \mid \forall \tau_1, \tau_2$. s.t. $Dep(x^i, \tau_1, \tau_2, y^j) \}$.

⇒ 1. Dep on 2 traces. redefine.

⇒ 2. Control Dep. ??
⇒ add iteration id.
in order to know the value changed in the 2 traces are in the same iteration.
then,

vs possible unsound ??
if cases $\forall \tau$. observe the same value, but in different iterations ? ⇒
then intuitively no-dependency. it is good.

⇒3. $Diff(\tau_1, \tau_2, x^i)$
⇒ ~~Hard~~

$$seq(\tau_1)\, x^i \triangleq \begin{cases} \tau_1 = \tau_1' :: (x, i, \nu, \cdot) \to seq(\xi :: \tau_1') :: \nu \\ \tau_1 = \tau_1' :: (x, i, \nu, \alpha) \to seq(\tau_1') :: \alpha. \\ \tau_1 = [] \to [] \\ o.w. \to seq(\tau_1') \end{cases}$$

without loop iteration id.
for $seq(\tau_2)\, x^i$. where $Dep(y^j, x^i, \tau_1, \tau_2)$.
cases of $seq(\tau_1) \neq seq(\tau_2)$:
① $len_1 \neq len_2$. ⇒ $Diff(seq(\tau_1), seq(\tau_2), x^i) = \{k \mid seq(\tau_1')[k] \neq seq(\tau_2)[k]\} \cup \{len\tau_1, len_1+1, \ldots len_2\}$.
② $len_1 = len_2$ ⇒ $Diff(seq(\tau_1), seq(\tau_2), x^i) = \{k \mid seq(\tau_1)[k] \neq seq(\tau_2)[k]\}$
⇒ potential issue: the different observed value is in different iterations.
case: if query variable is affected by moving to different iteration, but it always executed the same time with same value.
then it doesn't matter which iteration it is in, there left dependence / adaptivity.
⇒ for case ②. where they have same len, it doesn't loss adaptivity if I observe the same value in the same location of $seq(\tau_1)$ and $seq(\tau_2)$
to sound where they actually comes from different loop iteration.
i.e. it is sound that only concern the diff of some location.

⇒. for case ①. $len_1 \neq len_2$.
in the 2 seqs from $[0, min(l_1, l_2)]$, it is still sound by just observe the different value in same location,
for the same reason as case ②.

in the longer seq from $[min(l_1, l_2), max(l_1, l_2)]$.
case $i = 0$
while $i < x$ do:
$i = i + 1$
$y = f(a)$.
⇒ when $x = 100$
still safe.
$(y, 1, v_1)$ $(y, 1, v_2)$ $(y, 1, v_3)$ $(y, 2, 0 \ldots)$
$(y, 1, v_1')$ $(y, 1, v_2')$ $(y, 1, v_3')$ $(y, 2, 0) \ldots$
⇒ $Gy^*(inf, 3)$.

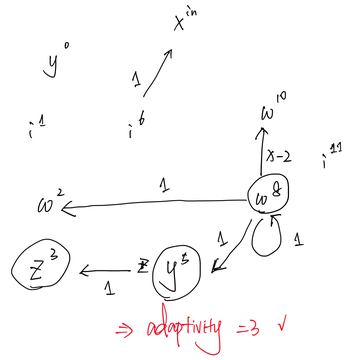
in order to get different length, only way is by changing value affect guards.
if in the longer seq from $min(l_1, l_2)$ to $max(l_1, l_2)$.
$(x, in, 1)$ $(y, 1, v_1)$
$(x, in, 100)$, $(y, 1, v_1)$, $(y, 1, v_2)$ ...
⇒ all the value are the same. then adaptivity is actually 1.

⇒ if they are different, then either (guard variable ω is used in evaluating y. / or y itself is used (through a chain) is used in evaluating y.) — 2 approaches:

$i = 0$
while $i < x$.
$\omega = f(\omega)$
if $i < 3$, $\omega = 0$, skip
$i = i + 1$.
$x = 2$ $(\omega, 0)$, $(\omega, 1, 0)$
$x = 100$. $(\omega, 1, 0)$ $(\omega, 1, 0)$ $(\omega, 1, f(1))$ $(\omega, 1, f(f(1)))$ ....
Dep: infinite.

case: $i = 0$
while $i < x$:
$\omega = f(\omega)$
if $i >= 3$, $\omega = 0$, skip. $x = 100$: $(\omega, 1, f(1))$, $(\omega, 1, f(3))$ $(\omega, 1, f(3))$, $(\omega, 1, 0)$ $(\omega, 1, 0)$ ...
$i = i + 1$

two approaches:
① prove there must exist value dependency from x to y (unclear to prove, because change x could change length and value in the same time.) or from other variable to y.
② define $|Diff|$ as $\{ seq(\tau_{max})[min : max] \}$. ⇝ sound. set of unique values.
possible case. $(1, 0, 1, 0, 1, 0)$ I actually have no intuition does this adaptive or not.
⇒ if it is. the $|Diff|$ defined as $\{ k \mid seq[k-1] \neq seq[k] \}$ can capture.
but "intuitive adaptivity" is unclear.



⇒ adaptivity =3 ✓

⇒ $Dep(w^8, \tau_1, \tau_2, w^8, c)$, with $max |Diff(\tau_1, \tau_2, w^8)|$
$\forall x \in \mathbb{N}$.
$x = 4$
$\tau_1 = [(w^8, f_3(0,0), V_1), (w^{10}, 0), (i, 1),$
$(w^8, f_3(0,0), V_1), (w^{10}, 0)(i, 1),$
$(w^8, f_3(0,0), V_1), (i, x-1),$
$(w^8, f_3(0, V_1), V_2), (i, x)]$.

