Refined Resource Analysis Based on Cost Relations

Reiner Hähnle, Antonio Flores-Montoya

Technische Universität Darmstadt (TUD)
Fachbereich Informatik

15th International Workshop on Termination, Obergurgl, Austria

5 September 2016

The Goal

Statically obtain a bound on the resource consumption of a program (amount of resources required to execute it) on any input data

Static Resource Analysis

The Goal

Statically obtain a bound on the resource consumption of a program (amount of resources required to execute it) on any input data

Typical Resources

- Execution steps (execution time)
- Number of visits to a program point
- Amount of memory used

Static Resource Analysis

The Goal

Statically obtain a bound on the resource consumption of a program (amount of resources required to execute it) on any input data

Typical Resources

- Execution steps (execution time)
- Number of visits to a program point
- Amount of memory used

Good Reasons for Having It

- Performance evaluation, avoid performance bugs
- Code documentation and certification
- Real cost estimation in virtual environments, provisioning

Classic Approach to Resource Analysis

Most early work on resource analysis based on extracting and solving recurrence relations

- [Wegbreit, 1975] (Lisp programs)
- [Debray and Lin, 1993] (Logic programs)
- [Vasconcelos and Hammond, 2004] (Functional programs)

Limitations of Approach Based on Recurrence Relations

Extracting as well as solving of recurrence relations from programs is challenging and leads to highly complex and/or sub-optimal solutions

Limitations of Approach Based on Recurrence Relations

Extracting as well as solving of recurrence relations from programs is challenging and leads to highly complex and/or sub-optimal solutions

```
while (i<n) {
 if (r>0) {
  i=rnd(n-1);
  r--;
 } else
  i++;
}
```

Limitations of Approach Based on Recurrence Relations

Extracting as well as solving of recurrence relations from programs is challenging and leads to highly complex and/or sub-optimal solutions

```
      while (i<n) {
        if (r>0) {
p1:       i=rnd(n-1);
p1:       r--;
        } else
p2:       i++;
      }
```

- Multiple paths:
  Systems of recurrence relations?
  Abstract into a worst case path?

Limitations of Approach Based on Recurrence Relations

Extracting as well as solving of recurrence relations from programs is challenging and leads to highly complex and/or sub-optimal solutions

```
     while (i<n) {
      if (r>0) {
p1:     i=rnd(n-1);
p1:     r--;
      } else
p2:     i++;
     }
```

- Multiple paths:
  Systems of recurrence relations?
  Abstract into a worst case path?
- Non-determinism:
  Abstract to the worst case? What is the worst case?
  Cost does not have monotonic behavior!

Limitations of Approach Based on Recurrence Relations

Extracting as well as solving of recurrence relations from programs is challenging and leads to highly complex and/or sub-optimal solutions

```
     while (i<n) {
      if (r>0) {
p1:     i=rnd(n-1);
p1:     r--;
      } else
p2:     i++;
     }
```

Cost depends on i, n, r

- Multiple paths:
  Systems of recurrence relations?
  Abstract into a worst case path?
- Non-determinism:
  Abstract to the worst case? What is the worst case?
  Cost does not have monotonic behavior!
- Cost can depend on multiple variables:
  Pre-analysis, counter instrumentation, . . .

Extract and solve cost relations instead [Albert et al., 2007]

Cost Relations

Extract and solve cost relations instead [Albert et al., 2007]

```
while (i<n) {
 if (r>0) {
  i=rnd(n-1);
  r--;
 } else
  i++;
}
```

$1 : wh(i, n, r) = 0 \quad \{i \geq n\}$

$2 : wh(i, n, r) = 1 + wh(i', n, r') \; \{i < n,\ r > 0,\ 0 \leq i' < n,\ r' = r - 1\}$

$3 : wh(i, n, r) = 1 + wh(i', n, r) \; \{i < n,\ r \leq 0,\ i' = i + 1\}$

Cost Relations

Extract and solve cost relations instead [Albert et al., 2007]

```
while (i<n) {
 if (r>0) {
  i=rnd(n-1);
  r--;
 } else
  i++;
}
```

$1 : wh(i, n, r) = 0 \quad \{i \geq n\}$

$2 : wh(i, n, r) = 1 + wh(i', n, r') \; \{i < n,\ r > 0,\ 0 \leq i' < n,\ r' = r - 1\}$

$3 : wh(i, n, r) = 1 + wh(i', n, r) \; \{i < n,\ r \leq 0,\ i' = i + 1\}$

- Cost relations similar to recurrence relations with linear constraints

Cost Relations

Extract and solve cost relations instead [Albert et al., 2007]

```
p1:while (i<n) {
    if (r>0) {
      i=rnd(n-1);
      r--;
    } else
      i++;
  }
```

$$\mathbf{1} : wh(i,n,r) = 0 \quad \{i \geq n\}$$

$$2 : wh(i,n,r) = 1 + wh(i',n,r') \ \{i < n,\ r > 0,\ 0 \leq i' < n,\ r' = r - 1\}$$

$$3 : wh(i,n,r) = 1 + wh(i',n,r) \ \{i < n,\ r \leq 0,\ i' = i + 1\}$$

- Cost relations similar to recurrence relations with linear constraints
- One cost equation per loop path

Cost Relations

Extract and solve cost relations instead [Albert et al., 2007]

```
p2: while (i<n) {
p2:   if (r>0) {
p2:     i=rnd(n-1);
p2:     r--;
      } else
        i++;
    }
```

$1 : wh(i, n, r) = 0 \quad \{i \geq n\}$

$\mathbf{2} : wh(i, n, r) = 1 + wh(i', n, r')\ \{i < n,\ r > 0,\ 0 \leq i' < n,\ r' = r - 1\}$

