Inference of Linear Upper-Bounds on the Expected Cost by Solving Cost Relations

Alicia Merayo and Samir Genaim

Complutense University of Madrid

Oxford, 07/19/2018

Overview

Cost Analysis

Cost Analysis

Application of static techniques in order to determine the amount of resources that are needed to execute a program.

- Runtime (execution steps)
- Memory consumption
- Generic cost model: ticks

Cost Analysis

Cost Analysis

Application of static techniques in order to determine the amount of resources that are needed to execute a program.

- Runtime (execution steps)
- Memory consumption
- Generic cost model: ticks

Cost Analysis

- **Runtime**: Sort

```
for (i = 0; i < length(a); i++){
  val = a[i];
  j = i-1;
  while(j > 0 && a[j] > val){
    a[j+1] = a[j];
    j = j-1;
    tick(1);
  }
  a[j+1] = val;
  tick(1);
}
```

- **Worst case cost**:

$$\frac{\text{length}(a)^2 + \text{length}(a)}{2}$$

- **Memory**: Dynamic Array

```
while(! empty(list)){
  if list[i] > 0{
    if (j>=n){
      enlarge(a,l,n);
      tick(2*l*n);
    }
    a[j] = head(list);
    list = tail(list);
    j = j+1;
  }
  i = i+1;
}
```

- **Worst case cost**:

$$2 * l * n * (length(list))$$

Cost Analysis

- **Runtime**: Sort

```
for (i = 0; i < length(a); i++){
  val = a[i];
  j = i-1;
  while(j > 0 && a[j] > val){
    a[j+1] = a[j];
    j = j-1;
    tick(1);
  }
  a[j+1] = val;
  tick(1);
}
```

- **Worst case cost**:

$$\frac{\text{length}(a)^2 + \text{length}(a)}{2}$$

- **Memory**: Dynamic Array

```
while(! empty(list)){
  if list[i] > 0{
    if (j>=n){
      enlarge(a,l,n);
      tick(2*l*n);
    }
    a[j] = head(list);
    list = tail(list);
    j = j+1;
  }
  i = i+1;
}
```

- **Worst case cost**:

$$2 * l * n * (length(list))$$

Cost Analysis

- **Runtime**: Sort

```
for (i = 0; i < length(a); i++){
  val = a[i];
  j = i-1;
  while(j > 0 && a[j] > val){
    a[j+1] = a[j];
    j = j-1;
    tick(1);
  }
  a[j+1] = val;
  tick(1);
}
```

- **Worst case cost**:

$$\frac{\text{length}(a)^2 + \text{length}(a)}{2}$$

- **Memory**: Dynamic Array

```
while(! empty(list)){
  if list[i] > 0{
    if (j>=n){
      enlarge(a,l,n);
      tick(2*l*n);
    }
    a[j] = head(list);
    list = tail(list);
    j = j+1;
  }
  i = i+1;
}
```

- **Worst case cost**:

$$2 * l * n * (length(list))$$

Cost Analysis

- **Runtime**: Sort

```
for (i = 0; i < length(a); i++){
  val = a[i];
  j = i-1;
  while(j > 0 && a[j] > val){
    a[j+1] = a[j];
    j = j-1;
    tick(1);
  }
  a[j+1] = val;
  tick(1);
}
```

- **Worst case cost**:

$$\frac{\text{length}(a)^2 + \text{length}(a)}{2}$$

- **Memory**: Dynamic Array

```
while(! empty(list)){
  if list[i] > 0{
    if (j>=n){
      enlarge(a,l,n);
      tick(2*l*n);
    }
    a[j] = head(list);
    list = tail(list);
    j = j+1;
  }
  i = i+1;
}
```

- **Worst case cost**:

$$2 * l * n * (length(list))$$

SACO

SACO

Static analyzer for automatically inferring upper/lower bounds on the worst/best-case resource consumption of ABS programs.
`http://costa.fdi.ucm.es/saco/web/`

ABS

- Functional + imperative.
- Based on concurrent objects.

SACO

SACO

Static analyzer for automatically inferring upper/lower bounds on the worst/best-case resource consumption of ABS programs.
`http://costa.fdi.ucm.es/saco/web/`

ABS

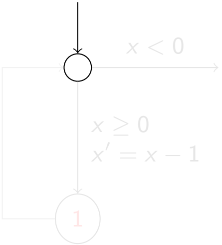
- Functional + imperative.
- Based on concurrent objects.

Cost Analysis using SACO

```
while(x>=0){
  x = x-1;
  tick(1);
}
```

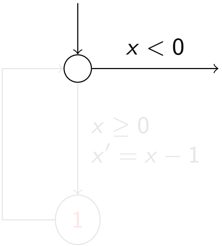
Cost Analysis using SACO

```
while(x>=0){
  x = x-1;
  tick(1);
}
```



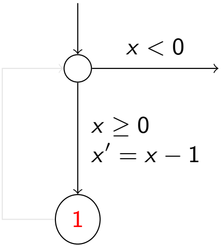
Cost Analysis using SACO

```
while(x>=0){
  x = x-1;
  tick(1);
}
```



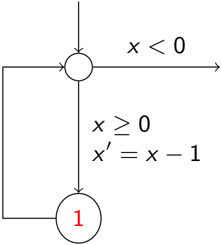
Cost Analysis using SACO

```
while(x>=0){
  x = x-1;
  tick(1);
}
```

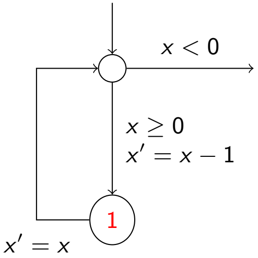


Cost Analysis using SACO

```
while(x>=0){
  x = x-1;
  tick(1);
}
```



Cost Analysis using SACO



- **From CFG to Cost Relations**

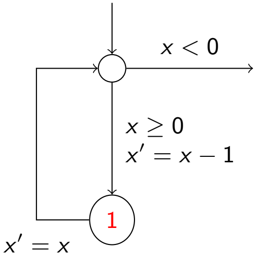
$$
\begin{array}{ll}
C(x) = 0 & \{x < 0\} \\
C(x) = D(x') & \{x >= 0, x' = x - 1\} \\
D(x) = 1 + C(x') & \{x' = x\}
\end{array}
$$

- **Simplifying cost relations**

$$
\begin{array}{ll}
C(x) = 0 & \{x < 0\} \\
C(x) = 1 + C(x') & \{x \geq 0,\ x' = x - 1\}
\end{array}
$$