$\tau_2 = [(w^8, f_3(0,0), V_1'),$
$(w^8, f_3(0,0), V_1)$
$(w^8, f_3(0,0), V_1'), (i, x-1)$
$(w^8, f_3(0, V_1'), V_2), (i, x)]$.

$\Rightarrow |Diff(\tau_1, \tau_2, w^8)| = 1$.

However.
$max \{ |Diff(\tau_1, \tau_2, w^8)| \mid \forall \tau_1, \tau_2.$
$Dep(w^{10}, \tau_1, \tau_2, w^8, c) \}$
$= 1$.

$E(c) = \{ (x^i, y^j, \omega) \mid \omega = max \{ |Diff(\tau_1, \tau_2, y^j)| \mid \forall \tau_1, \tau_2$, s.t. $Dep(x^i, \tau_1, \tau_2, y^j, c) \}$
$\wedge \omega \geq 1 \wedge x^i, y^j \in LV(c) \}$

$Diff(\tau_1, \tau_2, x^i) \triangleq \{ k \mid (k = 0, \ldots, l_{min}-1 \wedge Seq(\tau_1) x^i[k] \neq Seq(\tau_2, x^i)[k])$
$\wedge (Seq(\tau_{max}, x^i)[k-1] \neq Seq(\tau_{max}, x^i)[k] \wedge k = l_{min}, \ldots, l_{max})$
$\wedge l_{min} = min(|Seq(\tau_1, x^i)|, |Seq(\tau_2, x^i)|)$
$l_{max} = max(\ldots)$
τ_{max} = longer Sequence. τ $\}$

$$\Rightarrow Seq(\tau, x^i) \triangleq \begin{cases} seq(\tau', x^i) :: \nu & \tau = \tau' :: (x, i, \nu, \cdot) \\ seq(\tau', x^i) :: \alpha. & \tau = \tau' :: (x, i, \nu, \alpha) \\ [] & \tau = [] \\ seq(\tau_1') & o.w. \end{cases}$$

$Dep(x^i, \tau_1, \tau_2, y^j, C) \triangleq$
$\exists \xi_1', \tau_0, \tau_0'$
$\xi_1 = (x, i, \_, \_)$. $\quad \xi_2 = (y, j, \_, \_)$.
$\xi_1 \neq_{event} \xi_1'$
$\langle C, \tau_0 \rangle \to \langle C_1, \tau_0' :: \xi_1 \rangle \to \langle skip, \tau_0' :: \xi_1 ++ \tau_1 \rangle$
$\wedge \quad \langle C_1, \tau_0' :: \xi_1' \rangle \overset{*}{\to} \langle skip, \tau_0' :: \xi_1 ++ \tau_1 \rangle$
$Diff(\tau_1, \tau_2, y^j) \neq \phi$.

⇒ example:

Limitation-I-Optimize Flowsto

Monday, May 23, 2022 2:18 PM

⇒ Redue the "flowsto" Over-approximate

I: eliminate Useless Data-Control Flow.

① Redundant If Branch.

case 1: Normalized Boolean Exp in Guard is constant.

⇒ Remove another Block.

⇒ Remove the data-control "flowsto" of variables assigned inside the Block on Variables in guard.

$x \leftarrow 0$.

if $(x<3, [y=1], [y=2])$

$\rightsquigarrow$ $\exists v, \forall \tau, \langle \tau, b \rangle \Downarrow v$

case 2: 2 Blocks in two Branches are syntactically identity.

⇒ Remove another Block.

⇒ Remove the data-control "flowsto" from all Variables in guard to variables assigned inside the Block.

if $(b, [y=1], [y=1])$.

$\rightsquigarrow$ 2 branches are the same

② Redundant While loop.

Case 1: Normalized Boolean Exp in Guard is constant false.

⇒ Remove the Block of the while Body

⇒ Remove the data-control "flowsto" from all Variables in guard to variables assigned inside the Block.

$x \leftarrow 0$

while $x > 0$,

$y = 1$.

$\rightsquigarrow$ same as if-case 1. $\exists v. \forall \tau, \langle \tau, b \rangle \Downarrow v$.

③ Redundant variables in Guard.

For b of guard in every if and while command.

$\forall x \in VAR(b). \quad x \notin VAR(NF(b))$.

Remove all the data-control "flowsto" from x to variables in the Body Blocks

$x \leftarrow q(0)$

$\rightsquigarrow$ if $([x-x=0], [y=1], [y=2])$.

variable disappeared from VAR(b), or VAR(e) or VAR(α) by normalization

II. Remove redundant data-value "flowsto".

for the e and ψ in every assignment command. $y \leftarrow e$. and $y \leftarrow \psi$

$\forall x \in VAR(e)$, $x \in VAR(NF(e))$, or $x \in VAR(\psi) \wedge x \notin VAR(NF(\psi))$

Remove the data-value "flowsto" from x to y

$x \leftarrow q(0)$

$y \leftarrow q(x-x)$.

$\rightsquigarrow$ same as boolean expression.

remove dependency after normalization.

III: Non-trival "liveness Analysis" $\rightsquigarrow$ variable Dead.

example 1

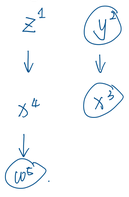
$z \leftarrow 0$

$y \leftarrow q(0)$;

$x \leftarrow q(y+1)$;

$x \leftarrow z+1$;

$w \leftarrow q(x)$.