$3 : wh(i, n, r) = 1 + wh(i', n, r)\ \{i < n,\ r \leq 0,\ i' = i + 1\}$

- Cost relations similar to recurrence relations with linear constraints
- One cost equation per loop path

Extract and solve cost relations instead [Albert et al., 2007]

```
p3:while (i<n) {
p3:  if (r>0) {
       i=rnd(n-1);
       r--;
     } else
p3:    i++;
   }
```

$$1 : wh(i, n, r) = 0 \quad \{i \geq n\}$$

$$2 : wh(i, n, r) = 1 + wh(i', n, r') \; \{i < n,\ r > 0,\ 0 \leq i' < n,\ r' = r - 1\}$$

$$\mathbf{3} : wh(i, n, r) = 1 + wh(i', n, r) \; \{i < n,\ r \leq 0,\ i' = i + 1\}$$

- Cost relations similar to recurrence relations with linear constraints
- One cost equation per loop path

Extract and solve cost relations instead [Albert et al., 2007]

```
while (i<n) {
 if (r>0) {
  i=rnd(n-1);
  r--;
 } else
  i++;
}
```

$$1 : wh(i,n,r) = \mathbf{0} \quad \{i \geq n\}$$

$$2 : wh(i,n,r) = \mathbf{1} + wh(i',n,r') \; \{i < n,\ r > 0,\ 0 \leq i' < n,\ r' = r-1\}$$

$$3 : wh(i,n,r) = \mathbf{1} + wh(i',n,r) \; \{i < n,\ r \leq 0,\ i' = i+1\}$$

- Cost relations similar to recurrence relations with linear constraints
- One cost equation per loop path
- Each cost equation defines a cost

Cost Relations

Extract and solve cost relations instead [Albert et al., 2007]

```
while (i<n) {
  if (r>0) {
    i=rnd(n−1);
    r−−;
  } else
    i++;
}
```

$$1: wh(i,n,r) = 0 \quad \{\mathbf{i} \geq \mathbf{n}\}$$

$$2: wh(i,n,r) = 1 + wh(i',n,r') \ \{\mathbf{i} < \mathbf{n},\ \mathbf{r} > \mathbf{0},\ 0 \leq i' < n,\ r' = r-1\}$$

$$3: wh(i,n,r) = 1 + wh(i',n,r) \ \{\mathbf{i} < \mathbf{n},\ \mathbf{r} \leq \mathbf{0},\ i' = i+1\}$$

- Cost relations similar to recurrence relations with linear constraints
- One cost equation per loop path
- Each cost equation defines a cost
- The constraints represent:
    - applicability conditions for each cost equation

Cost Relations

Extract and solve cost relations instead [Albert et al., 2007]

```
while (i<n) {
 if (r>0) {
  i=rnd(n-1);
  r--;
 } else
  i++;
}
```

$1 : wh(i, n, r) = 0 \quad \{i \geq n\}$

$2 : wh(i, n, r) = 1 + wh(i', n, r') \; \{i < n,\ r > 0,\ \mathbf{0} \leq \mathbf{i'} < \mathbf{n},\ \mathbf{r'} = \mathbf{r} - \mathbf{1}\}$

$3 : wh(i, n, r) = 1 + wh(i', n, r) \; \{i < n,\ r \leq 0,\ \mathbf{i'} = \mathbf{i} + \mathbf{1}\}$

- Cost relations similar to recurrence relations with linear constraints
- One cost equation per loop path
- Each cost equation defines a cost
- The constraints represent:
  - applicability conditions for each cost equation
  - input/output behavior (possibly non-deterministic)

Extract and solve cost relations instead [Albert et al., 2007]

```
while (i<n) {
 if (r>0 && a[r]) {
  i=rnd(n-1);
  r--;
 } else
  i++;
}
```

CE 2 and 3 can interleave

$1 : wh(i, n, r) = 0 \quad \{i \geq n\}$

$2 : wh(i, n, r) = 1 + wh(i', n, r') \; \{i < n, \; r > 0, \; 0 \leq i' < n, \; r' = r - 1\}$

$3 : wh(i, n, r) = 1 + wh(i', n, r) \; \{i < n, \cancel{r \leq 0}, \; i' = i + 1\}$

- Cost relations similar to recurrence relations with linear constraints
- One cost equation per loop path
- Each cost equation defines a cost
- The constraints represent:
  - applicability conditions for each cost equation
  - input/output behavior (possibly non-deterministic)
- Not necessarily mutually exclusive

Properties of Cost Relations

Cost relations $\approx$ subclass of constraint logic program

- Easy to extract
- Uniform treatment of loops and recursion
- Language independent (frontends for many higher PLs)
- Good for incremental solving (one loop/function at a time)

Properties of Cost Relations

Cost relations $\approx$ subclass of constraint logic program

- Easy to extract
- Uniform treatment of loops and recursion
- Language independent (frontends for many higher PLs)
- Good for incremental solving (one loop/function at a time)

Address main limitations in solving process

- Multi-phase loops
- Loops with resets
- Amortized cost

Two phases

1. Control Flow Refinement
   - Break down the control flow of each cost relation into a set of possible execution patterns (called chains)
   - Prove termination
   - Compute invariants and summaries for each pattern
2. Bound Computation

Overall Approach to Solving Cost Relations

Two phases

1. **Control Flow refinement**
   - Break down the control flow of each cost relation into a set of possible execution patterns (called chains)
   - Prove termination
   - Compute invariants and summaries for each pattern
2. Bound Computation