Cost Analysis using SACO

- **From CFG to Cost Relations**

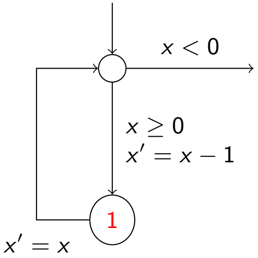


$$
\begin{array}{ll}
C(x) = 0 & \{x < 0\} \\
C(x) = D(x') & \{x >= 0, x' = x - 1\} \\
D(x) = 1 + C(x') & \{x' = x\}
\end{array}
$$

- **Simplifying cost relations**

$$
\begin{array}{ll}
C(x) = 0 & \{x < 0\} \\
C(x) = 1 + C(x') & \{x \geq 0,\ x' = x - 1\}
\end{array}
$$

Cost Analysis using SACO



- **From CFG to Cost Relations**

$$\begin{array}{ll} C(x) = 0 & \{x < 0\} \\ C(x) = D(x') & \{x >= 0, x' = x - 1\} \\ D(x) = 1 + C(x') & \{x' = x\} \end{array}$$

- **Simplifying cost relations**

$$\begin{array}{ll} C(x) = 0 & \{x < 0\} \\ C(x) = 1 + C(x') & \{x \geq 0,\ x' = x - 1\} \end{array}$$

Cost Analysis using SACO

Cost relations are like recurrence relations, but the have some difference like nondeterminism.

```
while(x>=0){
  x = x-1;
  tick(1);
}
```

```
while(x>=0){
  if(*)
    x = x-1;
  else
    x = x-2;
  tick(1);
}
```

$$\begin{array}{ll} C(x) = 0 & \{x < 0\} \\ C(x) = 1 + C(x') & \{x \geq 0,\ x' = x - 1\} \end{array}$$

$$\begin{array}{ll} C(x) = 0 & \{x < 0\} \\ C(x) = 1 + C(x') & \{x \geq 0,\ x' = x - 1\} \\ C(x) = 1 + C(x') & \{x \geq 0,\ x' = x - 2\} \end{array}$$

Cost Analysis using SACO

Cost relations are like recurrence relations, but the have some difference like nondeterminism.

```
while(x>=0){
  x = x-1;
  tick(1);
}
```

```
while(x>=0){
  if(*)
    x = x-1;
  else
    x = x-2;
  tick(1);
}
```

$$
\begin{array}{ll}
C(x) = 0 & \{x < 0\} \\
C(x) = 1 + C(x') & \{x \geq 0,\ x' = x - 1\}
\end{array}
$$

$$
\begin{array}{ll}
C(x) = 0 & \{x < 0\} \\
C(x) = 1 + C(x') & \{x \geq 0,\ x' = x - 1\} \\
C(x) = 1 + C(x') & \{x \geq 0,\ x' = x - 2\}
\end{array}
$$

Cost Analysis using SACO

Upper Bound of a Cost Relation

Upper-bound on the set of values to which a query might evaluate. It is an upper bound of the worst case of the execution.

```
while(x>=0){
  if(*)
    x = x-1;
  else
    x = x-2;
  tick(1);
}
```

$$
\begin{array}{ll}
C(x) = 0 & \{x < 0\} \\
C(x) = 1 + C(x') & \{x \geq 0,\ x' = x - 1\} \\
C(x) = 1 + C(x') & \{x \geq 0,\ x' = x - 2\}
\end{array}
$$

$$C(10) = 10, 9, 8, \ldots$$

Cost Analysis using SACO

Upper Bound of a Cost Relation

Upper-bound on the set of values to which a query might evaluate. It is an upper bound of the worst case of the execution.

```
while(x>=0){
  if(*)
    x = x-1;
  else
    x = x-2;
  tick(1);
}
```

$$
\begin{array}{ll}
C(x) = 0 & \{x < 0\} \\
C(x) = 1 + C(x') & \{x \geq 0,\ x' = x - 1\} \\
C(x) = 1 + C(x') & \{x \geq 0,\ x' = x - 2\}
\end{array}
$$

$$C(10) = 10, 9, 8, \ldots$$

Cost Analysis using SACO

Upper Bound of a Cost Relation

Upper-bound on the set of values to which a query might evaluate. It is an upper bound of the worst case of the execution.

```
while(x>=0){
  if(*)
    x = x-1;
  else
    x = x-2;
  tick(1);
}
```

$$
\begin{array}{ll}
C(x) = 0 & \{x < 0\} \\
C(x) = 1 + C(x') & \{x \geq 0,\ x' = x - 1\} \\
C(x) = 1 + C(x') & \{x \geq 0,\ x' = x - 2\}
\end{array}
$$

$$C(10) = 10, 9, 8, \ldots$$

When worst-case is not enough...

```
while(msgs != null){
  if(send(head(data))){
    msgs = tail(msgs);
  }
  tick(1);
}
```

$$
\begin{array}{ll}
C(n) = 0 & \{n \leq 0\} \\
C(n) = 1 + C(n') & \{n \geq 1, n' = n - 1\} \\
C(n) = 1 + C(n') & \{n \geq 1, n' = n\}
\end{array}
$$

- Infinite complexity
- What happens if we trust the quality of the network to some degree?

When worst-case is not enough...

```
while(msgs != null){
  if(send(head(data))){
    msgs = tail(msgs);
  }
  tick(1);
}
```

$$
\begin{array}{ll}
C(n) = 0 & \{n \leq 0\} \\
C(n) = 1 + C(n') & \{n \geq 1, n' = n - 1\} \\
C(n) = 1 + C(n') & \{n \geq 1, n' = n\}
\end{array}
$$

- Infinite complexity
- What happens if we trust the quality of the network to some degree?

When worst-case is not enough...

```
while(msgs != null){
  if(send(head(data))){
    msgs = tail(msgs);
  }
  tick(1);
}
```

$$
\begin{array}{ll}
C(n) = 0 & \{n \leq 0\} \\
C(n) = 1 + C(n') & \{n \geq 1, n' = n - 1\} \\
C(n) = 1 + C(n') & \{n \geq 1, n' = n\}
\end{array}
$$

- Infinite complexity
- What happens if we trust the quality of the network to some degree?

When worst-case is not enough...

```
while(msgs != null){
  if(send(head(data))){
    msgs = tail(msgs);
  }
  tick(1);
}
```

$$
\begin{array}{ll}
C(n) = 0 & \{n \leq 0\} \\
C(n) = 1 + C(n') & \{n \geq 1, n' = n - 1\} \\
C(n) = 1 + C(n') & \{n \geq 1, n' = n\}
\end{array}
$$

- Infinite complexity
- What happens if we trust the quality of the network to some degree?