⇒ Expected Adapt = 2

if liveness refined ⇒

⇒ Analysed Adapt = 1.

un-sound. ×

$\rightsquigarrow$ cannot further improve.

⇒ Cannot further Optimize.

examples:

Event Reexitting occurs in a longer adaptivity chain, and removing the other vertex is safe, keeping both flow doesn't lead to over-approximation.

V.

eg: $z \leftarrow q(0)$

$y \leftarrow 0$

$x \leftarrow y+1$

$x \leftarrow q(z+1)$

⇒ Expected Adapt = 2

⇒ liveness refined ⇒

⇒ Adapt = 2.

No over-approximation

IV: Non-trivial Over-Approximation:

path Sensitive syntactically but not semantically:

⇒ 2 if command have the same guard, the take the same branch always.

the adaptivity Expected by $\max\{ Adap_{true\,branch\,1} + Adap_{true\,branch\,2},\ Adap_{f\text{-}branch\,1} + Adap_{f\text{-}branch\,2} \}$

the $Adapt_{program} = \max\{ Adap_{true\,branch\,1} + Adap_{true\,branch\,2},\ Adap_{f\text{-}branch\,1} + Adap_{f\text{-}branch\,2},\ A_{t\text{-}branch\,1} + A_{f\text{-}branch\,2},\ A_{f\text{-}branch\,1} + A_{t\text{-}branch\,2} \}$

eg: input a: $y \leftarrow q(0)$

if $a > 0$

$y \leftarrow q^{100}(y)$

else

$y \leftarrow q(y)$.

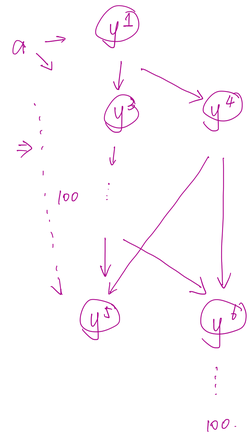
if

$a > 0$:

$y \leftarrow q(y)$

else

$y \leftarrow q^{100}(y)$.



longest path: 201. if $100 = n$ ⇒ $2n+1$

Expected Adapt: 102. ⇒ $n+2$.

Limitation-I-Flowsto-Overapproxi

Friday, May 20, 2022 10:28 AM

inthe CFG Analysis: in $x = y - y$.

Semantic dependency $Dep(x^i, y^j, c)$ is traditionally

① over-approximated syntactically by VAR(Ae). and VAR(b).

we can improve by normolize the expression syntactically.

$$VAR(a_1 - a_2) = \{y \mid y \in VAR(a_1) \cup VAR(a_2) \wedge NF(a_1) \neq NF(a_2)\}.$$

$$VAR(b(a_1 = a_2)) = \{y \mid y \in VAR(a_1) \cup VAR(a_2) \wedge NF(a_1) \neq NF(a_2)\}.$$

$x = 1$

if $x = x$

then $y = 1$

else $y = 2$.

or

$x = 1$

if $x = 1$:

then $y = 1$

else $y = 2$

② also over-approximated syntactically by conditional commands:

$x = 2$

if $x > 0$

then $y = 1$

else $y = 1$.

Example-II

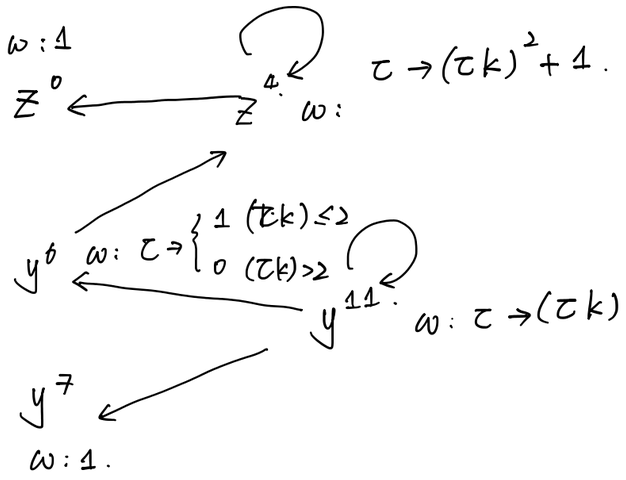
Tuesday, July 12, 2022 4:12 PM

Example II:

```
[z = query(0)]^0
j = k^2 + 1   ^1
while j > 0   ^2
 { j ← j - 1  ^3
    z ← query(z * X[j])^4 }
if  k ≤ 2   ^5
 then {[y ← query(z)]^6}
 else {y ← 0 ; }^7
i ← k ^8
while  i > 0 ^9 do
 {  i ← i - 1 ;^10
    y ← query(y) ;^11 }
```

in previous definition.

⇒



Iv consider both vertices weight & ~~edge~~ weight. v.

restricted walk : $\tau_0 \to$

{ edge seq : $(e_1, \cdots e_n)$ s.t. $cnt(e_i) \le w_{e_i}(\tau_0)$

with Vertices seq $(v_1, \cdots v_{n+1})$. $cnt(v_i) \le w_{v_i}(\tau_0)$.

$q_{len}(k, c) : \tau_0 \to \mathbb{N}$

$\Rightarrow q_{len}(k,c)(\tau_0) = \#\{ 1 \mid v_i \in \bigcup V(c) \wedge v_i \in k(\tau_0) \wedge k \in WALK(c) \}$.

$\Rightarrow A(c) : \tau_0 \to \mathbb{N}$.

$A(c) = \tau_0 \to \max\{ q_{len}(k,c)(\tau_0) \mid k \in \mathcal{WK}(c) \}$.

⇓

better ?