Control Flow Refinement: Phases and Chains

```
while (i<n) {
 if (r>0) {
  i=rnd(n-1);
  r--;
 } else
  i++;
}
```

$\mathbf{1} : wh(i, n, r) = 0 \quad \{i \geq n\}$

$\mathbf{2} : wh(i, n, r) = 1 + wh(i', n, r') \; \{i < n,\ r > 0,\ 0 \leq i' < n,\ r' = r - 1\}$

$\mathbf{3} : wh(i, n, r) = 1 + wh(i', n, r) \; \{i < n,\ r \leq 0,\ i' = i + 1\}$

Control Flow Refinement: Phases and Chains

```
while (i<n) {
  if (r>0) {
    i=rnd(n-1);
    r--;
  } else
    i++;
}
```

$\mathbf{1} : wh(i,n,r) = 0 \quad \{i \geq n\}$

$\mathbf{2} : wh(i,n,r) = 1 + wh(i',n,r') \ \{i < n,\ r > 0,\ 0 \leq i' < n,\ r' = r - 1\}$

$\mathbf{3} : wh(i,n,r) = 1 + wh(i',n,r) \ \{i < n,\ r \leq 0,\ i' = i + 1\}$

- Generate dependency (call) graph among the cost equations using their constraint sets (e.g., $2 \not\longrightarrow 1$)

$$\begin{tikzcd} 2 \arrow[loop above] \arrow[r] & 3 \arrow[loop above] \arrow[r] & 1 \end{tikzcd}$$

Control Flow Refinement: Phases and Chains

```
while (i<n) {
  if (r>0) {
    i=rnd(n-1);
    r--;
  } else
    i++;
}
```

$\mathbf{1} : wh(i,n,r) = 0 \quad \{i \geq n\}$

$\mathbf{2} : wh(i,n,r) = 1 + wh(i',n,r') \ \{i < n,\ r > 0,\ 0 \leq i' < n,\ r' = r - 1\}$

$\mathbf{3} : wh(i,n,r) = 1 + wh(i',n,r) \ \{i < n,\ r \leq 0,\ i' = i + 1\}$

- Generate dependency (call) graph among the cost equations using their constraint sets (e.g., $2 \not\longrightarrow 1$)
- Enumerate possible execution patterns

$$\begin{tikzcd} 2 \arrow[loop above] \arrow[r] & 3 \arrow[loop above] \arrow[r] & 1 \end{tikzcd}$$

```
while (i<n) {
  if (r>0) {
    i=rnd(n-1);
    r--;
  } else
    i++;
}
```

1 : $wh(i, n, r) = 0 \quad \{i \geq n\}$

2 : $wh(i, n, r) = 1 + wh(i', n, r') \; \{i < n,\ r > 0,\ 0 \leq i' < n,\ r' = r - 1\}$

3 : $wh(i, n, r) = 1 + wh(i', n, r) \; \{i < n,\ r \leq 0,\ i' = i + 1\}$

- Generate dependency (call) graph among the cost equations using their constraint sets (e.g., $2 \not\longrightarrow 1$)
- Enumerate possible execution patterns
  - Each SCC forms a phase
    iterative (e.g., $(2)^+$) or non-iterative (e.g., 1)

$$\begin{tikzcd} 2 \arrow[loop above] \arrow[r] & 3 \arrow[loop above] \arrow[r] & 1 \end{tikzcd}$$

Phases
$(2)^+$
$(3)^+$
1

Control Flow Refinement: Phases and Chains

```
while (i<n) {
  if (r>0) {
    i=rnd(n-1);
    r--;
  } else
    i++;
}
```

$\mathbf{1} : wh(i, n, r) = 0 \quad \{i \geq n\}$

$\mathbf{2} : wh(i, n, r) = 1 + wh(i', n, r') \ \{i < n,\ r > 0,\ 0 \leq i' < n,\ r' = r - 1\}$

$\mathbf{3} : wh(i, n, r) = 1 + wh(i', n, r) \ \{i < n,\ r \leq 0,\ i' = i + 1\}$

- Generate dependency (call) graph among the cost equations using their constraint sets (e.g., $2 \not\longrightarrow 1$)
- Enumerate possible execution patterns
  - Each SCC forms a phase
    iterative (e.g., $(2)^+$) or non-iterative (e.g., 1)
  - Chains are possible sequences of phases in the dependency graph

$$\begin{tikzcd} 2 \arrow[loop above] \arrow[r] & 3 \arrow[loop above] \arrow[r] & 1 \end{tikzcd}$$

Phases
$(2)^+$
$(3)^+$
1

Chains

Non-terminating	Terminating
$(2)^+(3)^+$	$(2)^+(3)^+1$
$(3)^+$	$(3)^+1$
$(2)^+$	1

Control Flow Refinement: Termination

```
while (i<n) {
 if (r>0) {
  i=rnd(n-1);
  r--;
 } else
  i++;
}
```

$\mathbf{1} : wh(i, n, r) = 0 \quad \{i \geq n\}$

$\mathbf{2} : wh(i, n, r) = 1 + wh(i', n, r') \; \{i < n,\ r > 0,\ 0 \leq i' < n,\ r' = r - 1\}$

$\mathbf{3} : wh(i, n, r) = 1 + wh(i', n, r) \; \{i < n,\ r \leq 0,\ i' = i + 1\}$

```
while (i<n) {
 if (r>0) {
  i=rnd(n-1);
  r--;
 } else
  i++;
}
```

$\mathbf{1} : wh(i,n,r) = 0 \quad \{i \geq n\}$

$\mathbf{2} : wh(i,n,r) = 1 + wh(i',n,r') \; \{i < n,\ r > 0,\ 0 \leq i' < n,\ r' = r - 1\}$