Probabilistic programs

Addition of probabilistic choices to the set of instructions:

$$\text{inst}_1 \oplus_p \text{inst}_2$$

where

- $p \in [0, 1]$
- inst_1 and inst_2 will execute with probability p and $1 - p$, respectively.
- It can be generalized such that inst_i is executed with probability p_i, and $\sum_i p_i = 1$

```
while(x < y){
  x = x − 1 ⊕_{1/3} y = y − 2;
  tick(1);
}
```

Probabilistic programs

Addition of probabilistic choices to the set of instructions:

$$\text{inst}_1 \oplus_p \text{inst}_2$$

where

- $p \in [0, 1]$
- inst_1 and inst_2 will execute with probability p and $1 - p$, respectively.
- It can be generalized such that inst_i is executed with probability p_i, and $\sum_i p_i = 1$

```
while(x < y){
  x = x − 1 ⊕_{1/3} y = y − 2;
  tick(1);
}
```

Probabilistic programs

Addition of probabilistic choices to the set of instructions:

$$\text{inst}_1 \oplus_p \text{inst}_2$$

where

- $p \in [0, 1]$
- inst_1 and inst_2 will execute with probability p and $1 - p$, respectively.
- It can be generalized such that inst_i is executed with probability p_i, and $\sum_i p_i = 1$

while$(x < y)\{$

$x = x - 1 \oplus_{\frac{1}{3}} y = y - 2;$

tick(1);

$\}$

Probabilistic programs

Addition of probabilistic choices to the set of instructions:

$$\text{inst}_1 \oplus_p \text{inst}_2$$

where

- $p \in [0, 1]$
- inst_1 and inst_2 will execute with probability p and $1 - p$, respectively.
- It can be generalized such that inst_i is executed with probability p_i, and $\sum_i p_i = 1$

$$\begin{array}{l} \text{while}(x < y)\{ \\ \quad x = x - 1 \oplus_{\frac{1}{3}} y = y - 2; \\ \quad \text{tick}(1); \\ \} \end{array}$$

Expected cost

Packages transmission on a net

We have got a net for transmitting packages on which the probability of succeeding is $\frac{2}{3}$, and we want to transmit n packages.

```
while(msgs != null){
  if(send(head(data)))
    msgs = tail(msgs);
  tick(1);
}
```

$\Longrightarrow$

$$
\begin{array}{l}
\text{while } (n > 0)\{ \\
\quad n = n - 1; \ \oplus_{\frac{2}{3}} \text{ skip}; \\
\quad \text{tick}(1); \\
\}
\end{array}
$$

Expected cost

Packages transmission on a net

We have got a net for transmitting packages on which the probability of succeeding is $\frac{2}{3}$, and we want to transmit n packages.

$$
\begin{array}{l}
\text{while}(\text{msgs} \;! = \;\text{null})\{ \\
\quad \text{if}(\text{send}(\text{head}(\text{data}))) \\
\quad\quad \text{msgs} = \text{tail}(\text{msgs}); \\
\quad \text{tick}(1); \\
\}
\end{array}
\quad \Longrightarrow \quad
\begin{array}{l}
\text{while } (n > 0)\{ \\
\quad n = n - 1; \;\oplus_{\frac{2}{3}} \text{ skip}; \\
\quad \text{tick}(1); \\
\}
\end{array}
$$

Expected cost

Packages transmission on a net

We have got a net for transmitting packages on which the probability of succeeding is $\frac{2}{3}$, and we want to transmit n packages.

$$\begin{array}{l} \text{while}(\text{msgs} \mathrel{!}= \text{ null})\{ \\ \quad \text{if}(\text{send}(\text{head}(\text{data}))) \\ \quad\quad \text{msgs} = \text{tail}(\text{msgs}); \\ \quad \text{tick}(1); \\ \} \end{array} \quad \Longrightarrow \quad \begin{array}{l} \text{while } (n > 0)\{ \\ \quad n = n - 1; \ \oplus_{\frac{2}{3}} \text{ skip}; \\ \quad \text{tick}(1); \\ \} \end{array}$$

Expected cost

Expected cost

Let

- S be an infinite sequence of independent choices for the probabilistic operation.
- $\texttt{cost}(n, S)$ be the cost of the execution (with input n) when taking the choices indicated in S.
- $\texttt{Pr}(S)$ be the probability of taking such choices

then the *expected cost* is defined as

$$\sum_{S} \texttt{Pr}(S) \cdot \texttt{cost}(n, S)$$

Expected cost

Expected cost

Let

- S be an infinite sequence of independent choices for the probabilistic operation.
- $\texttt{cost}(n, S)$ be the cost of the execution (with input n) when taking the choices indicated in S.
- $\texttt{Pr}(S)$ be the probability of taking such choices

then the *expected cost* is defined as

$$\sum_{S} \texttt{Pr}(S) \cdot \texttt{cost}(n, S)$$

Expected cost

Expected cost

Let

- S be an infinite sequence of independent choices for the probabilistic operation.
- $\texttt{cost}(n, S)$ be the cost of the execution (with input n) when taking the choices indicated in S.
- $\texttt{Pr}(S)$ be the probability of taking such choices

then the *expected cost* is defined as

$$\sum_{S} \texttt{Pr}(S) \cdot \texttt{cost}(n, S)$$

Expected cost

Expected cost

Let

- S be an infinite sequence of independent choices for the probabilistic operation.
- $\texttt{cost}(n, S)$ be the cost of the execution (with input n) when taking the choices indicated in S.
- $\texttt{Pr}(S)$ be the probability of taking such choices

then the *expected cost* is defined as

$$\sum_S \texttt{Pr}(S) \cdot \texttt{cost}(n, S)$$

Expected cost

Expected cost

Let

- S be an infinite sequence of independent choices for the probabilistic operation.
- $\mathtt{cost}(n, S)$ be the cost of the execution (with input n) when taking the choices indicated in S.
- $\mathtt{Pr}(S)$ be the probability of taking such choices

then the *expected cost* is defined as

$$\sum_{S} \mathtt{Pr}(S) \cdot \mathtt{cost}(n, S)$$

Expected cost

$$\sum_{S} \mathtt{Pr}(S) \cdot \mathtt{cost}(n, S)$$

The sum is over all possible sequences S.