⇒

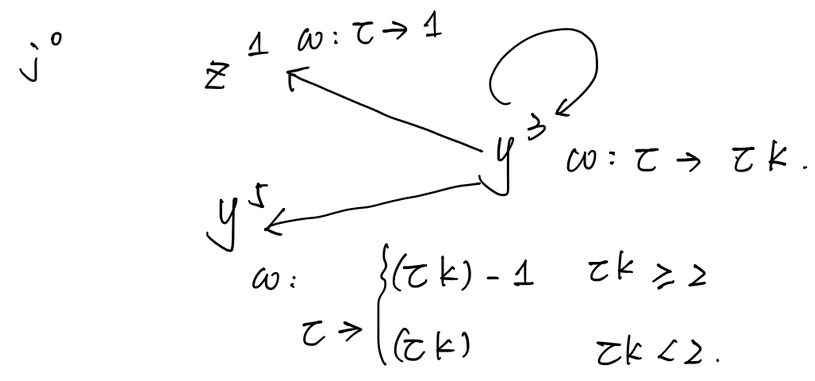
$$A(c) = \tau \to \begin{cases} (\tau k)^2 + 2 + (\tau k) & k \le 2 \\ (\tau k)^2 + 1 & k > 2. \end{cases}$$

Example Limitation:

still the example of limitation:

Simplified: MultiRounds Single(k):

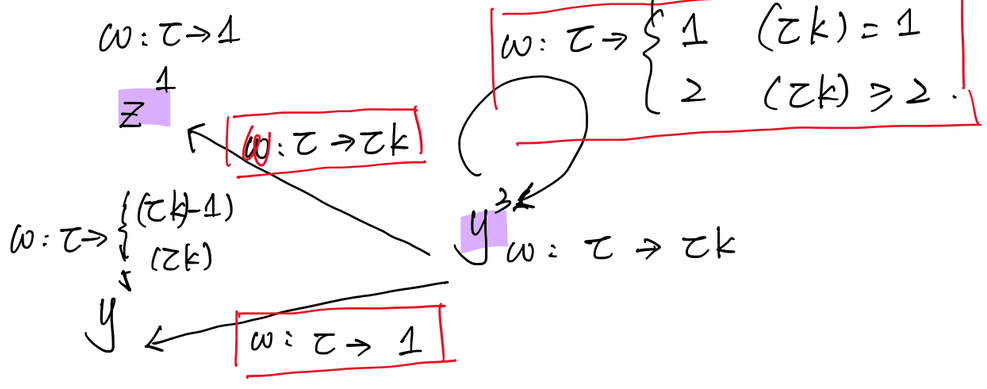
```
[j ← k]^0
[z ← query(0)]^1
while [j > 0]^2 do
{ [y ← query(y * z)]^3;
  if ( [j ≠ 2]^4 , [y ← 0]^5 , [skip]^6 ) ;
  [j ← j - 1]^7
}
```



⇒ adaptivity. indeed $\tau \to \begin{cases} 1 & (\tau k) = 0 \\ 2 & (\tau k) = 1 \\ 3 & (\tau k) \ge 2 \end{cases}$

$A(c) = \tau \to (\tau k) + 1$.

under New Definition.



⇒ $A(c) = \tau \to \begin{cases} 1 & (\tau k) = 0 \\ 2 & (\tau k) = 1 \\ 3 & (\tau k) \ge 2 \end{cases}$

New Over-Approximate Examples

Monday, July 11, 2022 11:21 AM

⇒ over-approximate dependency between variable outside loop. v.s. inside.

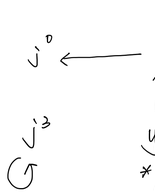
⇒ change a constant assignment for variable x outside loop.

will cause sequence of difference for variable y inside loop. if $y \in x$.

⇒ dependency actually 1. but edge weight is iteration #.

Example:

```
[j ← k]^0
[x ← query(0)]^1
while [j ≥ 0]^2 do
    [j ← j - 1]^3
    [y ← query(x)]^4.
```



edge (y^4, x^1).

$\tau_0 = [(k, in, \cdot), \cdot)]$.

$[(j,2)$ $(x, 1, V_1, 0), (j, 1), (y, 4, V_2, V_1)$ ⇒ $Seq(\tau_1, \tau_2, y^4)$

$(j, 0)$ $(y, 4, V_2', V_1), (j, -1), (y, 4, V_2'', V_1)]$ ⇒ $[V_1; V_1, V_1]$

$[V_1', V_1', V_1']$.

$[(j, >), (x, 1, V_1', 0)$ $(y, 4, V_2, V_1'')$

$(y, 4, V_2', V_1'')$ $(y, 4, V_2'', V_1'')]$

⇒ $|Diff| = 2$.

⇒ $w = \tau \to \tau k$.

⇒ 2 ways:

I √ consider both vertices weight & ~~edge~~ weight. V.

restricted walk : $\tau_0 \to$

$\{$ edge seq : $(e_1, \cdots e_n)$ s.t. $cnt(e_i) \leq w_{e_i}(\tau_0)$

with Vertices seq $(v_1, \cdots v_{n+1})$. $cnt(v_i) \leq w_{v_i}(\tau_0)$.

$flen(k, c) : \tau_0 \to \mathbb{N}$

⇒ $flen(k, c)(\tau_0) = \#\{ i \mid v_i \in Q V(c) \wedge v_i \in k(\tau_0) \wedge k \in WALK(c) \}$.

⇒ $A(c) : \tau_0 \to \mathbb{N}$.

$A(c) = \tau_0 \to \max\{ flen(k, c)(\tau_0) \mid k \in WK(c) \}$.

⇓

better ?

both works.