$\mathbf{3} : wh(i,n,r) = 1 + wh(i',n,r) \; \{i < n,\ r \leq 0,\ i' = i + 1\}$

- Attempt to prove termination of each iterative phase with (lexicographic) ranking function

Phases	**Ranking function**
$(3)^+$	$n - i$
$(2)^+$	r
1	—

Chains

Non-terminating	Terminating
$(2)^+(3)^+$	$(2)^+(3)^+1$
$(3)^+$	$(3)^+1$
$(2)^+$	1

```
while (i<n) {
 if (r>0) {
  i=rnd(n-1);
  r--;
 } else
  i++;
}
```

$\mathbf{1}: wh(i, n, r) = 0 \quad \{i \geq n\}$

$\mathbf{2}: wh(i, n, r) = 1 + wh(i', n, r') \; \{i < n,\ r > 0,\ 0 \leq i' < n,\ r' = r - 1\}$

$\mathbf{3}: wh(i, n, r) = 1 + wh(i', n, r) \; \{i < n,\ r \leq 0,\ i' = i + 1\}$

- Attempt to prove termination of each iterative phase with (lexicographic) ranking function
- Discard all non-terminating chains ending in a non-terminating phase

Phases	**Ranking function**
$(3)^+$	$n - i$
$(2)^+$	r
1	—

Chains Non-terminating	Terminating
~~$(2)^+(3)^+$~~	$(2)^+(3)^+1$
~~$(3)^+$~~	$(3)^+1$
~~$(2)^+$~~	1

```
while (i<n) {
  if (r>0) {
    i=rnd(n-1);
    r--;
  } else
    i++;
}
```

$$\mathbf{1} : wh(i,n,r) = 0 \quad \{i \geq n\}$$

$$\mathbf{2} : wh(i,n,r) = 1 + wh(i',n,r') \; \{i < n,\ r > 0,\ 0 \leq i' < n,\ r' = r - 1\}$$

$$\mathbf{3} : wh(i,n,r) = 1 + wh(i',n,r) \; \{i < n,\ r \leq 0,\ i' = i + 1\}$$

- Attempt to prove termination of each iterative phase with (lexicographic) ranking function
- Discard all non-terminating chains ending in a non-terminating phase

Phases	**Ranking function**
$(3)^+$	$n - i$
$(2)^+$	r
1	—

Chains

Non-terminating	Terminating
$\cancel{(2)^+(3)^+}$	$(2)^+(3)^+1$
$\cancel{(3)^+}$	$(3)^+1$
$\cancel{(2)^+}$	1

Remaining chains describe all possible behavior

Control Flow Refinement: Invariants and Propagation

Propagate Information Forward and Backward along each Chain

Compute polyhedral invariants

- Backward propagation: necessary precondition for each chain
- Precondition permits further refinement of other cost equations (example later)

Control Flow Refinement: Invariants and Propagation

Propagate Information Forward and Backward along each Chain

Compute polyhedral invariants

- Backward propagation: necessary precondition for each chain
- Precondition permits further refinement of other cost equations (example later)

Chains		**Preconditions**
$(2)^{+}(3)^{+}1$	$\longrightarrow$	$i < n \wedge r > 0$
$(3)^{+}1$	$\longrightarrow$	$i < n \wedge r \leq 0$
1	$\longrightarrow$	$i \geq n$

Overall Approach to Solving Cost Relations

Two phases

1. Control Flow Refinement
   - Break down the control flow of each cost relation into a set of possible execution patterns (called chains)
   - Prove termination
   - Compute invariants and summaries for each pattern
2. Bound Computation

Overall Approach to Solving Cost Relations

Two phases

1. Control Flow Refinement
   - Break down the control flow of each cost relation into a set of possible execution patterns (called chains)
   - Prove termination
   - Compute invariants and summaries for each pattern
2. **Bound Computation**
   - Bottom-up incremental approach
   - Represent cost with data structure called cost structure
   - Can infer and compose costs precisely and efficiently

Chain Evaluation Schema

$\mathbf{1} : wh(i, n, r) = 0 \quad \{i \geq n\}$
$\mathbf{2} : wh(i, n, r) = 1 + wh(i', n, r') \; \{i < n,\ r > 0,\ 0 \leq i' < n,\ r' = r - 1\}$
$\mathbf{3} : wh(i, n, r) = 1 + wh(i', n, r) \; \{i < n,\ r \leq 0,\ i' = i + 1\}$

$$\frac{\text{Chain: } (2)^{+}(3)^{+}1}{wh(i, n, r)}$$

Chain Evaluation Schema

$\mathbf{1}$: $wh(i, n, r) = 0 \quad \{i \geq n\}$
$\mathbf{2}$: $wh(i, n, r) = 1 + wh(i', n, r')\ \{i < n,\ r > 0,\ 0 \leq i' < n,\ r' = r - 1\}$
$\mathbf{3}$: $wh(i, n, r) = 1 + wh(i', n, r)\ \{i < n,\ r \leq 0,\ i' = i + 1\}$

