The Decidability and Complexity of Interleaved Bidirected Dyck Reachability

Adam Husted Kjelstrøm and Andreas Pavlogiannis
Dyck Reachability

\[ \Sigma = \{ (1,)_1, \ldots, (k,)_k \} \cup \{ \epsilon \} \]

\[ S \rightarrow S \ S \mid (1 \ S )_1 \mid \ldots \mid (k \ S )_k \mid \epsilon \]

\[ G = (V, E, \lambda : E \rightarrow \Sigma) \]
Dyck Reachability

$$\Sigma = \{(1,)_1, \ldots, (k,)_k\} \cup \{\epsilon\}$$

$$G = (V, E, \lambda : E \rightarrow \Sigma)$$

$$S \rightarrow S \ S \ | \ (1 \ S)_1 \ | \ \ldots \ | \ (k \ S)_k \ | \ \epsilon$$

$$P : x \rightsquigarrow z \quad \text{with} \quad \lambda(P) = (1(2)_2)_1$$

Diagram with nodes labeled as $y$, $w$, $x$, $z$ and unspecified edges and labels.
Dyck Reachability

$\Sigma = \{(1,)_1, \ldots (k,)_k\} \cup \{\epsilon\}$

$G = (V, E, \lambda : E \rightarrow \Sigma)$

$S \rightarrow SS | (1 \ S)_1 | \ldots | (k \ S)_k | \epsilon$

$P : x \rightsquigarrow z \text{ with } \lambda(P) = (1(2)_2)_1$

- Alias analysis
- Data-dependence analysis
- Data-flow analysis
- Databases
- ...  
- Impact analysis
- Bloat analysis
- Program slicing
- Network analysis
- ...
Dyck Reachability

\[ \Sigma = \{(1,)_1, \ldots (k,)_k\} \cup \{\epsilon\} \]

\[ G = (V, E, \lambda : E \rightarrow \Sigma) \]

- Data-dependence analysis
- Data-flow analysis
- Databases
- ...  

- Bloat analysis
- Program slicing
- Network analysis
- ...
```c
void setX(Point p, int v){
    p.x = v;
}

int getX(Point r){
    return r.x;
}
...
int a,b;
Point q;
setX(q, a);
b=getX(q);
```
void setX(Point p, int v){
    p.x = v;
}
int getX(Point r){
    return r.x;
}
...
int a,b;
Point q;
setX(q, a);
b=getX(q);
void setX(Point p, int v){
    p.x = v;
}

int getX(Point r){
    return r.x;
}

...  

int a, b;
Point q;
setX(q, a);

b = getX(q);
### Interleaved Dyck Reachability

```c
void setX(Point p, int v){
    p.x = v;
}

int getX(Point r){
    return r.x;
}
...

int a, b;
Point q;
setX(q, a);
setX(q, a);
b = getX(q);
```

**Context sensitivity**

**Field sensitivity**

![Diagram](attachment:image.png)
void setX(Point p, int v) {
    p.x = v;
}

int getX(Point r) {
    return r.x;
}

...  

int a, b;
Point q;
setX(q, a);

b = getX(q);
Interleaved Dyck Reachability

```c
void setX(Point p, int v){
    p.x = v;
}

int getX(Point r){
    return r.x;
}
...
int a,b;
Point q;
setX(q, a);
b=getX(q);
```

- Reachability wrt two Dyck languages $\mathcal{D}_k^1$, $\mathcal{D}_k^2$
- $\{_{10}\}_{10}\{_{11}\}_{11} \in \mathcal{D}_k^1$ and $[_{x}]_{x} \in \mathcal{D}_k^2$
- Thus $\{_{10}[_{x}]_{10}\{_{11}\}_{x}}_{11} \in \mathcal{D}_k^1 \odot \mathcal{D}_k^2$
Interleaved Dyck Reachability

- Interleaved Dyck Reachability has large modeling power in static analysis
Interleaved Dyck Reachability

- Interleaved Dyck Reachability has large modeling power in static analysis
- Perhaps “too” large

**Theorem (Reps ’00)**
\[ \mathcal{D}_k \odot \mathcal{D}_k \text{ reachability is undecidable} \]
Interleaved Dyck Reachability

- Interleaved Dyck Reachability has large modeling power in static analysis
- Perhaps “too” large

**Theorem (Reps '00)**

$D_k \circ D_k$ reachability is undecidable

- Still highly used in practice — approximations, e.g.,


...
Bidirected graphs

\[ \forall u, v \in V : \lambda(u, v) = (i \iff \lambda(v, u) = )_i \]