⇒ better.

II :

$Diff\ seq(\tau_1, \tau_2, x^l) \triangleq$

$\{ k \mid (k = 0 \wedge Seq_1[0] \neq Seq_2[0]$. or $Seq_1 = [])$

$\vee$ $(k = 1 \ldots len(Seq_2),\ Seq_2[k-1] \neq Seq_2[k]) \}$.

$\cup \{ Seq_1[0] \mid Seq_2 = [] \}$.

$Diff \triangleq (len(Seq_1) \neq len(Seq_2)) \vee (Seq_1[0] \neq Seq_2[0])$

⇒ $Diff\ seq(\tau_1, \tau_2, x^l) \triangleq$

$\{ \vee \mid V = Seq_1[0]$ $(Seq_1 \neq [] \wedge Seq_2 = [])$

$V = Seq_2[k]$ $(Seq_2 \neq [] \wedge Seq_1 = [] \wedge Seq[k] \neq Seq_2[k+1],\ k = 0, \ldots len(Seq_2)$

$V = Seq_2[k]$ $(Seq_2 \neq [] \wedge Seq_1 \neq [] \wedge Seq_1[0] \neq Seq_2[0],\ Seq[k] \neq Seq_2[k+1],\ k = 0, \ldots len(Seq_2)$ $\}$

⇒ $|Diff(\tau_1, \tau_2, y)|$ in example is 1. ⇒ weight of edge $\tau \to 1$ ⇏.

but it causes over approximate still.

because even $\vee Seq[k+1] \neq Seq[k]$, the variation still doesn't comes from modification of variable outside of loop.

→ $|Diff| = 1$.

New Definition

Monday, July 11, 2022 9:26 AM

⇒ Dep Graph:

$V_P = \{ x^i \mid x^i \in LV(C) \}$

$E_P = \left\{ (x^i, y^j, w) \;\middle|\; w = \max\{ |Diff(\tau_1, \tau_2, y^j)| \mid \forall \tau_1, \tau_2 .\ s.t.\ Dep(x^i, \tau_1, \tau_0, y^j) \} \right.$
$\left. \wedge\ w \geq 1 \ \wedge\ x^i, y^j \in LV(C) \right\}$

functional ⇒ $\begin{cases} w : \tau_0 \to \mathbb{N} . \ \wedge\ x^i, y^j \in LV(C) \ \wedge \\ w(\tau_0) = \max\{ |Diff(\tau_1, \tau_2, y^j)| \mid \forall \tau_1, \tau_2 .\ s.t.\ Dep(x^i, y^j, \tau_1, \tau_2, \tau_0, C) \} \end{cases}$

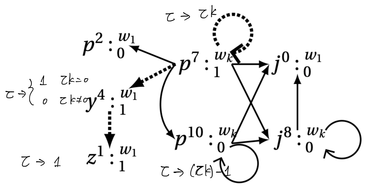
$Diff(\tau_1, \tau_2, x^i) \triangleq \{ k \mid (k = 0, \ldots, l_{min}-1 \wedge Seq(\tau_1) x^i[k] \neq Seq(\tau_2, x^i)[k]) \}$
$\wedge\ (Seq(\tau_{max}, x^i)[k+1] \neq Seq(\tau_{max}, x^i)[k] \wedge k = l_{min}, \ldots, l_{max})$
$\wedge\ l_{min} = \min(|Seq(\tau_1, x^i)|, |Seq(\tau_2, x^i)|)$
$l_{max} = \max(\cdots)$
τ_{max} = longer Sequence. t

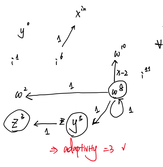
⇒ $Seq(\tau, x^i) \triangleq \begin{cases} seq(c, x^i) := v & \tau = \tau' :: (G, l, v, \cdot) \\ seq(c, x) = \alpha & \tau = \tau' :: (A, l, v, \alpha) \\ \square & \tau = \square \\ seq(\tau') & o.w. \end{cases}$

$Dep(x^i, \tau_1, \tau_2, y^j, C) \triangleq$
$\exists\ \tau_1', \epsilon_1', \ \epsilon_1 = (x^k, i, \_, \_) , \ \epsilon_2 = (y, j, \_, \_) . \ \epsilon_1 \neq \epsilon_1'$
$\Rightarrow$ $\langle C, \tau_1 \rangle \to \langle C_1, \tau_0' :: \epsilon_1' \rangle \xrightarrow{*} \langle skip, \tau_0' :: \epsilon_1 ++ \tau_1 \rangle$
$\langle C_1, \tau_0' :: \epsilon_1' \rangle \xrightarrow{*} \langle skip, \tau' :: \epsilon_1' ++ \tau_2 \rangle$
$\wedge$ $Diff(\tau_1, \tau_2, y^j) \neq \emptyset$.

⇒ Example:

```
multipleRoundsSingle(k)
[j ← 0]^1; [z ← query(0)]^2; [p ← 0]^3;
if ([k = 0]^3, [y ← query(z)]^4, [skip]^5);
while [j ≠ k]^6 do
([p ← query(y[y] + p)]^7; [j ← j + 1]^8
if ([j ≠ k − 2]^9, [p ← 0]^10, [skip]^11));
```





⇒ $Dep(w^8, \tau_1, \tau_2, w^8, C)$, with $\max |Diff(\tau_1, \tau_2, w^8)|$

$\forall x \in \mathbb{N} .$
$x = 0$
$\tau_1 = [\langle w^8, \phi_{\beta_1}(0,0), V_1 \rangle, \langle w^{10}, 0 \rangle, \langle i, 1 \rangle, \langle w^8, \phi_{\beta_1}(1,0), V_1 \rangle, \langle w^{10}, 0 \rangle \langle i, x \rangle, \langle w^8, \phi_{\beta_1}(0,0), V_1 \rangle, \langle i, x-1 \rangle, \langle w^8, \phi_{\beta_1}(0, V_1), V_1 \rangle, \langle i, x \rangle]$

$\tau_2 = [\langle w^8, \phi_{\beta_1}(0,0), V_1' \rangle, \langle w^8, \phi_{\beta_1}(0,0), V_2 \rangle \langle w^8, \phi_{\beta_1}(0, V_2), V_2 \rangle, \langle i, x-1 \rangle \langle w^8, \phi_{\beta_1}(0, V_2), V_2 \rangle, \langle i, x \rangle]$.