Chain: $(2)^+(3)^+1$

$wh(i, n, r)$

$\searrow$ CE 2

$1 + wh(i_2, n_2, r_2)$

$\mathbf{1} : wh(i, n, r) = 0 \quad \{i \geq n\}$
$\mathbf{2} : wh(i, n, r) = 1 + wh(i', n, r') \; \{i < n,\; r > 0,\; 0 \leq i' < n,\; r' = r - 1\}$
$\mathbf{3} : wh(i, n, r) = 1 + wh(i', n, r) \; \{i < n,\; r \leq 0,\; i' = i + 1\}$

$$\frac{\text{Chain: } (2)^{+}(3)^{+}1}{wh(i, n, r)}$$

$\searrow$ CE 2

$1 + wh(i_2, n_2, r_2)$

$\searrow$ CE 2

$\ldots$

Chain Evaluation Schema

$\mathbf{1} : wh(i,n,r) = 0 \quad \{i \geq n\}$

$\mathbf{2} : wh(i,n,r) = 1 + wh(i',n,r') \; \{i < n,\ r > 0,\ 0 \leq i' < n,\ r' = r - 1\}$

$\mathbf{3} : wh(i,n,r) = 1 + wh(i',n,r) \; \{i < n,\ r \leq 0,\ i' = i + 1\}$

Chain: $(2)^{+}(3)^{+}1$

$$
\begin{array}{l}
wh(i,n,r) \\
\qquad \searrow \text{CE 2} \\
\qquad 1 + wh(i_2,n_2,r_2) \\
\qquad\qquad \searrow \text{CE 2} \\
\qquad\qquad \dots \\
\qquad\qquad\quad \searrow \text{CE 2} \\
\qquad\qquad 1 + wh(i_j,n_j,r_j)
\end{array}
$$

Chain Evaluation Schema

$\mathbf{1}$: $wh(i,n,r) = 0 \quad \{i \geq n\}$
$\mathbf{2}$: $wh(i,n,r) = 1 + wh(i',n,r') \; \{i < n,\ r > 0,\ 0 \leq i' < n,\ r' = r-1\}$
$\mathbf{3}$: $wh(i,n,r) = 1 + wh(i',n,r) \; \{i < n,\ r \leq 0,\ i' = i+1\}$

$$\frac{\text{Chain: } (2)^+(3)^+1}{\;}$$

$$
\begin{aligned}
& wh(i,n,r) \\
& \quad \searrow \text{CE 2} \\
& \qquad 1 + wh(i_2,n_2,r_2) \\
& \qquad\quad \searrow \text{CE 2} \\
& \qquad\quad \dots \\
& \qquad\quad\; \searrow \text{CE 2} \\
& \qquad\qquad 1 + wh(i_j,n_j,r_j) \\
& \qquad\qquad\quad \searrow \text{CE 2} \\
& \qquad\qquad\qquad 1 + wh(i_f,n_f,r_f)
\end{aligned}
$$

Chain Evaluation Schema

$\mathbf{1}: wh(i,n,r) = 0 \quad \{i \geq n\}$

$\mathbf{2}: wh(i,n,r) = 1 + wh(i',n,r')\ \{i < n,\ r > 0,\ 0 \leq i' < n,\ r' = r-1\}$

$\mathbf{3}: wh(i,n,r) = 1 + wh(i',n,r)\ \{i < n,\ r \leq 0,\ i' = i+1\}$

$$\frac{\text{Chain: } (2)^{+}(3)^{+}1}{wh(i,n,r)}$$

$$\searrow \text{CE } 2$$

$$1 + wh(i_2, n_2, r_2)$$

$$\searrow \text{CE } 2$$

$$\ldots$$

$$\searrow \text{CE } 2$$

$$1 + wh(i_j, n_j, r_j)$$

$$\searrow \text{CE } 2$$

$$1 + wh(i_f, n_f, r_f)$$

$$\searrow \text{CE } 3$$

$$\ldots$$

$$\searrow \text{CE } 3$$

$$1 + wh(i_{f_3}, n_{f_3}, r_{f_3})$$

Chain Evaluation Schema

$\mathbf{1}: wh(i,n,r) = 0 \quad \{i \geq n\}$
$\mathbf{2}: wh(i,n,r) = 1 + wh(i',n,r') \; \{i < n,\, r > 0,\, 0 \leq i' < n,\, r' = r - 1\}$
$\mathbf{3}: wh(i,n,r) = 1 + wh(i',n,r) \; \{i < n,\, r \leq 0,\, i' = i + 1\}$

Chain: $(2)^+(3)^+1$

$$
\begin{array}{l}
wh(i,n,r) \\
\quad \searrow \text{CE } 2 \\
\quad 1 + wh(i_2,n_2,r_2) \\
\qquad \searrow \text{CE } 2 \\
\qquad \ldots \\
\qquad\quad \searrow \text{CE } 2 \\
\qquad\quad 1 + wh(i_j,n_j,r_j) \\
\qquad\qquad \searrow \text{CE } 2 \\
\qquad\qquad 1 + wh(i_f,n_f,r_f) \\
\qquad\qquad\quad \searrow \text{CE } 3 \\
\qquad\qquad\quad \ldots \\
\qquad\qquad\qquad \searrow \text{CE } 3 \\
\qquad\qquad\qquad 1 + wh(i_{f_3},n_{f_3},r_{f_3}) \\
\qquad\qquad\qquad\quad \searrow \text{CE } 1 \\
\qquad\qquad\qquad\quad 0
\end{array}
$$

Bound Computation Strategy

Bottom-up incremental approach

Chain: $(2)^{+}(3)^{+}1$

$$
\begin{array}{l}
wh(i,n,r) \\
\quad \searrow \text{CE } 2 \\
\quad 1 + wh(i_2,n_2,r_2) \\
\qquad \searrow \text{CE } 2 \\
\qquad \ldots \\
\qquad\quad \searrow \text{CE } 2 \\
\qquad\quad 1 + wh(i_j,n_j,r_j) \\
\qquad\qquad \searrow \text{CE } 2 \\
\qquad\qquad 1 + wh(i_f,n_f,r_f) \\
\qquad\qquad\quad \searrow \text{CE } 3 \\
\qquad\qquad\quad \ldots \\
\qquad\qquad\qquad \searrow \text{CE } 3 \\
\qquad\qquad\qquad 1 + wh(i_{f_3},n_{f_3},r_{f_3}) \\
\qquad\qquad\qquad\quad \searrow \text{CE } 1 \\
\qquad\qquad\qquad\quad 0
\end{array}
$$