- $\sum_S \mathtt{Pr}(S) = 1$

while $(n > 0)\{$
  $n = n - 1;\ \oplus_{\frac{2}{3}}$ skip;
  tick(1);
$\}$

Expected cost

$$\sum_{S} \texttt{Pr}(S) \cdot \texttt{cost}(n, S)$$

- The sum is over all possible sequences S.
- $\sum_S \texttt{Pr}(S) = 1$

```
while (n > 0){
   n = n − 1; ⊕_{2/3} skip;
   tick(1);
}
```

Expected cost

$$\sum_{S} \mathtt{Pr}(S) \cdot \mathtt{cost}(n, S)$$

- The sum is over all possible sequences S.
- $\sum_S \mathtt{Pr}(S) = 1$

while $(n > 0)\{$
  $n = n - 1;\ \oplus_{\frac{2}{3}}$ skip;
  tick(1);
$\}$

Expected cost

$$\sum_{S} \mathtt{Pr}(S) \cdot \mathtt{cost}(n, S)$$

- The sum is over all possible sequences S.
- $\sum_{S} \mathtt{Pr}(S) = 1$

while $(n > 0)\{$
  $n = n - 1;\ \oplus_{\frac{2}{3}}$ skip;
  tick(1);
$\}$

Expected cost by CRs

```
while (n > 0){
   n = n − 1; ⊕_{2/3} skip;
   tick(1);
}
```

By definition, the expected cost is

$$\sum_{S} \mathtt{Pr}(S) \cdot \mathtt{cost}(n, S)$$

CRs:

$$C(n) = 0 \qquad \{n \leq 0\}$$

$$C(n) = 1 + \frac{2}{3}C(n') + \frac{1}{3}C(n'') \qquad \{n \geq 1, n' = n - 1, n'' = n\}$$

Expected cost: $\frac{3}{2}n$

Expected cost by CRs

$$
\begin{array}{l}
\text{while } (n > 0)\{ \\
\quad n = n - 1; \ \oplus_{\frac{2}{3}} \ \text{skip}; \\
\quad \text{tick}(1); \\
\}
\end{array}
$$

By definition, the expected cost is

$$\sum_{S} \mathtt{Pr}(S) \cdot \mathtt{cost}(n, S)$$

CRs:

$$C(n) = 0 \qquad \{n \leq 0\}$$

$$C(n) = 1 + \frac{2}{3} C(n') + \frac{1}{3} C(n'') \qquad \{n \geq 1, n' = n - 1, n'' = n\}$$

Expected cost: $\frac{3}{2} n$

Expected cost by CRs

$$
\begin{array}{l}
\text{while } (n > 0)\{ \\
\quad n = n - 1; \ \oplus_{\frac{2}{3}} \ \text{skip}; \\
\quad \text{tick}(1); \\
\}
\end{array}
$$

By definition, the expected cost is

$$\sum_{S} \mathtt{Pr}(S) \cdot \mathtt{cost}(n, S)$$

CRs:

$$
\begin{array}{ll}
C(n) = 0 & \{n \leq 0\} \\
C(n) = 1 + \frac{2}{3}C(n') + \frac{1}{3}C(n'') & \{n \geq 1, n' = n - 1, n'' = n\}
\end{array}
$$

Expected cost: $\frac{3}{2}n$

Expected cost by CRs

```
while (n > 0){
   n = n − 1; ⊕_{2/3} skip;
   tick(1);
}
```

By definition, the expected cost is

$$\sum_{S} \mathtt{Pr}(S) \cdot \mathtt{cost}(n, S)$$

CRs:

$$C(n) = 0 \qquad \{n \leq 0\}$$

$$C(n) = 1 + \frac{2}{3}C(n') + \frac{1}{3}C(n'') \qquad \{n \geq 1, n' = n - 1, n'' = n\}$$

Expected cost: $\frac{3}{2}n$

Our goal

We want to introduce probabilities into SACO.

Steps

- Introduction of probabilities into CFGs.
- Development of techniques for inferring expected cost of CFGs via CRs.
- Specify probabilities directly in ABS.

Our goal

We want to introduce probabilities into SACO.

Steps

- Introduction of probabilities into CFGs.
- Development of techniques for inferring expected cost of CFGs via CRs.
- Specify probabilities directly in ABS.

Our goal

We want to introduce probabilities into SACO.

Steps

- Introduction of probabilities into CFGs.
- Development of techniques for inferring expected cost of CFGs via CRs.
- Specify probabilities directly in ABS.

Our goal

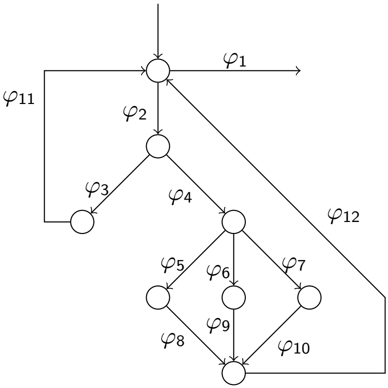
We want to introduce probabilities into SACO.

Steps

- Introduction of probabilities into CFGs.
- Development of techniques for inferring expected cost of CFGs via CRs.
- Specify probabilities directly in ABS.

CFGs with probabilities

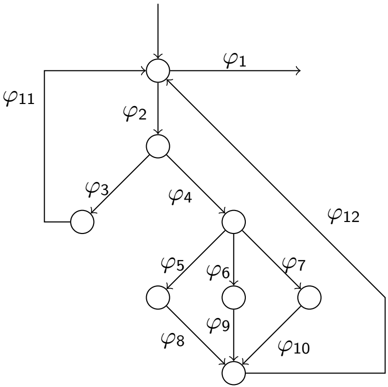
CFG without probabilities:



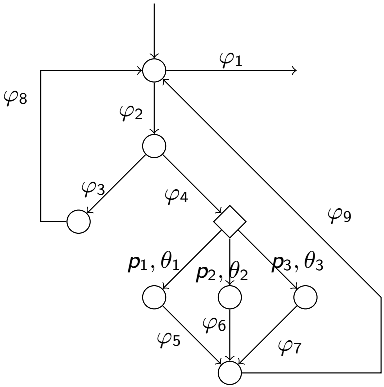
CFG with probabilities:

CFGs with probabilities

CFG without probabilities:

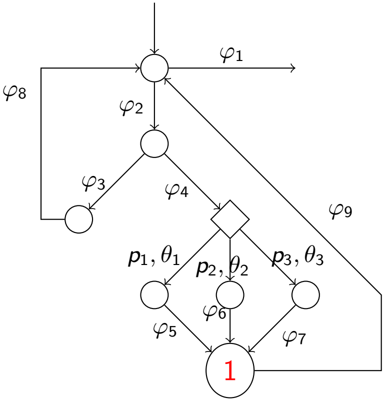


CFG with probabilities:



CFGs with probabilities

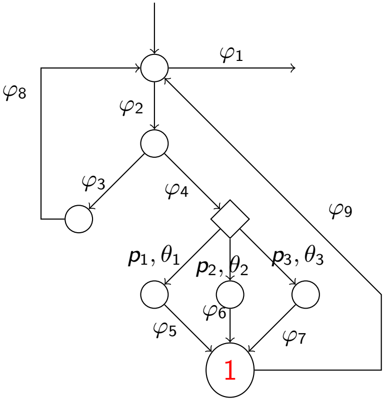
CFG with probabilities:



- Regular instructions represented by circles.
- Probability instructions represented by squares.
- Edges with updates and
  - Probabilities
  - Constraints
- Nodes with cost.
- $\sum_i p_i = 1$
- Choices in normal nodes cover the whole state space.