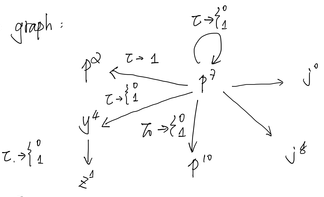
$\max \{ |Diff(\tau_1, \tau_2, w^8)| \mid \forall \tau_1, \tau_2 . \ Dep(w^0, \tau_1, \tau_2, w^8, C) \}$ (Hmmer)
$= 1$

⇒ $|Diff(\tau_1, \tau_2, w^8)| = 1$.

⇒ $[j \leftarrow k]^0 ; [z \leftarrow query(0)]^1 ; [p \leftarrow 0]^2 ;$
$if ([k = 0]^3, [y \leftarrow query(z)]^4, [skip]^5) ;$
$while [j \geq 0]^6 \ do$
$([p \leftarrow query(y + p)]^7 ;$
$[j \leftarrow j - 1]^8 ;$
$if ([j \neq 1]^9, [p \leftarrow 0]^{10}, [skip]^{11}) ;)$

New Dep graph:

⇒



edge: (y^4, z^1) $\quad \tau_2 = \tau_0 +$ . . . . . $(y, 4, V_2', V_2')]$

$\tau_0 = [\langle k, in, 0, \cdot \rangle] . \quad \tau_1 = [\langle p, 2, 0, \cdot \rangle, \langle k=0, 3, true, \cdot \rangle, \langle y, 4, V_2, V_2 \rangle]$

$Dep(y^4, z^1, \tau_1, \tau_2, C) \quad \epsilon_1 = (z, 4, 1, V_2, \phi_0) \quad \epsilon_2' = (z, 1, V_2', 0)$

$Diff(\tau_1, \tau_2, y^4) = \{ (V_2', V_2) \}$.

⇒ $(z^1, y^4, \omega_1 : \tau_0 \to \begin{cases} 1 & \rho(\tau_0) k = 0 \\ 0 & \rho(\tau_0) k \neq 0 \end{cases} .)$

edge: (p^7, p^7)

⇒ $\tau_0 = [\langle k, in, 3, \cdot \rangle] .$

$\tau_1 = [\langle p, 7, V_2, 0 \rangle, \langle j, 8, 2, \cdot \rangle, \langle j \neq 1, true, \cdot \rangle, \langle p, 10, 0, \cdot \rangle,$
$\langle j \geq 0, true \rangle, \langle p, 7, V_1', 0 \rangle, \langle j, 1 \rangle \quad \langle j \neq 1, false \rangle,$ ⇒
$\langle j \geq 0, true \rangle, \langle p, 7, V_1'', V_1' \rangle, \langle j, 0 \rangle \quad \langle j \neq 1, true \rangle \quad \langle p, 10, 0 \rangle .$
$\langle j \geq 0, true \rangle \quad \langle p, 7, V_1''', 0 \rangle, \langle j, -1 \rangle \quad \langle j \neq 1, true \rangle \quad \langle p, 10, 0 \rangle,$
$\langle j \geq 0, false \rangle] .$

$\tau_2 = [\langle p, 7, V_1, 0 \rangle, \langle j, 8, 2, \cdot \rangle \quad \langle j \neq 1, true \rangle \quad \langle p = 0 \rangle$
$\langle p, 7, V_2, 0 \rangle \ \langle j, 1 \rangle \quad (false)$
$\langle p, 7, V_2', V_2 \rangle \ \langle j, 0 \rangle \quad (true) \quad \langle p, 0 \rangle$
$\langle p, 7, V_2'', 0 \rangle \quad \langle j, -1 \rangle \quad (true) \quad \langle p, 0 \rangle$

⇒ $Seq(\tau_1, \tau_2, p^7) = \langle \begin{matrix} V_1' \cdot 0 \\ V_2 \cdot 0 \end{matrix} \Rightarrow |Diff| = 1 .$

if $\tau_0 = [\langle k, in, 1, 0, \cdot \rangle] \quad \tau_2 = [\langle k, in, 0, \cdot \rangle] . \ |Diff| = 0.$

⇒ $\tau \Rightarrow \begin{cases} 1 & \tau_0 k \geq 2 . \\ 0 & \tau_0 k \leq 1 . \end{cases}$

edge (p^7, p^{10}) :

$\tau_0 \ k = [\langle k, in, 3, \cdot \rangle] .$

⇒ $\tau_2 = [\cdots \quad \langle p, 10, 1, \cdot \rangle,$
$\ldots \langle p, 7, V_1', 1 \rangle \langle j, 1 \rangle$
$\langle p, 7, V_1'', V_1' \rangle \langle j, 0 \rangle \quad \langle p, 0 \rangle$
$\langle p, 7, V_1''', 0 \rangle \quad \langle j, -1 \rangle \quad \langle p, 0 \rangle] .$