- Used to handle mutable heap data
- Demand-driven alias analysis
- CFL formulation of pointer analysis


Bidirected graphs

\[ \forall u, v \in V : \lambda(u, v) = (i \iff \lambda(v, u) = i) \]

- Used to handle mutable heap data
- Demand-driven alias analysis
- CFL formulation of pointer analysis


Inclusion-based Alias Analysis

- A Dyck-reachability formulation
- If heap object $o$ Dyck-reaches variable $x$ then $o \in PointsToSet(x)$
Inclusion-based Alias Analysis

- A Dyck-reachability formulation
- If heap object $o$ Dyck-reaches variable $x$ then $o \in \text{PointsToSet}(x)$

```java
1 x = new O(); // Object o1
2 y = new O(); // Object o2
3 ...
4 y = x.f;
5 z = x.f;
```
Inclusion-based Alias Analysis

- A Dyck-reachability formulation
- If heap object \( o \) Dyck-reaches variable \( x \) then \( o \in \text{PointsToSet}(x) \)

```java
1 x = new O(); //Object o1
2 y = new O(); //Object o2
3 ...
4 y = x.f;
5 z = x.f;
```

Inclusion based
If \( y \) may alias \( z \) and \( o \in \text{PointsToSet}(y) \) then \( o \in \text{PointsToSet}(z) \)
Inclusion-based Alias Analysis

- A Dyck-reachability formulation
- If heap object $o$ Dyck-reaches variable $x$ then $o \in \text{PointsToSet}(x)$

```
1. x = new O(); // Object o1
2. y = new O(); // Object o2
3. ...  
4. y = x.f;
5. z = x.f;
```

Inclusion based

If $y$ may alias $z$ and $o \in \text{PointsToSet}(y)$ then $o \in \text{PointsToSet}(z)$
Interleaved Bidirected Dyck Reachability

Interleaving + Bidirectedness, Variants →

1. $D_1 \odot D_1$

2. $D_k \odot D_1$

3. $D_k \odot D_k$

What do we know about this problem?

- $D_1 \odot D_1$ yields a 2-dimensional Vector Addition System with States (VASS)
- Reachability in NL
- Bidirected $D_1 \odot D_1$ in $O(n^7)$
- Bidirected $D_k \odot D_k$ is NP-hard
Interleaved Bidirected Dyck Reachability

Interleaving + Bidirectedness, Variants →

\[
\begin{align*}
1. & \quad D_1 \circ D_1 & \quad \xrightarrow{+1} \quad \xrightarrow{+1} \quad \xrightarrow{-1} \quad \xrightarrow{-1} \\
2. & \quad D_k \circ D_1 & \quad \xrightarrow{\{+1\}} \quad \xrightarrow{-1} \\
3. & \quad D_k \circ D_k & \quad \xrightarrow{\{[\text{ }]\}} \quad \xrightarrow{\text{ ]}}
\end{align*}
\]

What do we know about this problem?

- \( D_1 \circ D_1 \) yields a 2-dimensional Vector Addition System with States (VASS)
- Reachability in NL
- Bidirected \( D_1 \circ D_1 \) in \( O(n^7) \)
- Bidirected \( D_k \circ D_k \) is NP-hard
## Main Results

<table>
<thead>
<tr>
<th></th>
<th>$D_1 \odot D_1$</th>
<th>$D_k \odot D_1$</th>
<th>$D_k \odot D_1$ (bounded counter)</th>
<th>$D_k \odot D_k$</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Upper Bound</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Lower Bound</strong></td>
<td>-</td>
<td>-</td>
<td></td>
<td>-</td>
</tr>
</tbody>
</table>
## Main Results

<table>
<thead>
<tr>
<th></th>
<th>( \mathcal{D}_1 \otimes \mathcal{D}_1 )</th>
<th>( \mathcal{D}_k \otimes \mathcal{D}_1 )</th>
<th>( \mathcal{D}_k \otimes \mathcal{D}_1 ) (bounded counter)</th>
<th>( \mathcal{D}_k \otimes \mathcal{D}_k )</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Upper Bound</strong></td>
<td>( O(n^3 \cdot \alpha(n)) )(^\dagger)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Lower Bound</strong></td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\(^\dagger\) Improves over previous \( O(n^7) \)
### Main Results

<table>
<thead>
<tr>
<th></th>
<th>$\mathcal{D}_1 \circ \mathcal{D}_1$</th>
<th>$\mathcal{D}_k \circ \mathcal{D}_1$</th>
<th>$\mathcal{D}_k \circ \mathcal{D}_1$ (bounded counter)</th>
<th>$\mathcal{D}_k \circ \mathcal{D}_k$</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Upper Bound</strong></td>
<td>$O(n^3 \cdot \alpha(n))$†</td>
<td>Decidable‡</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td><strong>Lower Bound</strong></td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
</tbody>
</table>

† Improves over previous $O(n^7)$

‡ Without bidirectedness, decidability is open (PVASS)
## Main Results

<table>
<thead>
<tr>
<th></th>
<th>$\mathcal{D}_1 \odot \mathcal{D}_1$</th>
<th>$\mathcal{D}_k \odot \mathcal{D}_1$</th>
<th>$\mathcal{D}_k \odot \mathcal{D}_1$ (bounded counter)</th>
<th>$\mathcal{D}_k \odot \mathcal{D}_k$</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Upper Bound</strong></td>
<td>$O(n^3 \cdot \alpha(n))$†</td>
<td>Decidable‡</td>
<td>$O(n^2 \cdot \alpha(n))$</td>
<td></td>
</tr>
<tr>
<td><strong>Lower Bound</strong></td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

† Improves over previous $O(n^7)$
‡ Without bidirectedness, decidability is open (PVASS)
## Main Results

<table>
<thead>
<tr>
<th></th>
<th>$\mathcal{D}_1 \circ \mathcal{D}_1$</th>
<th>$\mathcal{D}_k \circ \mathcal{D}_1$</th>
<th>$\mathcal{D}_k \circ \mathcal{D}_1$ (bounded counter)</th>
<th>$\mathcal{D}_k \circ \mathcal{D}_k$</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Upper Bound</strong></td>
<td>$O(n^3 \cdot \alpha(n))$†</td>
<td>Decidable‡</td>
<td>$O(n^2 \cdot \alpha(n))$</td>
<td>-</td>
</tr>
<tr>
<td><strong>Lower Bound</strong></td>
<td>-</td>
<td>-</td>
<td>OV-hard§</td>
<td>-</td>
</tr>
</tbody>
</table>

† Improves over previous $O(n^7)$
‡ Without bidirectedness, decidability is open (PVASS)
§ $O(n^2)$ is tight (wrt polynomial improvements)
## Main Results

<table>
<thead>
<tr>
<th></th>
<th>$D_1 \odot D_1$</th>
<th>$D_k \odot D_1$</th>
<th>$D_k \odot D_1$ (bounded counter)</th>
<th>$D_k \odot D_k$</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Upper Bound</strong></td>
<td>$O(n^3 \cdot \alpha(n))$†</td>
<td>Decidable‡</td>
<td>$O(n^2 \cdot \alpha(n))$</td>
<td>-</td>
</tr>
<tr>
<td><strong>Lower Bound</strong></td>
<td>-</td>
<td>-</td>
<td>OV-hard◊</td>
<td>Undecidable*</td>
</tr>
</tbody>
</table>

† Improves over previous $O(n^7)$
‡ Without bidirectedness, decidability is open (PVASS)
◊ $O(n^2)$ is tight (wrt polynomial improvements)
* Improves over previous NP-hard
### Main Results

<table>
<thead>
<tr>
<th></th>
<th>$D_1 \circ D_1$</th>
<th>$D_k \circ D_1$</th>
<th>$D_k \circ D_1$ (bounded counter)</th>
<th>$D_k \circ D_k$</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Upper Bound</strong></td>
<td>$O(n^3 \cdot \alpha(n))$†</td>
<td>Decidable‡</td>
<td>$O(n^2 \cdot \alpha(n))$</td>
<td>-</td>
</tr>
<tr>
<td><strong>Lower Bound</strong></td>
<td>-</td>
<td>-</td>
<td>OV-hard○</td>
<td>Undecidable*</td>
</tr>
</tbody>
</table>

† Improves over previous $O(n^7)$
‡ Without bidirectedness, decidability is open (PVASS)
○ $O(n^2)$ is tight (wrt polynomial improvements)
* Improves over previous NP-hard
Theorem
Bidirected $D_1 \otimes D_1$ reachability can be solved in $O(n^3 \cdot \alpha(n))$ time.

- $n$ is the number of nodes
- $\alpha(n)$ is the inverse Ackermann function (practically constant)

Lemma
Without loss of generality, both counters along any witness path $P: u \rightarrow v$ remain bounded by $O(n^2)$. 
**Theorem**

*Bidirected $\mathcal{D}_1 \odot \mathcal{D}_1$ reachability can be solved in $O(n^3 \cdot \alpha(n))$ time.*

- $n$ is the number of nodes
- $\alpha(\cdot)$ is the inverse Ackermann function (practically constant)
Theorem

Bidirected $\mathcal{D}_1 \circ \mathcal{D}_1$ reachability can be solved in $O(n^3 \cdot \alpha(n))$ time.

- $n$ is the number of nodes
- $\alpha(\cdot)$ is the inverse Ackermann function (practically constant)

Lemma

Without loss of generality, both counters along any witness path $P: u \leadsto v$ remain bounded by $O(n^2)$. 
One Key Idea

Bidirectedness $\implies$ Boundedness

If $u$ reaches $v$ then there is a witness path where the counters are bounded.

$\implies$ limits the search space for witness paths
One Key Idea

Bidirectedness $\implies$ Boundedness

If $u$ reaches $v$ then there is a witness path where the counters are bounded.

$\implies$ limits the search space for witness paths

If we had just 1 counter (instead of 2):

$\text{Cnt}_1(H) > 0$  \hspace{2cm} $\text{Cnt}_1(C) < 0$

With 2 counters, more involved, gives $O(n^2)$ bound instead
One Key Idea

**Bidirectedness \( \implies \) Boundedness**

If \( u \) reaches \( v \) then there is a witness path where the counters are bounded.

\[ \implies \text{limits the search space for witness paths} \]

If we had just 1 counter (instead of 2):

\[ \text{Cnt}_1(H) > 0 \quad \text{Cnt}_1(C) < 0 \]

With 2 counters, more involved, gives \( O(n^2) \) bound instead.
One Key Idea

Bidirectedness $\Rightarrow$ Boundedness

If $u$ reaches $v$ then there is a witness path where the counters are bounded.

$\Rightarrow$ limits the search space for witness paths

If we had just 1 counter (instead of 2):

\[ \text{Cnt}_1(H) > 0 \quad \text{Cnt}_1(C) < 0 \]

With 2 counters, more involved, gives $O(n^2)$ bound instead
One Key Idea

Bidirectedness $\implies$ Boundedness

If $u$ reaches $v$ then there is a witness path where the counters are bounded.

$\implies$ limits the search space for witness paths

If we had just 1 counter (instead of 2):

$$\text{Cnt}_1(H) > 0$$

$$\text{Cnt}_1(C) < 0$$

With 2 counters, more involved, gives $O(n^2)$ bound instead
Using the $O(n^2)$ Counter Bound
Using the $O(n^2)$ Counter Bound
Using the $O(n^2)$ Counter Bound


Total time $O(n^3 \alpha(n))$
Using the $O(n^2)$ Counter Bound

\[ D_1 \circ D_1 \text{ on } G \]
with $n$ nodes

\[ D_1 \text{ on } G' \]
with $n^3$ nodes


Total time $O(n^3 \alpha(n))$
Using the $O(n^2)$ Counter Bound

$\mathcal{D}_1 \odot \mathcal{D}_1$ on $G$
with $n$ nodes

$\mathcal{D}_1$ on $G'$
with $n^3$ nodes


Total time $O(n^3 \alpha(n))$
The Counter Bound $O(n^2)$ is Tight

Both counters reach a quadratic value
Bidirected $D_k \circ D_k$

Theorem

- Bidirected formalisms of context + field sensitivity are undecidable.
- Need coarser approximations.
- Or just techniques that work well in practice.
Bidirected $\mathcal{D}_k \odot \mathcal{D}_k$

**Theorem**

*Bidirected $\mathcal{D}_k \odot \mathcal{D}_k$ reachability is **undecidable**.*
Bidirected $\mathcal{D}_k \odot \mathcal{D}_k$

Theorem

Bidirected $\mathcal{D}_k \odot \mathcal{D}_k$ reachability is **undecidable**.

- Even bidirected formalisms of context + field sensitivity are undecidable
- Need coarser approximations
- Or just techniques that work well in practice
Undecidability - Sketch

Directed

\[
\begin{align*}
\alpha_2 & \quad \beta_1 \\
\alpha_1 & \quad \bar{\alpha}_1 \\
\bar{\beta}_1 & \quad \bar{\alpha}_2
\end{align*}
\]
Undecidability - Sketch

Directed

Bidirected

Stack 1

Stack 2
Undecidability - Sketch

Directed

\[
\begin{align*}
\beta_1 & \quad & \alpha_1 & \quad & \overline{\alpha}_1 \\
\alpha_2 & \quad & \overline{\beta}_1 & \quad & \overline{\alpha}_2
\end{align*}
\]