CFGs with probabilities

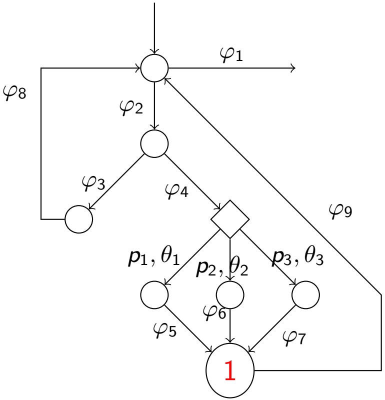
CFG with probabilities:



- Regular instructions represented by circles.
- Probability instructions represented by squares.
- Edges with updates and
  - Probabilities
  - Constraints
- Nodes with cost.
- $\sum_i p_i = 1$
- Choices in normal nodes cover the whole state space.

CFGs with probabilities

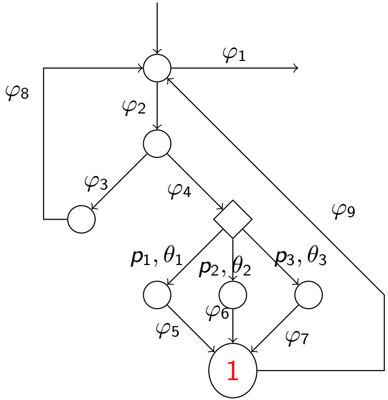
CFG with probabilities:



- Regular instructions represented by circles.
- Probability instructions represented by squares.
- Edges with updates and
  - Probabilities
  - Constraints
- Nodes with cost.
- $\sum_i p_i = 1$
- Choices in normal nodes cover the whole state space.

CFGs with probabilities

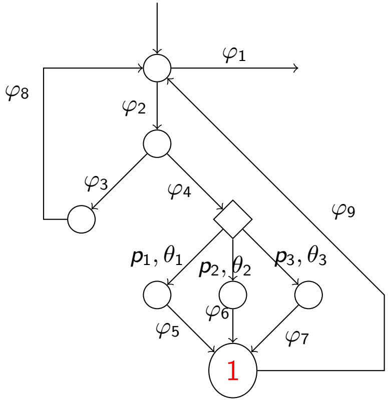
CFG with probabilities:



- Regular instructions represented by circles.
- Probability instructions represented by squares.
- Edges with updates and
  - Probabilities
  - Constraints
- Nodes with cost.
- $\sum_i p_i = 1$
- Choices in normal nodes cover the whole state space.

CFGs with probabilities

CFG with probabilities:



- Regular instructions represented by circles.
- Probability instructions represented by squares.
- Edges with updates and
  - Probabilities
  - Constraints
- Nodes with cost.
- $\sum_i p_i = 1$
- Choices in normal nodes cover the whole state space.

CFGs with probabilities

```
while (n > 0){
   n = n − 1; ⊕_{2/3} skip;
   tick(1);
}
```

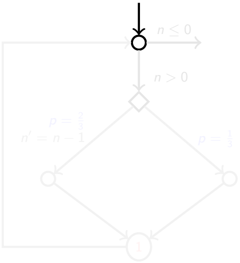
CFGs with probabilities

while $(n > 0)\{$

$n = n - 1;\ \oplus_{\frac{2}{3}}$ skip;

tick(1);

$\}$



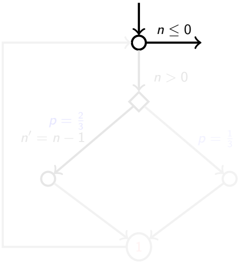
CFGs with probabilities

while $(n > 0)\{$

$n = n - 1;\ \oplus_{\frac{2}{3}}$ skip;

tick(1);

$\}$



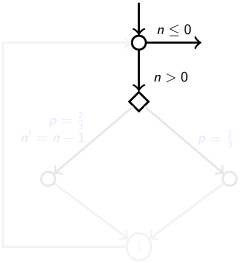
CFGs with probabilities

while $(n > 0)\{$

$n = n - 1;\ \oplus_{\frac{2}{3}}$ skip;

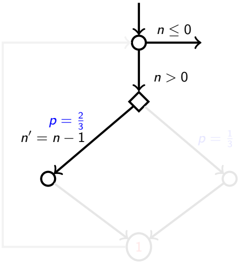
tick(1);

$\}$



CFGs with probabilities

while ($n > 0$){
  $n = n - 1;\ \oplus_{\frac{2}{3}}$ skip;
  tick(1);
}



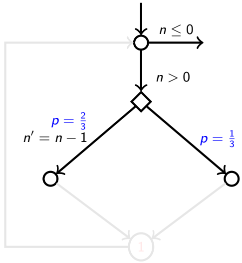
CFGs with probabilities

while $(n > 0)\{$

$\quad n = n - 1; \ \oplus_{\frac{2}{3}}$ skip;

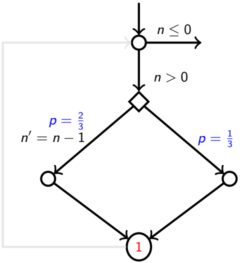
$\quad$ tick(1);

$\}$



CFGs with probabilities

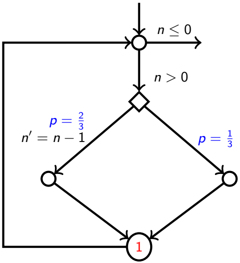
while $(n > 0)\{$
  $n = n - 1; \oplus_{\frac{2}{3}}$ skip;
  tick(1);
$\}$



CFGs with probabilities

```
while (n > 0){
   n = n − 1; ⊕_{2/3} skip;
   tick(1);
}
```

$$\text{while } (n > 0)\{ \quad n = n - 1; \ \oplus_{\frac{2}{3}} \ \text{skip}; \quad \text{tick}(1); \quad \}$$