⇒ $Seq(\tau_1, \tau_2, p^7) = \langle \begin{matrix} 2 \cdot V_1' \cdot 0 \\ 1 \cdot V_1' \cdot 0 \end{matrix} \Rightarrow |Diff| = 1 .$

$\wedge\ \tau_0 \ k = 0 \Rightarrow |Diff| = 0$

⇒ $\tau \Rightarrow \begin{cases} 1 & \tau_0 k \geq 1 \\ 0 & \tau_0 k = 0 . \end{cases}$

⇒ adaptivity : $\begin{cases} 2 & \tau_0 k = 0 \\ 0 & \tau_0 k = 1 \\ 1 & \tau_0 k \geq 2 \end{cases}$

⇒ the over-approximation for dependency depth between variables inside loop caused from weight, & Dependency based on different traces.

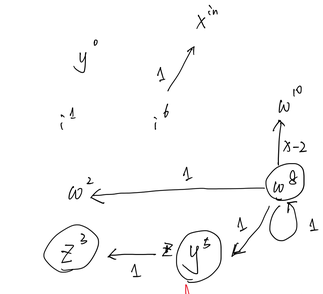
Motivating Example

Friday, June 24, 2022 12:36 PM

$E = \{ (x^i, \omega, y^j) \mid x^i, y^j \in LVar \wedge \omega = max\{ \mathcal{F}(\tau) \mid \forall \tau. Dep(x^i, y^j, \tau, c) \} \}$

example: input x:

```
y=0;
i=0;
w=0;
z <- q1;
if x = 0
{ y <- q2(z); i=x-1}
else
{skip};
while i \neq x
{w <- q3(y,w); if i neq x-2 then w=0 else skip ; i=i+1}
```



⇒ adaptivity = 3 ✓

a trace τ_1. $\mathcal{E}_1 \overset{=}{\mathcal{E}} \omega, 2$. $\mathcal{E}_2 \overset{=}{\mathcal{E}} \omega, 2$.

change $\mathcal{E}_2 \overset{=}{\mathcal{E}} \omega, 2'$ ⇒ $\mathcal{E}_2' = \mathcal{E}_{\omega, 2}'$

then another trace: $\mathcal{E}_1 = \omega, 2$. $\mathcal{E}_2 = \omega, 2$, $\mathcal{E}_3 = \omega, 2$.

change ↓ $\mathcal{E}_1' = \omega, 2, v'$. ||| $\mathcal{E}_2 = \omega_8, 2$. ↓change $\mathcal{E}_3' = \omega^8, 2, v_3'$.

∀ trace. s.t. $\mathcal{E}_2, \tau, \mathcal{E}$ Dep $(\omega^2, \tau, \omega^2)$, # of occurrence time of 2 ≥ # of event where value of ω^2 is changed.

then. count: ∀ trace τ s.t. Dep $(\omega^2, \tau, \omega^2)$.

$|Diff(\tau_1, \tau_2, \omega^2)|$

⇒ weight on edge (x^1, y^j, ω).

$\omega = max\{ |Diff(\tau_1, \tau_2, y^j)| \mid \forall \tau_1, \tau_2$. s.t. Dep $(x^i, \tau_1, \tau_2, y^j) \}$.

⇒ 1. Dep on 2 traces. redefine.

⇒ 2. Control Dep. ??

⇒ add iteration id.

incentive to know the value changed in the 2 traces are in the same iteration.

than,

vs possible unsound ??

if cases ∀ t. observe the same value, but in different iterations ? ⇒

than intuitively no-dependency. it is good.

⇒⇒. Diff (τ_1, τ_2, x^i)

⇒ ~~trace~~

$$seq(\tau_2)\, x^i \triangleq \begin{cases} \tau_2 = \tau_2' :: (x, i, \nu, \iota) \to seq(\ell\, \tau_2') :: \nu \\ \tau_2 = \tau_2' :: (x, i, \nu, \alpha) \to seq(\tau_2') :: \alpha. \\ \tau_2 = [] \to \ell[] \\ o.w. \to seq(\tau_2') \end{cases}$$

without loop iteration id.

for $seq(\tau_2)\, x^i$. where Dep $(y^j, x^i, \tau_1, \tau_2)$.

cases of $seq(\tau_1) \neq seq(\tau_2)$:

① $len_1 \neq len_2$. ⇒ $Diff(seq(\tau_1), seq(\tau_2), x^i) = \{k \mid seq(\tau_1)[k] \neq seq(\tau_2)[k]\} \cup \{len_{\leq 1}, len_1+1, \ldots len_2\}$.

② $len_1 = len_2$ ⇒ $Diff(seq(\tau_1), seq(\tau_2), x^i) = \{k \mid seq(\tau_1)[k] \neq seq(\tau_2)[k]\}$

⇒ potential issue: the different observed value is in different iterations.

case: if guard is affected by moving to different iteration, but it always executed the same time with same value.

then it doesn't matter which iteration it is in, there isn't dependence / adaptivity.

⇒ for case ②. where they have some len, it doesn't loss adaptivity if I observe the same value in the same location of $seq(\tau_1)$ and $seq(\tau_2)$

because where they actually comes from different loop iteration.

i.e. it is assured that only concern the diff of some location.

⇒ for case ①. $len_1 \neq len_2$.

in the 2 seqs from $[0, min(l_1, l_2)]$. it is still sound by just observe the different value in some location,

for the same reason as case ②.

in the longer seq from $[min(l_1, l_2), max(l_1, l_2)]$.

case i = 0
while i < x do:
i = i + 1
y = f(a).