Bottom-up incremental approach

1. Compute bound per cost equation without considering recursive calls

Chain: $(2)^{+}(3)^{+}1$

$wh(i, n, r)$

$\searrow$ CE 2

$\boxed{C_2}$ $wh(i_2, n_2, r_2)$

$\searrow$ CE 2

$\boxed{C_2}$ …

$\searrow$ CE 2

$\boxed{C_2}$ $wh(i_j, n_j, r_j)$

$\searrow$ CE 2

$\boxed{C_2}$ $wh(i_f, n_f, r_f)$

$\searrow$ CE 3

$\boxed{C_3}$ …

$\searrow$ CE 3

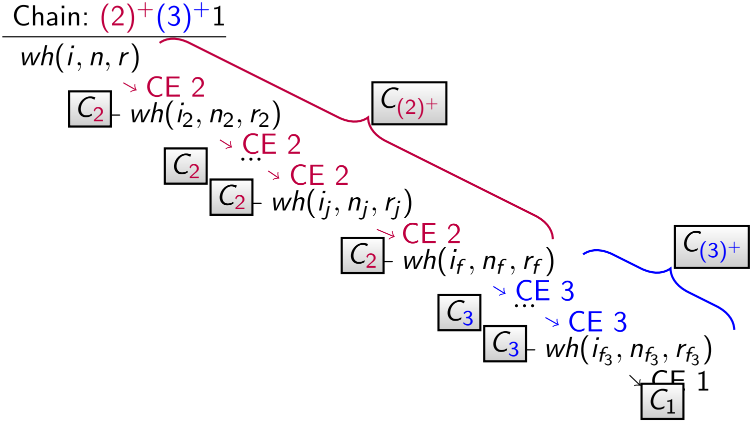
$\boxed{C_3}$ $wh(i_{f_3}, n_{f_3}, r_{f_3})$

$\searrow$ CE 1

$\boxed{C_1}$

Bottom-up incremental approach

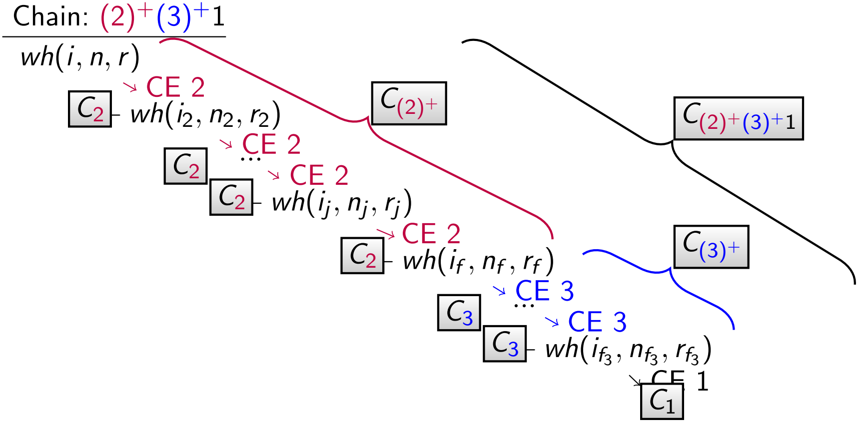
1. Compute bound per cost equation without considering recursive calls
2. Compute cost of each phase by composing the cost of their CEs



Bound Computation Strategy

Bottom-up incremental approach

1. Compute bound per cost equation without considering recursive calls
2. Compute cost of each phase by composing the cost of their CEs
3. Compose cost of phases to obtain the cost of whole chain



Cost Representation: Separation of Concerns

Decompose Symbolic Cost Expressions

1. Simple expressions over (auxiliary) intermediate variables (**iv**)
2. Constraints that bind intermediate variables to program/cost relation variables

```
while (i<n) {
 if (r>0) {
  i=rnd(n-1);
  r--;
 } else
  i++;
}
```

To represent the upper bound $|\mathbf{r}| + |\mathbf{n}|$:

1 $\mathbf{iv_1} + \mathbf{iv_2}$ 2 $\{\mathbf{iv_1} \leq |\mathbf{r}|, \mathbf{iv_2} \leq |\mathbf{n}|\}$

$|x|$ abbreviates $\max(x, 0)$

Cost Representation: Separation of Concerns

Decompose Symbolic Cost Expressions

1. Simple expressions over (auxiliary) intermediate variables (**iv**)
2. Constraints that bind intermediate variables to program/cost relation variables

Advantages:

- Consider multiple bound candidates

```
while (x>0 && y>0) {
   x--;
   y--;
}
```

$$\boxed{\mathbf{iv_1} \quad \{\mathbf{iv_1} \leq |\mathbf{x}|, \mathbf{iv_1} \leq |\mathbf{y}|\}}$$

Cost Representation: Separation of Concerns

Decompose Symbolic Cost Expressions

1. Simple expressions over (auxiliary) intermediate variables (**iv**)
2. Constraints that bind intermediate variables to program/cost relation variables

Advantages:

- Consider multiple bound candidates
- Facilitates incremental bound inference and composition

```
while (x>0 && *)
  x--;
//exit value x'
while (x>0)
  x--;
```