Bidirected

\[
\begin{align*}
(u, y), \epsilon & \quad & (x, u), \overline{\beta}_1 \\
(y, y), \beta_1 & \quad & (y, v), \epsilon \\
(v, v), \epsilon & \quad & (v, v), \epsilon \\
(u, x), \epsilon & \quad & (u, y), \alpha_2 \\
(u, x), \alpha_1 & \quad & (v, v), \overline{\alpha}_1 \\
(y, y), \epsilon & \quad & (y, v), \overline{\alpha}_2 \\
(y, v), \epsilon & \quad & (y, v), \overline{\alpha}_2 \quad & \overline{\nu}
\end{align*}
\]
Undecidability - Sketch

Directed

\[
\begin{align*}
&\alpha_1 : \{u, x\} \\
&\beta_1 : \{y, v\} \\
&\alpha_2 : \{y\} \\
&\bar{\alpha}_1 : \{v\} \\
&\bar{\beta}_1 : \{y\}
\end{align*}
\]

Bidirected

\[
\begin{align*}
&(u, y), \epsilon \\
&(y, y), \beta_1 \\
&(y, v), \epsilon \\
&(v, v), \epsilon \\
&(u, x), \epsilon
\end{align*}
\]
Undecidability - Sketch

Directed

\[
\begin{array}{c}
u \\
\downarrow \alpha_2 \\
y \\
\downarrow \bar{\beta}_1 \\
v \\
\end{array}
\]
\[
\begin{array}{c}
\downarrow \alpha_1 \\
x \\
\end{array}
\]
\[
\begin{array}{c}
\beta_1 \\
\end{array}
\]

Bidirected

\[
\begin{array}{c}
\bar{s} \\
\end{array}
\]
\[
\begin{array}{c}
(x, u), \bar{\beta}_1 \\
(u, y), \epsilon \\
(y, y), \beta_1 \\
(y, v), \epsilon \\
v, v), \epsilon \\
\end{array}
\]
\[
\begin{array}{c}
(u, x), \epsilon \\
(u, \bar{u}), \epsilon \\
\bar{u}, x), \alpha_1 \\
(u, y), \alpha_2 \\
(y, v), \bar{\alpha}_2 \\
(y, \bar{y}), \epsilon \\
\end{array}
\]
\[
\begin{array}{c}
\bar{v} \\
t \\
\end{array}
\]
Undecidability - Sketch

Directed

Bidirected

Stack 1

Stack 2
Undecidability - Sketch

Directed

\[ u \xrightarrow{\alpha_1} x \xleftarrow{\beta_1} \]
\[ y \xrightarrow{\alpha_2} v \xleftarrow{\bar{\alpha}_2} \]
\[ y \xrightarrow{\bar{\beta}_1} \]

Bidirected

\[ (u, y), \epsilon \]
\[ (x, u), \bar{\beta}_1 \]
\[ (y, y), \beta_1 \]
\[ (y, v), \epsilon \]
\[ (v, v), \epsilon \]
\[ (\bar{x}, u), \epsilon \]
\[ (u, x), \alpha_1 \]
\[ (u, y), \alpha_2 \]
\[ (v, v), \bar{\alpha}_1 \]
\[ (y, y), \epsilon \]
\[ (y, v), \bar{\alpha}_2 \]

Stack 1

\[ (y, y) \]
\[ (y, v) \]

Stack 2

\[ \beta_1 \]
### Undecidability - Sketch

#### Directed

![Directed Graph](image)

- Edges: $\alpha_1, \beta_1, \alpha_2, \beta_1\overline{\alpha}_1, \overline{\beta}_1$

<table>
<thead>
<tr>
<th>Stack 1</th>
<th>Stack 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$(u, y)$</td>
<td></td>
</tr>
<tr>
<td>$(y, y)$</td>
<td></td>
</tr>
<tr>
<td>$(y, v)$</td>
<td>$\beta_1$</td>
</tr>
</tbody>
</table>

#### Bidirected

![Bidirected Graph](image)

- Edges: $(u, y), \epsilon, (x, u), \overline{\beta}_1, (y, y), \beta_1, (v, v), \epsilon, (v, v), \epsilon, (u, x), \epsilon, (\overline{x}, u), \epsilon, (\overline{u}, x), \alpha_1, (\overline{u}, y), \alpha_2, (v, v), \overline{\alpha}_1, (y, v), \overline{\alpha}_2, (y, y), \epsilon$

- Vertices: $s, u, x, y, v, t$

- Stack 2
  - $\beta_1$
Undecidability - Sketch

Directed

\[
\begin{align*}
&\quad u \\
&\quad \quad \alpha_1 \\
&\quad \quad \uparrow \\
&\quad x \\
&\quad \quad \beta_1 \\
&\quad \quad \downarrow \\
&\quad y \\
&\quad \quad \alpha_2 \\
&\quad \quad \downarrow \\
&\quad v \\
&\quad \quad \overline{\alpha}_2 \\
&\quad \quad \downarrow \\
&\quad \quad \overline{\beta}_1 \\
\end{align*}
\]

Stack 1

<table>
<thead>
<tr>
<th>(x, u)</th>
<th>(u, y)</th>
<th>(y, y)</th>
<th>(y, v)</th>
</tr>
</thead>
</table>

Stack 2

Bidirected

\[
\begin{align*}
&(u, y), \epsilon \\
&(x, u), \overline{\beta}_1 \\
&(y, y), \beta_1 \\
&(y, v), \epsilon \\
&(v, v), \epsilon \\
&(u, x), \epsilon \\
&(\overline{x}, u), \epsilon \\
&(u, x), \alpha_1 \\
&(u, y), \alpha_2 \\
&(v, v), \overline{\alpha}_1 \\
&(y, y), \epsilon \\
&(y, v), \overline{\alpha}_2 \\
\end{align*}
\]
Undecidability - Sketch

Directed

Bidirected

Stack 1

Stack 2

\[
\begin{array}{c}
(x, u) \\
(u, y) \\
(y, y) \\
(y, v) \\
\end{array}
\]

\[
\begin{array}{c}
\end{array}
\]

\[
\begin{array}{c}
\nu
\end{array}
\]
Undecidability - Sketch

Directed

Bidirected

Stack 1

Stack 2
Undecidability - Sketch

Directed

Bidirected

Stack 1

Stack 2
Undecidability - Sketch

Directed

\[
\begin{align*}
& u \quad \alpha_1 \quad x \\
& y \quad \alpha_2 \quad v \\
& \beta_1 \\
& \overline{\beta}_1
\end{align*}
\]

Bidirected

\[
\begin{align*}
&(u, y), \epsilon \\
&(x, u), \overline{\beta}_1 \\
&(y, y), \beta_1 \\
&(y, v), \epsilon \\
&(v, v), \epsilon \\
&(s) \\
&\nu \\
&(\overline{x}, u), \epsilon \\
&(u, \overline{x}), \alpha_1 \\
&(u, y), \alpha_2 \\
&(v, v), \overline{\alpha}_1 \\
&(y, \overline{y}), \epsilon \\
&(\overline{y}, v), \overline{\alpha}_2 \\
&\overline{\nu}
\end{align*}
\]

\begin{align*}
\text{Stack 1} & \quad \text{Stack 2} \\
(y, y) & \quad \alpha_2 \\
(y, v) & \quad \nu
\end{align*}
Undecidability - Sketch

Directed

\[
\begin{align*}
&\quad \beta_1 \\
&\alpha_1 \quad \alpha_1 \\
&\quad \beta_1 \\
&\alpha_2 \quad \alpha_2 \\
&\quad \beta_1 \\
&\alpha_2 \quad \alpha_2 \\
&\quad \beta_1 \\
&\alpha_2 \quad \alpha_2 \\
\end{align*}
\]

Stack 1

\[(y, \nu)\]

Stack 2

\[\nu\]

Bidirected

\[
\begin{align*}
&(u, y), \epsilon \\
&(x, u), \bar{\beta}_1 \\
&(y, y), \beta_1 \\
&(y, \nu), \epsilon \\
&(\nu, \nu), \epsilon \\
&(v, v), \epsilon \\
&(\nu, \nu), \epsilon \\
&(x, u), \epsilon \\
&(u, x), \epsilon \\
&(u, y), \alpha_2 \\
&(u, x), \alpha_1 \\
&(u, \nu), \alpha_2 \\
&(v, \nu), \bar{\alpha}_1 \\
&(y, \nu), \epsilon \\
&(y, y), \epsilon \\
&(y, \nu), \bar{\alpha}_2 \\
&(y, y), \epsilon \\
&(y, v), \epsilon \\
&(v, v), \epsilon \\
\end{align*}
\]

Stack 1

\[(y, \nu)\]

Stack 2

\[\nu\]
Undecidability - Sketch

Directed

\[ \alpha_1 \] \quad \beta_1 \quad \alpha_2 \quad \overline{\alpha}_1 \quad \overline{\beta}_1

Bidirected

\[ (u, y), \epsilon \]
\[ (x, u), \overline{\beta}_1 \]
\[ (y, y), \beta_1 \]
\[ (y, v), \epsilon \]
\[ (v, v), \epsilon \]
\[ \nu \]
\[ (\overline{x}, u), \epsilon \]
\[ (\overline{u}, x), \alpha_1 \]
\[ (\overline{u}, y), \alpha_2 \]
\[ (y, v), \overline{\alpha}_2 \]
\[ (y, y), \epsilon \]
\[ (\overline{y}, y), \epsilon \]
\[ (y, v), \overline{\alpha}_2 \]
\[ \nu \]

Stack 1
\[ \nu \]

Stack 2
Undecidability - Sketch

Directed

\[ u \xrightarrow{\alpha_1} x, \quad x \xrightarrow{\alpha_1} y, \quad y \xrightarrow{\beta_1} u \]

\[ y \xrightarrow{\alpha_2} v, \quad v \xrightarrow{\beta_1} y \]

Bidirected

\[ (u, y), \epsilon \]

\[ (x, u), \overline{\beta_1} \]

\[ (y, y), \beta_1 \]

\[ (v, v), \epsilon \]

\[ (u, x), \epsilon \]

\[ (\overline{u}, u), \epsilon \]

\[ (\overline{u}, x), \alpha_1 \]

\[ (u, y), \alpha_2 \]

\[ (v, v), \overline{\alpha_1} \]

\[ (y, y), \epsilon \]

\[ (y, v), \overline{\alpha_2} \]

Stack 1

Stack 2
Experiments

• DaCapo benchmarks
• (field + context)-sensitive alias analysis → Bidirected $D_k \circ D_k$ reachability
• $D_1 \circ D_1$ and bounded $D_k \circ D_1$ reachability by abstracting on one language

Summary
• Previously $D_1 \circ D_1$ took more than 2 days
• Now the whole dataset takes $\sim 5$ mins on a laptop for each case $D_1 \circ D_1$ and $D_k \circ D_1$
• Times are usable!
Experiments

Setup

- DaCapo benchmarks
- (field + context)-sensitive alias analysis → Bidirected $D_k \circ D_k$ reachability
- $D_1 \circ D_1$ and bounded $D_k \circ D_1$ reachability by abstracting on one language

Summary

- Previously $D_1 \circ D_1$ took more than 2 days
- Now the whole dataset takes $\sim 5$ mins on a laptop for each case $D_1 \circ D_1$ and $D_k \circ D_1$ reachability
- Times are usable!
Experiments

Setup

- DaCapo benchmarks
- (field + context)-sensitive alias analysis → Bidirected $D_k \odot D_k$ reachability
- $D_1 \odot D_1$ and bounded $D_k \odot D_1$ reachability by abstracting on one language

Summary

- Previously $D_1 \odot D_1$ took more than 2 days
- Now the whole dataset takes $\sim 5$ mins on a laptop for each case $D_1 \odot D_1$ and $D_k \odot D_1$
- Times are usable!
Thank you!

Appendix
### Experiments

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>( n )</th>
<th>( D_1 \odot D_1 )</th>
<th>( D_k \odot D_1 ) (bounded counter)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ID-CCs</td>
<td>Time (s)</td>
<td>ID-CCs</td>
</tr>
<tr>
<td>antlr</td>
<td>29831</td>
<td>26793</td>
<td>40.5</td>
</tr>
<tr>
<td>bloat</td>
<td>36181</td>
<td>32693</td>
<td>16.2</td>
</tr>
<tr>
<td>chart</td>
<td>67535</td>
<td>60787</td>
<td>32.8</td>
</tr>
<tr>
<td>eclipse</td>
<td>30981</td>
<td>27812</td>
<td>16.0</td>
</tr>
<tr>
<td>fop</td>
<td>61016</td>
<td>54671</td>
<td>29.8</td>
</tr>
<tr>
<td>hsqldb</td>
<td>27494</td>
<td>24584</td>
<td>15.5</td>
</tr>
<tr>
<td>jython</td>
<td>36162</td>
<td>31811</td>
<td>26.3</td>
</tr>
<tr>
<td>luindex</td>
<td>28595</td>
<td>25610</td>
<td>15.6</td>
</tr>
<tr>
<td>lusearch</td>
<td>29530</td>
<td>26417</td>
<td>17.8</td>
</tr>
<tr>
<td>pmd</td>
<td>31333</td>
<td>28064</td>
<td>18.2</td>
</tr>
<tr>
<td>xalan</td>
<td>27358</td>
<td>24498</td>
<td>15.2</td>
</tr>
</tbody>
</table>
1. Double-self-loops

2. Trimming

3. CFL underapproximation

1. Treat $D_k \circ D_k$ as one Dyck language over the union alphabet
2. Perform reachability and collapse components
3. Solve $D_k \circ D_k$ on the quotient graph
Coverability

$u$ covers $(v, k)$ if $u$ can reach $v$ with an empty stack and counter at least $k$
Coverability

$u$ covers $(v, k)$ if $u$ can reach $v$ with an empty stack and counter at least $k$.