⇒ when x = 100, still safe.

$(y, 1, v_1)$ $(y, 1, v_2)$ $(y, 1, v_3)$ $(y, 2, 0 \ldots)$

$(y, 1, v_1')$ $(y, 1, v_2')$ $(y, 1, v_3')$ $(y, 2, 0) \ldots$

⇒ $Guy^*(inf, 1)$.

in order to get different length, only way is by changing value affect guards.

if in the ~~long~~ seq from $min(l_1, l_2)$ to $max(l_1, l_2)$.

$(x, in, 1)$ $(y, 1, v_2)$

$(x, in, 100)$, $(y, 1, v_2)$, $(y, 1, v_3)$ ...

⇒ all the value are the same. than adaptivity is actually 1.

⇒ if they are different, then either ⟨ guard variable x is used in evaluating y. or y itself is used (through a chain) is used in evaluating y.

2 approaches: ⇝

i = 0
while i < x.
ω = f(ω)
if i < 3, ω = 0, skip
i = i + 1.

x = 2: (ω, 0), (ω, 1, 0)

x = 100: (ω, 1, 0) (ω, 1, 0) (ω, 1, f(1)) (ω, 1, f(f(1))) ....

Dep: infinite.

case: i = 0
while i < x:
ω = f(ω)
if i > 3, ω = 0, skip. x = 100: (ω, 1, f(1)). (ω, 1, f(1)) (ω, 1, f(3)), (ω, 1, 0) (ω, 1, 0) ...
i = i + 1

⇓

two approaches:

① prove there must exist value dependency from x to y (unclear to prove, because change x could change length and value in the same time.) or from other variable to y.

② define |Diff| as { seq(τmax)[min : max] }. ⇝ sound. set of unique values.

possible case. (1, 0, 1, 0, 1, 0) I acctually have no intuition does this adaptive or not.

⇒ if it is. the |Diff| defined as { k | seq[k-1] ≠ seq[k] } can capture.

but "intuitive adaptivity" is unclear.

⇒ Dep $(\omega^8, \tau_1, \tau_2, \omega^8, c)$, with max $|Diff(\tau_1, \tau_2, \omega^8)|$

∀ x ∈ ℕ.
x = 4

$\tau_1 = [(\omega^8, f_3(0,0), V_1), (\omega^{10}, 0), (i, 1),$
$(\omega^8, f_3(0,0), V_1), (\omega^{10}, 0) (i, 2),$
$(\omega^8, f_3(0,0), V_1), (i, x-1),$
$(\omega^8, f_3(0, V_1), V_2), (i, x)]$.

$\tau_2 = [(\omega^8, f_3(0,0), V_1'),$
$(\omega^8, f_3(0,0), V_1')$
$(\omega^8, f_3(0,0), V_1'), (i, x-1)$
$(\omega^8, f_3(0, V_1'), V_2'), (i, x)]$.

Thm or.

$max\{ |Diff(\tau_1, \tau_2, \omega^8)| \mid \forall \tau_1, \tau_2. Dep(\omega^{10}, \tau_1, \tau_2, \omega^8, c) \} = 1$.

⇒ $|Diff(\tau_1, \tau_2, \omega^8)| = 1$.

⇒ Dep Graph:

$V(c) = \{ x^i \mid x^i \in LV(c) \}$

$E(c) = \{ (x^i, y^j, \omega) \mid \omega = max\{ |Diff(\tau_1, \tau_2, y^j)| \mid \forall \tau_1, \tau_2, s.t. Dep(x^i, \tau_1, \tau_2, y^j, c) \} \wedge \omega \geq 1 \wedge x^i, y^j \in LV(c) \}$

$$Diff(\tau_1, \tau_2, x^i) \triangleq \{ k \mid (k = 0, \ldots, l_{min}-1 \wedge Seq(\tau_1)\, x^i[k] \neq Seq(\tau_2, x^i)[k]) \}$$
$$\wedge (Seq(\tau_{max}, x^i)[k-1] \neq Seq(\tau_{max}, x^i)[k] \wedge k = l_{min}, \ldots, l_{max})$$
$$\wedge l_{min} = min(|Seq(\tau_1, x^i)|, |Seq(\tau_2, x^i)|)$$
$$l_{max} = max(\cdots)$$
$$\tau_{max} = \text{longer Sequence. } \tau \}$$

$$\Rightarrow Seq(\tau, x^i) \triangleq \begin{cases} seq(\tau', x^i) :: \nu & \tau = \tau' :: (x, i, \nu, \iota) \\ seq(\tau', x^i) :: \alpha & \tau = \tau' :: (x, i, \nu, \alpha) \\ [] & \tau = [] \\ seq(\tau') & o.w. \end{cases}$$

$Dep(x^i, \tau_1, \tau_2, y^j, c) \triangleq$

$\exists \mathcal{E}_1', \tau_0, \tau_0'$

$\mathcal{E}_1 = (x, i, \nu_-, \_)$. $\mathcal{E}_2 = (y, j, \_, \_)$.

$\mathcal{E}_1 \neq_{event} \mathcal{E}_1'$

$\langle C, \tau_0 \rangle \to \langle C_1, \tau_0' :: \mathcal{E}_1 \rangle \to \langle skip, \tau_0' :: \mathcal{E}_1 +\!+ \tau_1 \rangle$

$\wedge \quad \langle C_1, \tau_0' :: \mathcal{E}_1' \rangle \overset{*}{\to} \langle skip, \tau' :: \mathcal{E}_1 +\!+ \tau_1 \rangle$

$Diff(\tau_1, \tau_2, y^j) \neq \phi$.

⇒ example: