Temporal Path Conditions in Dependence Graphs

Andreas Lochbihler
Gregor Snelting

University of Passau

30.09.2007

Path Conditions for PDG Paths

PDGs and Information Flow

Edges Data and control dependences

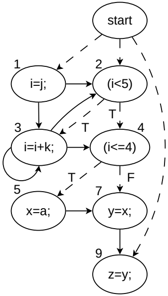
Paths Represent possible information flow

Aim Find an actual program execution for a given path!

Idea Generate formula with predicates over program variables

- Satisfying assignment yields witness
- Allow only one-sided error: Conservative approximation

```
i = j;
while (i<5)
{
  i = i+k;
  if (i<=4)
    x = a;
  else
    y = x;
}
z = y;
```



PDGs and Information Flow

Edges Data and control dependences

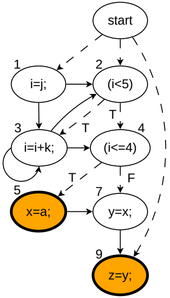
Paths Represent possible information flow

Aim Find an actual program execution for a given path!

Idea Generate formula with predicates over program variables

- Satisfying assignment yields witness
- Allow only one-sided error: Conservative approximation

```
i = j;
while (i<5)
{
  i = i+k;
  if (i<=4)
    x = a;
  else
    y = x;
}
z = y;
```



Information flow:

$$x = a \longrightarrow z = y$$

Path Conditions for PDG Paths

PDGs and Information Flow

Edges Data and control dependences

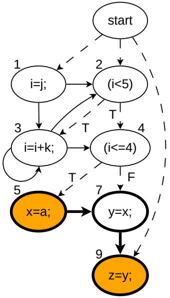
Paths Represent possible information flow

Aim Find an actual program execution for a given path!

Idea Generate formula with predicates over program variables

- Satisfying assignment yields witness
- Allow only one-sided error: Conservative approximation

```
i = j;
while (i<5)
{
  i = i+k;
  if (i<=4)
    x = a;
  else
    y = x;
}
z = y;
```



Information flow:

$$x = a \rightarrow_x y = x \rightarrow_y z = y$$

Boolean Path Conditions (Snelting, Krinke, Robschink)

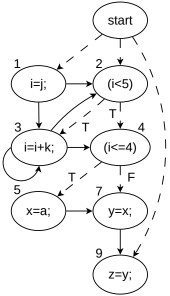
Original Approach:

- Every node on the path must be executed
- Use execution conditions from control dependence (SSA form)
- Formula is conjunction of execution conditions

Drawbacks:

- Variable renaming (loss of precision) in the presence of CFG loops
- Temporal relationships not expressible ($\wedge$ is commutative)
- Solving for input variables difficult

```
i = j;
while (i<5)
{
  i = i+k;
  if (i<=4)
    x = a;
  else
    y = x;
}
z = y;
```



Boolean Path Conditions (Snelting, Krinke, Robschink)

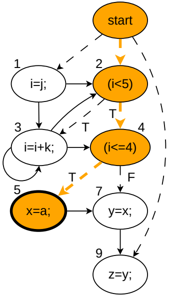
Original Approach:

- Every node on the path must be executed
- Use execution conditions from control dependence (SSA form)
- Formula is conjunction of execution conditions

Drawbacks:

- Variable renaming (loss of precision) in the presence of CFG loops
- Temporal relationships not expressible ($\wedge$ is commutative)
- Solving for input variables difficult

```
i = j;
while (i<5)
{
  i = i+k;
  if (i<=4)
    x = a;
  else
    y = x;
}
z = y;
```



$$\mathrm{E}(x = a) = i < 5 \wedge i \leq 4$$

Boolean Path Conditions (Snelting, Krinke, Robschink)

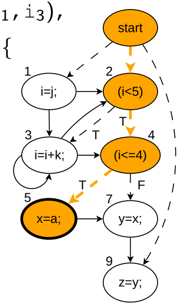
Original Approach:

- Every node on the path must be executed
- Use execution conditions from control dependence (SSA form)
- Formula is conjunction of execution conditions

Drawbacks:

- Variable renaming (loss of precision) in the presence of CFG loops
- Temporal relationships not expressible ($\wedge$ is commutative)
- Solving for input variables difficult

```
i1 = j;
while (i2=Φ(i1,i3),
       i2<5) {
  i3 = i2+k;
  if (i3<=4)
    x = a;
  else
    y = x;
}
z = y;
```



$$\mathrm{E}(x = a) = i_2 < 5 \wedge i_3 \leq 4$$

Boolean Path Conditions (Snelting, Krinke, Robschink)

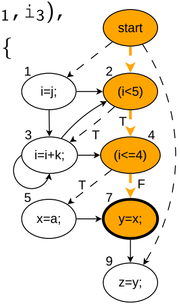
Original Approach:

- Every node on the path must be executed
- Use execution conditions from control dependence (SSA form)
- Formula is conjunction of execution conditions

Drawbacks:

- Variable renaming (loss of precision) in the presence of CFG loops
- Temporal relationships not expressible ($\wedge$ is commutative)
- Solving for input variables difficult

```
i1 = j;
while (i2=Φ(i1,i3),
          i2<5) {
  i3 = i2+k;
  if (i3<=4)
    x = a;
  else
    y = x;
}
z = y;
```



$$\mathrm{E}(x = a) = i_2 < 5 \wedge i_3 \leq 4$$
$$\mathrm{E}(y = x) = i_2 < 5 \wedge i_3 > 4$$

Boolean Path Conditions (Snelting, Krinke, Robschink)

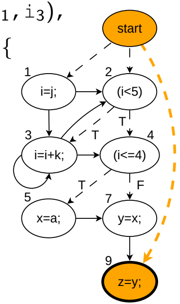
Original Approach:

- Every node on the path must be executed
- Use execution conditions from control dependence (SSA form)
- Formula is conjunction of execution conditions

Drawbacks:

- Variable renaming (loss of precision) in the presence of CFG loops
- Temporal relationships not expressible ($\wedge$ is commutative)
- Solving for input variables difficult

```
i1 = j;
while (i2=Φ(i1,i3),
          i2<5) {
  i3 = i2+k;
  if (i3<=4)
    x = a;
  else
    y = x;
}
z = y;
```



$$\mathrm{E}(x = a) = i_2 < 5 \wedge i_3 \leq 4$$
$$\mathrm{E}(y = x) = i_2 < 5 \wedge i_3 > 4$$
$$\mathrm{E}(z = y) = true$$

Boolean Path Conditions (Snelting, Krinke, Robschink)

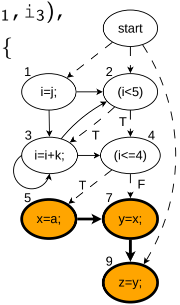
Original Approach:

- Every node on the path must be executed
- Use execution conditions from control dependence (SSA form)
- Formula is conjunction of execution conditions

Drawbacks:

- Variable renaming (loss of precision) in the presence of CFG loops
- Temporal relationships not expressible ($\wedge$ is commutative)
- Solving for input variables difficult

```
i1 = j;
while (i2=Φ(i1,i3),
       i2<5) {
  i3 = i2+k;
  if (i3<=4)
    x = a;
  else
    y = x;
}
z = y;
```



$$i_2 < 5 \wedge i_3 \leq 4 \wedge$$
$$i_2 < 5 \wedge i_3 > 4$$

Boolean Path Conditions (Snelting, Krinke, Robschink)

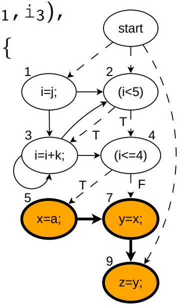
Original Approach:

- Every node on the path must be executed
- Use execution conditions from control dependence (SSA form)
- Formula is conjunction of execution conditions

Drawbacks:

- Variable renaming (loss of precision) in the presence of CFG loops
- Temporal relationships not expressible ($\wedge$ is commutative)
- Solving for input variables difficult

```
i1 = j;
while (i2=Φ(i1,i3),
        i2<5) {
  i3 = i2+k;
  if (i3<=4)
    x = a;
  else
    y = x;
}
z = y;
```



$$i_2 < 5 \wedge i_3 \leq 4 \wedge$$
$$i'_2 < 5 \wedge i'_3 > 4$$

satisfiable

Boolean Path Conditions (Snelting, Krinke, Robschink)

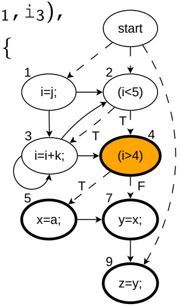
Original Approach:

- Every node on the path must be executed
- Use execution conditions from control dependence (SSA form)
- Formula is conjunction of execution conditions

Drawbacks:

- Variable renaming (loss of precision) in the presence of CFG loops
- Temporal relationships not expressible ($\wedge$ is commutative)
- Solving for input variables difficult

```
i1 = j;
while (i2=Φ(i1,i3),
          i2<5) {
  i3 = i2+k;
  if (i3>4)
    x = a;
  else
    y = x;
}
z = y;
```



$$i'_2 < 5 \wedge i'_3 \leq 4 \wedge$$
$$i_2 < 5 \wedge i_3 > 4$$

satisfiable

Temporal Path Conditions with LTL

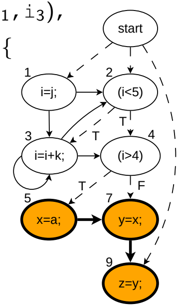
Our approach:

- Data dependences introduce temporal ordering
- Formula models are program state sequences
- Model checker finds a satisfying program trace

Advantages:

- Model checker produces a witness trace
- Extra conditions possible
- Precise CFG loop handling

```
i1 = j;
while (i2=Φ(i1,i3),
        i2<5) {
  i3 = i2+k;
  if (i3>4)
    x = a;
  else
    y = x;
}
z = y;
```



$$\Diamond\Big(\overbrace{i_2 < 5 \wedge i_3 > 4}^{=\mathbb{E}(x=a)} \wedge (i_2 < 5)\,\mathcal{U}\,\big(\underbrace{i_2 < 5 \wedge i_3 \leq 4}_{=\mathbb{E}(y=x)} \wedge \Diamond(i_2 \geq 5)\big)\Big)$$

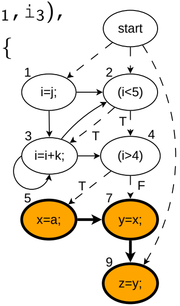
Temporal Reasoning

1. We know (in 5, 7): $i_3 = i_2 + k$
2. With $i_2 < 5 \land i_3 > 4$, we get $k > 0$
3. $i_3 > 4 \ldots \mathcal{U} \ldots i_3 \leq 4$ gives:
   i_3 must be decreased
4. Contradiction to $k > 0$, formula is not satisfiable by any program trace

$\Rightarrow$ No information flow possible along the path

$\Rightarrow$ Same result with model checking

```
1 i1 = j;
2 while (i2=Φ(i1,i3),
          i2<5) {
3   i3 = i2+k;
4   if (i3>4)
5     x = a;
6   else
7     y = x;
8 }
9 z = y;
```



$$\Diamond\Big(i_2 < 5 \land i_3 > 4 \land (i_2 < 5)\,\mathcal{U}\,\big(i_2 < 5 \land i_3 \leq 4 \land \Diamond(i_2 \geq 5)\big)\Big)$$

Temporal vs. Boolean Path Conditions

Temporal path conditions are more precise than boolean path conditions:

- Witness traces for a temporal path condition contain a satisfying assignment for the corresponding boolean path condition

- Temporal path conditions are strictly more precise
  - Extra constraints included
  - Reasoning about temporal ordering

Boolean	**Temporal**
$i_2' < 5 \land i_3' \leq 4 \land i_2 < 5 \land i_3 > 4$	$\Diamond\Big(i_2 < 5 \land i_3 > 4 \land (i_2 < 5)\,\mathcal{U}\,\big(i_2 < 5 \land i_3 \leq 4 \land \Diamond(i_2 \geq 5)\big)\Big)$
Satisfiable	Unsatisfiable
E.g.: $i_2' = 3,\ i_3' = 4,\ i_2 = 4,\ i_3 = 5$	No information flow possible

Conclusion

Temporal path conditions

- improve on boolean path conditions
  (temporal ordering, CFG loop handling, extra constraints)
- are fed to model checkers: Find witness traces
- can also be done for PDG chops

Application to Information Flow Control (noninterference)

Confidentiality Does confidential data flow to public variables?

Integrity Can critical computations be manipulated from outside?

Future work

- Extending the ideas to richer languages (procedures, objects, ...)
- Enhancing the prototype implementation
- Carrying out case studies