Cost of first loop:
$\mathbf{iv_1} \quad \{\mathbf{iv_1} \leq |\mathbf{x} - \mathbf{x'}|\}$

Cost of second loop:
$\mathbf{iv_2} \quad \{\mathbf{iv_1} \leq |\mathbf{x'}|\}$

Total cost:
$\mathbf{iv_1} + \mathbf{iv_2} \quad \{\mathbf{iv_1} + \mathbf{iv_2} \leq |\mathbf{x}|\}$

Represent cost with cost structures $\langle E, IC, FC \rangle$

- E is the main cost expression
  - Linear expression over intermediate variables (**iv**)

Cost Representation: Cost Structures

Represent cost with cost structures $\langle E, IC, FC \rangle$

- E is the main cost expression
  - Linear expression over intermediate variables (**iv**)
- IC, FC are constraint sets: non-final and final

Cost Representation: Cost Structures

Represent cost with cost structures $\langle E, IC, FC \rangle$

- E is the main cost expression
  - Linear expression over intermediate variables (**iv**)
- IC, FC are constraint sets: non-final and final

FC Final constraints bind sums of intermediate variables to linear expressions over cost relation variables, e.g.:

$$iv_1 + iv_2 \leq |x + y|$$

Cost Representation: Cost Structures

Represent cost with cost structures $\langle E, IC, FC \rangle$

- E is the main cost expression
  - Linear expression over intermediate variables (**iv**)
- IC, FC are constraint sets: non-final and final

FC Final constraints bind sums of intermediate variables to linear expressions over cost relation variables, e.g.:

$$iv_1 + iv_2 \leq |x + y|$$

IC (intermediate) non-final constraints bind sums of intermediate variables to expressions over other intermediate variables

- can be non-linear and contain max and min
- permit to represent non-linear bounds, e.g.:

$$iv_1 + iv_2 \leq iv_3 * iv_4$$

Cost Representation: Cost Structures

Represent cost with cost structures $\langle E, IC, FC \rangle$

- E is the main cost expression
  - Linear expression over intermediate variables (**iv**)
- IC, FC are constraint sets: non-final and final

FC Final constraints bind sums of intermediate variables to linear expressions over cost relation variables, e.g.:

$$iv_1 + iv_2 \leq |x + y|$$

IC (intermediate) non-final constraints bind sums of intermediate variables to expressions over other intermediate variables

- can be non-linear and contain max and min
- permit to represent non-linear bounds, e.g.:

$$iv_1 + iv_2 \leq iv_3 * iv_4$$

We will not use non-final constraints in this presentation

Computing Cost Structures: Cost Equations

$$\begin{aligned}
&\mathbf{1} : wh(i,n,r) = \underline{0} \quad \{i \geq n\}\\
&\mathbf{2} : wh(i,n,r) = \underline{1} + wh(i',n,r')\ \{i < n,\ r > 0,\ 0 \leq i' < n,\ r' = r-1\}\\
&\mathbf{3} : wh(i,n,r) = \underline{1} + wh(i',n,r)\ \{i < n,\ r \leq 0,\ i' = i+1\}
\end{aligned}$$

- Do not take into account the cost of recursive calls
- Use cost equation definition

$\boxed{C_2 : \langle \mathbf{iv_1}, \emptyset, \{\mathbf{iv_1} \leq \mathbf{1}\}\rangle}$

$\boxed{C_3 : \langle \mathbf{iv_2}, \emptyset, \{\mathbf{iv_2} \leq \mathbf{1}\}\rangle}$

$\boxed{C_1 : \langle \mathbf{0}, \emptyset, \emptyset\rangle}$

$$\begin{aligned}
&\mathbf{1} : wh(i,n,r) = 0 \quad \{i \geq n\}\\
&\mathbf{2} : wh(i,n,r) = 1 + wh(i',n,r')\ \{i < n,\ r > 0,\ 0 \leq i' < n,\ r' = r - 1\} = \varphi_2\\
&\mathbf{3} : wh(i,n,r) = 1 + wh(i',n,r)\ \{i < n,\ r \leq 0,\ i' = i + 1\}
\end{aligned}$$

- Assume CE 2 is evaluated $\#c_2$ times in phase $(2)^+$

$$\boxed{C_2 : \langle iv_1, \emptyset, \{iv_1 \leq 1\}\rangle} \longrightarrow \boxed{C_{(2)^+} :}$$

$$\boxed{C_3 : \langle iv_2, \emptyset, \{iv_2 \leq 1\}\rangle} \longrightarrow \boxed{C_{(3)^+} :}$$

$$\boxed{C_1 : \langle 0, \emptyset, \emptyset\rangle}$$

$$
\begin{aligned}
&\mathbf{1} : wh(i,n,r) = 0 \quad \{i \geq n\}\\
&\mathbf{2} : wh(i,n,r) = 1 + wh(i',n,r')\ \{i < n,\ r > 0,\ 0 \leq i' < n,\ r' = r-1\} = \varphi_2\\
&\mathbf{3} : wh(i,n,r) = 1 + wh(i',n,r)\ \{i < n,\ r \leq 0,\ i' = i+1\}
\end{aligned}
$$

- Assume CE 2 is evaluated $\#c_2$ times in phase $(2)^+$
- The j-th evaluation of CE 2 has cost $\langle iv_{1j}, \emptyset, \{iv_{1j} \leq 1\}\rangle$

$$
\begin{tikzcd}
\boxed{C_2 : \langle iv_1, \emptyset, \{iv_1 \leq 1\}\rangle} \arrow[r] & \boxed{C_{(2)^+} :} \\
\boxed{C_3 : \langle iv_2, \emptyset, \{iv_2 \leq 1\}\rangle} \arrow[r] & \boxed{C_{(3)^+} :} \\
\boxed{C_1 : \langle 0, \emptyset, \emptyset\rangle} &
\end{tikzcd}
$$