Generating CRs

- Each regular node $n \in N_r$ with outgoing transitions $n \xrightarrow{\varphi_i} n_i$, $1 \leq i \leq k$, contributes k cost relations of the form

$$C_n(\bar{x}) = \texttt{cost}(n) + C_{n_i}(\bar{x}') \qquad \varphi_i$$

- Each probabilistic node $n \in N_p$ with outgoing edges $n \xrightarrow{p_i,\theta_i} n_i$, for $1 \leq i \leq k$, contributes one cost relation

$$C_n(\bar{x}) = \texttt{cost}(n) + \sum_{i=1}^{k} p_i \cdot C_{n_i}(\bar{x}_i') \qquad \biguplus_{i=1}^{k} \theta_i$$

Generating CRs

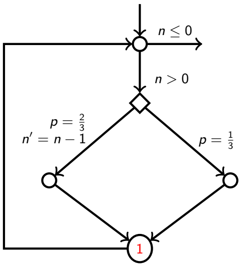
- Each regular node $n \in N_r$ with outgoing transitions $n \xrightarrow{\varphi_i} n_i$, $1 \leq i \leq k$, contributes k cost relations of the form

$$C_n(\bar{x}) = \texttt{cost}(n) + C_{n_i}(\bar{x}') \qquad\qquad \varphi_i$$

- Each probabilistic node $n \in N_p$ with outgoing edges $n \xrightarrow{p_i,\theta_i} n_i$, for $1 \leq i \leq k$, contributes one cost relation

$$C_n(\bar{x}) = \texttt{cost}(n) + \sum_{i=1}^{k} p_i \cdot C_{n_i}(\bar{x}_i') \qquad\qquad \biguplus_{i=1}^{k} \theta_i$$

Generating CRs

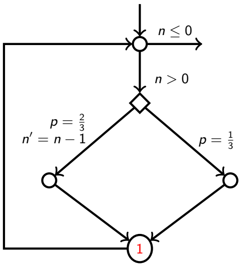


$$
\begin{array}{ll}
C(n) = 0 & \{n \le 0\} \\
C(n) = 1 + \dfrac{2}{3}D(n') + \dfrac{1}{3}D(n'') & \left\{n > 0,\ n' = n - 1,\ n'' = n\right\} \\
D(n) = C(n') & \left\{n' = n\right\}
\end{array}
$$

$$\Downarrow$$

$$
\begin{array}{ll}
C(n) = 0 & \{n \le 0\} \\
C(n) = 1 + \dfrac{2}{3}C(n') + \dfrac{1}{3}C(n'') & \left\{n > 0,\ n' = n - 1,\ n'' = n\right\}
\end{array}
$$

Generating CRs

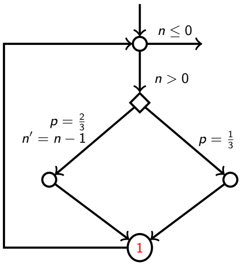


$$
\begin{array}{ll}
C(n) = 0 & \{n \leq 0\} \\
C(n) = 1 + \frac{2}{3} D(n') + \frac{1}{3} D(n'') & \left\{n > 0,\ n' = n - 1,\ n'' = n\right\} \\
D(n) = C(n') & \left\{n' = n\right\}
\end{array}
$$

$$\Downarrow$$

$$
\begin{array}{ll}
C(n) = 0 & \{n \leq 0\} \\
C(n) = 1 + \frac{2}{3} C(n') + \frac{1}{3} C(n'') & \left\{n > 0,\ n' = n - 1,\ n'' = n\right\}
\end{array}
$$

Generating CRs



$$
\begin{array}{ll}
C(n) = 0 & \{n \leq 0\} \\
C(n) = 1 + \dfrac{2}{3}D(n') + \dfrac{1}{3}D(n'') & \left\{n > 0,\ n' = n - 1,\ n'' = n\right\} \\
D(n) = C(n') & \left\{n' = n\right\}
\end{array}
$$

$$\Downarrow$$

$$
\begin{array}{ll}
C(n) = 0 & \{n \leq 0\} \\
C(n) = 1 + \dfrac{2}{3}C(n') + \dfrac{1}{3}C(n'') & \left\{n > 0,\ n' = n - 1,\ n'' = n\right\}
\end{array}
$$

Solving (probabilistic) CRs

$$C(n) = 0 \qquad \underbrace{\{n \leq 0\}}_{\varphi_1}$$

$$C(n) = 1 + \frac{2}{3}C(n') + \frac{1}{3}C(n'') \qquad \underbrace{\{n > 0,\ n' = n - 1,\ n'' = n\}}_{\varphi_2}$$

$$\forall n, \varphi_1 \implies f(n) \geq 0$$

$$\forall n, n', n'', \ \varphi_2 \implies f(n) \geq 1 + \frac{2}{3}f(n') + \frac{1}{3}f(n'')$$

Solving (probabilistic) CRs

$$C(n) = 0 \qquad \underbrace{\{n \leq 0\}}_{\varphi_1}$$

$$C(n) = 1 + \frac{2}{3}C(n') + \frac{1}{3}C(n'') \qquad \underbrace{\{n > 0,\ n' = n - 1,\ n'' = n\}}_{\varphi_2}$$

$$\forall n,\ \varphi_1 \implies f(n) \geq 0$$

$$\forall n, n', n'',\ \varphi_2 \implies f(n) \geq 1 + \frac{2}{3}f(n') + \frac{1}{3}f(n'')$$

Solving (probabilistic) CRs

$$\forall n,\ \varphi_1 \implies f(n) \geq 0$$
$$\forall n, n', n'',\ \varphi_2 \implies f(n) \geq 1 + \frac{2}{3} f(n') + \frac{1}{3} f(n'')$$

$$f(n) = a_1 n + a_0$$
$$\downarrow$$

$$\forall n,\ \varphi_1 \implies a_1 n + a_0 \geq 0$$
$$\forall n, n', n'',\ \varphi_2 \implies a_1 n + a_0 \geq 1 + \frac{2}{3}(a_1 n' + a_0) + \frac{1}{3}(a_1 n + a_0)$$

- We use Farkas Lemma to get constraints in terms of a_i from these equations.
- Solving these equations will give us the values for a_i.