Theorem (LST '15)

Coverability in PVASS is decidable.

1. $u$ covers $(v, 0)$ and $v$ covers $(u, 0)$?
1. $u$ covers $(v, 0)$ and $v$ covers $(u, 0)$
1. $u$ covers $(v, 0)$ and $v$ covers $(u, 0)$ ✓
2. $u$ covers $(v, 1)$ and $v$ covers $(u, 1)$?
Algorithm

1. $u$ covers $(v, 0)$ and $v$ covers $(u, 0)$ ✓
2. $u$ covers $(v, 1)$ and $v$ covers $(u, 1)$ ✓
\( D_k \odot D_1 \) Algorithm

1. \( u \) covers \((v, 0)\) and \( v \) covers \((u, 0)\) \(\checkmark\)
2. \( u \) covers \((v, 1)\) and \( v \) covers \((u, 1)\) \(\checkmark\)
3. Derive a stack height bound = \( \max(n^2, \text{height of coverability witnesses}) \)
1. $u$ covers $(v, 0)$ and $v$ covers $(u, 0)$ ✓
2. $u$ covers $(v, 1)$ and $v$ covers $(u, 1)$ ✓
3. Derive a stack height bound $= \max(n^2, \text{height of coverability witnesses})$
1. $u$ covers $(v, 0)$ and $v$ covers $(u, 0)$ ✓
2. $u$ covers $(v, 1)$ and $v$ covers $(u, 1)$ ✓
3. Derive a stack height bound $= \max(n^2, \text{height of coverability witnesses})$
$D_k \otimes D_1$ Algorithm

1. $u$ covers $(v, 0)$ and $v$ covers $(u, 0)$ ✓
2. $u$ covers $(v, 1)$ and $v$ covers $(u, 1)$ ✓
3. Derive a stack height bound $= \max(n^2, \text{height of coverability witnesses})$
1. $u$ covers $(v, 0)$ and $v$ covers $(u, 0)$ ✓
2. $u$ covers $(v, 1)$ and $v$ covers $(u, 1)$ ✓
3. Derive a stack height bound $= \max(n^2, \text{height of coverability witnesses})$

$C_u \circ P \circ C_v$
1. $u$ covers $(v, 0)$ and $v$ covers $(u, 0)$ ✓
2. $u$ covers $(v, 1)$ and $v$ covers $(u, 1)$ ✓
3. Derive a stack height bound $= \max(n^2, \text{height of coverability witnesses})$

$$C_u \circ P \circ C_v \circ \overline{P}$$
1. $u$ covers $(v, 0)$ and $v$ covers $(u, 0)$ ✓
2. $u$ covers $(v, 1)$ and $v$ covers $(u, 1)$ ✓
3. Derive a stack height bound $= \max(n^2, \text{height of coverability witnesses})$
Algorithm \( \mathcal{D}_k \odot \mathcal{D}_1 \)

1. \( u \) covers \((v, 0)\) and \( v \) covers \((u, 0)\)

2. \( u \) covers \((v, 1)\) and \( v \) covers \((u, 1)\)

3. Derive a stack height bound \( = \max(n^2, \text{height of coverability witnesses}) \)

\[
\begin{align*}
&\quad \sum_{u} + \quad C_u \\
\quad u \quad \text{arrow} \quad \text{arrow} \quad \text{arrow} \\
\quad \text{arrow} \quad \text{arrow} \quad \text{arrow} \\
\quad P, \text{ stack height } \leq n^2 \quad \text{arrow} \\
\quad \text{arrow} \quad \text{arrow} \quad \text{arrow} \\
&\quad \sum_{v} + \quad C_v
\end{align*}
\]

\[
C_u^* \circ P \circ C_v^* \circ \overline{P} \circ \overline{C_u^*} \circ P
\]
1. $u$ covers $(v, 0)$ and $v$ covers $(u, 0)$ ✓
2. $u$ covers $(v, 1)$ and $v$ covers $(u, 1)$ ✓
3. Derive a stack height bound $= \max(n^2, \text{height of coverability witnesses})$