Solving (probabilistic) CRs

$$\forall n,\ \varphi_1 \implies f(n) \geq 0$$
$$\forall n, n', n'',\ \varphi_2 \implies f(n) \geq 1 + \frac{2}{3} f(n') + \frac{1}{3} f(n'')$$

$$f(n) = a_1 n + a_0$$
$$\downarrow$$

$$\forall n,\ \varphi_1 \implies a_1 n + a_0 \geq 0$$
$$\forall n, n', n'',\ \varphi_2 \implies a_1 n + a_0 \geq 1 + \frac{2}{3}(a_1 n' + a_0) + \frac{1}{3}(a_1 n + a_0)$$

- We use Farkas Lemma to get constraints in terms of a_i from these equations.
- Solving these equations will give us the values for a_i.

Solving (probabilistic) CRs

$$\forall n,\ \varphi_1 \implies f(n) \geq 0$$
$$\forall n, n', n'',\ \varphi_2 \implies f(n) \geq 1 + \frac{2}{3} f(n') + \frac{1}{3} f(n'')$$

$$f(n) = a_1 n + a_0$$
$$\downarrow$$

$$\forall n,\ \varphi_1 \implies a_1 n + a_0 \geq 0$$
$$\forall n, n', n'',\ \varphi_2 \implies a_1 n + a_0 \geq 1 + \frac{2}{3}\left(a_1 n' + a_0\right) + \frac{1}{3}\left(a_1 n + a_0\right)$$

- We use Farkas Lemma to get constraints in terms of a_i from these equations.
- Solving these equations will give us the values for a_i.

Solving (probabilistic) CRs

$$\begin{aligned} &\forall n,\ \varphi_1 \implies f(n) \geq 0 \\ &\forall n, n', n'',\ \varphi_2 \implies f(n) \geq 1 + \frac{2}{3} f(n') + \frac{1}{3} f(n'') \end{aligned}$$

$$f(n) = a_1 n + a_0$$
$$\downarrow$$

$$\begin{aligned} &\forall n,\ \varphi_1 \implies a_1 n + a_0 \geq 0 \\ &\forall n, n', n'',\ \varphi_2 \implies a_1 n + a_0 \geq 1 + \frac{2}{3}\left(a_1 n' + a_0\right) + \frac{1}{3}\left(a_1 n + a_0\right) \end{aligned}$$

- We use Farkas Lemma to get constraints in terms of a_i from these equations.
- Solving these equations will give us the values for a_i.

Solving (probabilistic) CRs

$$\forall n,\ \varphi_1 \implies f(n) \geq 0$$
$$\forall n, n', n'',\ \varphi_2 \implies f(n) \geq 1 + \frac{2}{3} f(n') + \frac{1}{3} f(n'')$$

$$f(n) = a_1 n + a_0$$
$$\downarrow$$

$$\forall n,\ \varphi_1 \implies a_1 n + a_0 \geq 0$$
$$\forall n, n', n'',\ \varphi_2 \implies a_1 n + a_0 \geq 1 + \frac{2}{3}\left(a_1 n' + a_0\right) + \frac{1}{3}\left(a_1 n + a_0\right)$$

- We use Farkas Lemma to get constraints in terms of a_i from these equations.
- Solving these equations will give us the values for a_i.

Solving (probabilistic) CRs

Upper bound functions must be non-negative.

- We introduce *max* function.
- If we denote

$$F(n) = \max(0, a_1 n + a_0)$$

then we will have

$$\begin{aligned} \forall n,\ \varphi_1 &\implies F(n) \geq 0 \\ \forall n, n', n'',\ \varphi_2 &\implies F(n) \geq 1 + \frac{2}{3}F(n') + \frac{1}{3}F(n) \end{aligned}$$

- Since we are covering the whole state space with the constraints, we have to introduce *max* on both sides of the constraints.

Solving (probabilistic) CRs

Upper bound functions must be non-negative.

- We introduce *max* function.
- If we denote

$$F(n) = \max(0, a_1 n + a_0)$$

then we will have

$$\forall n,\ \varphi_1 \implies F(n) \geq 0$$
$$\forall n, n', n'',\ \varphi_2 \implies F(n) \geq 1 + \frac{2}{3}F(n') + \frac{1}{3}F(n)$$

- Since we are covering the whole state space with the constraints, we have to introduce *max* on both sides of the constraints.

Solving (probabilistic) CRs

$$\forall n,\ \varphi_1,\ a_1 n + a_0 < 0 \implies 0 \geq 0$$
$$\forall n,\ \varphi_1,\ a_1 n + a_0 \geq 0 \implies a_1 n + a_0 \geq 0$$

$$\forall n, n'\ \varphi_2,\ a_1 n + a_0 < 0,\ a_1 n' + a_0 \geq 0$$
$$\implies 0 \geq 1 + \frac{2}{3}(a_1 n' + a_0)$$
$$\forall n, n'\ \varphi_2,\ a_1 n + a_0 \geq 0,\ a_1 n' + a_0 < 0$$
$$\implies a_1 n + a_0 \geq 1 + \frac{1}{3}(a_1 n + a_0)$$
$$\forall n, n'\ \varphi_2,\ a_1 n + a_0 \geq 0,\ a_1 n' + a_0 \geq 0$$
$$\implies a_1 n + a_0 \geq 1 + \frac{2}{3}(a_1 n' + a_0) + \frac{1}{3}(a_1 n + a_0)$$

- Contraints on the left side of the implication lead us to non-linearity.
- We are in trouble with applying Farkas Lemma.

Solving (probabilistic) CRs

$$
\begin{array}{l}
\forall n,\ \varphi_1,\ a_1 n + a_0 < 0 \implies 0 \geq 0 \\
\forall n,\ \varphi_1,\ a_1 n + a_0 \geq 0 \implies a_1 n + a_0 \geq 0
\end{array}
$$

$$
\begin{array}{l}
\forall n, n'\ \varphi_2,\ a_1 n + a_0 < 0,\ a_1 n' + a_0 \geq 0 \\
\qquad \implies 0 \geq 1 + \frac{2}{3}(a_1 n' + a_0) \\
\forall n, n'\ \varphi_2,\ a_1 n + a_0 \geq 0,\ a_1 n' + a_0 < 0 \\
\qquad \implies a_1 n + a_0 \geq 1 + \frac{1}{3}(a_1 n + a_0) \\
\forall n, n'\ \varphi_2,\ a_1 n + a_0 \geq 0,\ a_1 n' + a_0 \geq 0 \\
\qquad \implies a_1 n + a_0 \geq 1 + \frac{2}{3}(a_1 n' + a_0) + \frac{1}{3}(a_1 n + a_0)
\end{array}
$$

- Contraints on the left side of the implication lead us to non-linearity.
- We are in trouble with applying Farkas Lemma.

Solving (probabilistic) CRs

$$\forall n,\ \varphi_1,\ a_1 n + a_0 < 0 \implies 0 \geq 0$$
$$\forall n,\ \varphi_1,\ a_1 n + a_0 \geq 0 \implies a_1 n + a_0 \geq 0$$

$$\forall n, n'\ \varphi_2,\ a_1 n + a_0 < 0,\ a_1 n' + a_0 \geq 0$$
$$\implies 0 \geq 1 + \frac{2}{3}(a_1 n' + a_0)$$
$$\forall n, n'\ \varphi_2,\ a_1 n + a_0 \geq 0,\ a_1 n' + a_0 < 0$$
$$\implies a_1 n + a_0 \geq 1 + \frac{1}{3}(a_1 n + a_0)$$
$$\forall n, n'\ \varphi_2,\ a_1 n + a_0 \geq 0,\ a_1 n' + a_0 \geq 0$$
$$\implies a_1 n + a_0 \geq 1 + \frac{2}{3}(a_1 n' + a_0) + \frac{1}{3}(a_1 n + a_0)$$

- Contraints on the left side of the implication lead us to non-linearity.
- We are in trouble with applying Farkas Lemma.

Example: infinite expected cost

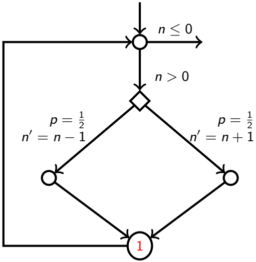
while $(n > 0)\{$

$\quad n = n - 1; \oplus_{\frac{1}{2}} n = n + 1;$

$\}$

Example: infinite expected cost

$$
\begin{array}{l}
\text{while } (n > 0)\{ \\
\quad n = n - 1; \ \oplus_{\frac{1}{2}} \ n = n + 1; \\
\}
\end{array}
$$



Multiphase example

```
while (x + 3 ≤ n){
   if (y < m)
      skip; ⊕_{1/2} y = y + 1;
   else
      skip; ⊕_{1/4} x = x + 1; ⊕_{1/4} x = x + 2; ⊕_{1/4} x = x + 3;
   tick(1);
}
```

$$f(x, y, n, m) = \max(0, a_1 x + a_2 y + a_3 n + a_4 m + a_0)$$

If we use this global template... we will fail.

Multiphase example

$$
\begin{array}{l}
\text{while } (x+3 \leq n)\{ \\
\quad \text{if } (y < m) \\
\quad\quad \text{skip}; \oplus_{\frac{1}{2}} y = y+1; \\
\quad \text{else} \\
\quad\quad \text{skip}; \oplus_{\frac{1}{4}} x = x+1; \oplus_{\frac{1}{4}} x = x+2; \oplus_{\frac{1}{4}} x = x+3; \\
\quad \text{tick}(1); \\
\}
\end{array}
$$

$$f(x,y,n,m) = \max(0, a_1x + a_2y + a_3n + a_4m + a_0)$$

If we use this global template... we will fail.

Multiphase example

$$
\begin{array}{l}
\text{while } (x + 3 \leq n)\{ \\
\quad \text{if } (y < m) \\
\quad\quad \text{skip}; \oplus_{\frac{1}{2}} y = y + 1; \\
\quad \text{else} \\
\quad\quad \text{skip}; \oplus_{\frac{1}{4}} x = x + 1; \oplus_{\frac{1}{4}} x = x + 2; \oplus_{\frac{1}{4}} x = x + 3; \\
\quad \text{tick}(1); \\
\}
\end{array}
$$

$$f(x, y, n, m) = \max(0, a_1 x + a_2 y + a_3 n + a_4 m + a_0)$$

If we use this global template... we will fail.

Multiphase example

```
while (x + 3 ≤ n){
   if (y < m)
      skip; ⊕_{1/2} y = y + 1;
   else
      skip; ⊕_{1/4} x = x + 1; ⊕_{1/4} x = x + 2; ⊕_{1/4} x = x + 3;
   tick(1);
}
```

We can start from a template that includes two parts (one for each phase).

$f(x, y, n, m) = \max(0, a_1 x + a_2 y + a_3 n + a_4 m + a_0) + \max(0, b_1 x + b_2 y + b_3 n + b_4 m + b_0)$

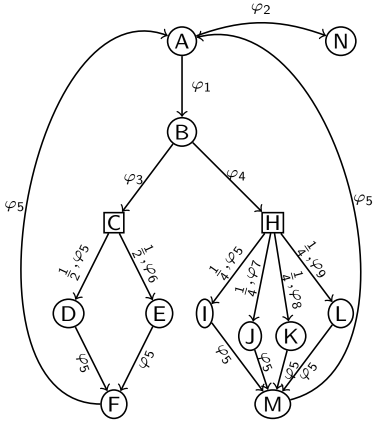
Multiphase example

$$
\begin{array}{l}
\text{while } (x + 3 \leq n)\{ \\
\quad \text{if } (y < m) \\
\quad\quad \text{skip}; \oplus_{\frac{1}{2}} y = y + 1; \\
\quad \text{else} \\
\quad\quad \text{skip}; \oplus_{\frac{1}{4}} x = x + 1; \oplus_{\frac{1}{4}} x = x + 2; \oplus_{\frac{1}{4}} x = x + 3; \\
\quad \text{tick}(1); \\
\}
\end{array}
$$

We can start from a template that includes two parts (one for each phase).

$$f(x, y, n, m) = \max(0, a_1 x + a_2 y + a_3 n + a_4 m + a_0) + \max(0, b_1 x + b_2 y + b_3 n + b_4 m + b_0)$$

Multiphase example



$$
\begin{aligned}
\varphi_1 &= \{x + 3 \leq n\} \cup \varphi_5 \\
\varphi_2 &= \{x + 3 > n\} \cup \varphi_5 \\
\varphi_3 &= \{y < m\} \cup \varphi_5 \\
\varphi_4 &= \{y' \geq m\} \cup \varphi_5 \\
\varphi_5 &= \{x' = x, n' = n, y' = y, m' = m\} \\
\varphi_6 &= \{x' = x, n' = n, y' = y + 1, m' = m\} \\
\varphi_7 &= \{x' = x + 1, n' = n, y' = y, m' = m\} \\
\varphi_8 &= \{x' = x + 2, n' = n, y' = y, m' = m\} \\
\varphi_9 &= \{x' = x + 3, n' = n, y' = y, m' = m\}
\end{aligned}
$$

Conclusions

- Our interest is to extend SACO to handle probabilistic programs
- We have extended the CFGs to have probabilities
- We have modeled the expected cost of CFGs with (probability) CRs.
- We have developed a technique for solving such CRs.
- Correctness: based on Farkas Lemma and Markov Decision Processes.