Computing Cost Structures: Phases

$$\begin{aligned}
&\mathbf{1} : wh(i,n,r) = 0 \quad \{i \geq n\}\\
&\mathbf{2} : wh(i,n,r) = 1 + wh(i',n,r')\ \{i < n,\ r > 0,\ 0 \leq i' < n,\ r' = r-1\} = \varphi_2\\
&\mathbf{3} : wh(i,n,r) = 1 + wh(i',n,r)\ \{i < n,\ r \leq 0,\ i' = i+1\}
\end{aligned}$$

- Assume CE 2 is evaluated $\#c_2$ times in phase $(2)^+$
- The j-th evaluation of CE 2 has cost $\langle iv_{1j}, \emptyset, \{iv_{1j} \leq 1\}\rangle$
- Create a new intermediate variable $iv_3 = \sum_{j=1}^{\#c_2} iv_{1j}$

$$\begin{tikzcd}
\boxed{C_2 : \langle iv_1, \emptyset, \{iv_1 \leq 1\}\rangle} \arrow[r] & \boxed{C_{(2)^+} :} \\
\boxed{C_3 : \langle iv_2, \emptyset, \{iv_2 \leq 1\}\rangle} \arrow[r] & \boxed{C_{(3)^+} :} \\
\boxed{C_1 : \langle 0, \emptyset, \emptyset\rangle} &
\end{tikzcd}$$

$$\begin{aligned}
&\mathbf{1} : wh(i,n,r) = 0 \quad \{i \geq n\}\\
&\mathbf{2} : wh(i,n,r) = 1 + wh(i',n,r')\ \{i < n,\ r > 0,\ 0 \leq i' < n,\ r' = r-1\} = \varphi_2\\
&\mathbf{3} : wh(i,n,r) = 1 + wh(i',n,r)\ \{i < n,\ r \leq 0,\ i' = i+1\}
\end{aligned}$$

- Assume CE 2 is evaluated $\#c_2$ times in phase $(2)^+$
- The j-th evaluation of CE 2 has cost $\langle iv_{1j}, \emptyset, \{iv_{1j} \leq 1\}\rangle$
- Create a new intermediate variable $iv_3 = \sum_{j=1}^{\#c_2} iv_{1j}$
- The cost of $(2)^+$ is represented by iv_3 — Bound?

$$\boxed{C_2 : \langle iv_1, \emptyset, \{iv_1 \leq 1\}\rangle} \longrightarrow \boxed{C_{(2)^+} : \langle \mathbf{iv_3}, \emptyset, ?\rangle}$$

$$\boxed{C_3 : \langle iv_2, \emptyset, \{iv_2 \leq 1\}\rangle} \longrightarrow \boxed{C_{(3)^+} :}$$

$$\boxed{C_1 : \langle 0, \emptyset, \emptyset\rangle}$$

Computing Cost Structures: Phases

$$\mathbf{1}: wh(i,n,r) = 0 \quad \{i \geq n\}$$
$$\mathbf{2}: wh(i,n,r) = 1 + wh(i',n,r') \ \{\mathbf{i < n, r > 0, 0 \leq i' < n, r' = r - 1}\} = \varphi_{\mathbf{2}}$$
$$\mathbf{3}: wh(i,n,r) = 1 + wh(i',n,r) \ \{i < n, r \leq 0, i' = i + 1\}$$

- Assume CE 2 is evaluated $\#c_2$ times in phase $(2)^+$
- The j-th evaluation of CE 2 has cost $\langle iv_{1j}, \emptyset, \{iv_{1j} \leq 1\}\rangle$
- Create a new intermediate variable $iv_3 = \sum_{j=1}^{\#c_2} iv_{1j}$
- The cost of $(2)^+$ is represented by iv_3 — Bound?
- Heuristics and templates to generate constraints for iv_3:
  - **Inductive Sum**: Farkas' Lemma with linear template $L(i,n,r)$ and constraint set φ_2 of CE 2: obtain a symbolic expression that satisfies: $\varphi_2 \Rightarrow (L(i,n,r) \geq 1 \wedge L(i,n,r) \geq 1 + L(i',n',r'))$

$\boxed{C_2 : \langle iv_1, \emptyset, \{iv_1 \leq 1\}\rangle} \longrightarrow \boxed{C_{(2)^+} : \langle \mathbf{iv_3}, \emptyset, \{\mathbf{iv_3} \leq |\mathbf{r}|\}\rangle}$

$\boxed{C_3 : \langle iv_2, \emptyset, \{iv_2 \leq 1\}\rangle} \longrightarrow \boxed{C_{(3)^+} :}$

$\boxed{C_1 : \langle 0, \emptyset, \emptyset\rangle}$

Computing Cost Structures: Phases

$\mathbf{1}: wh(i,n,r) = 0 \quad \{i \geq n\}$
$\mathbf{2}: wh(i,n,r) = 1 + wh(i',n,r') \; \{i < n,\ r > 0,\ 0 \leq i' < n,\ r' = r-1\} = \varphi_2$
$\mathbf{3}: wh(i,n,r) = 1 + wh(i',n,r) \; \{\mathbf{i} < \mathbf{n},\ \mathbf{r} \leq \mathbf{0},\ \mathbf{i}' = \mathbf{i} + \mathbf{1}\}$

- Assume CE 2 is evaluated $\#c_2$ times in phase $(2)^+$
- The j-th evaluation of CE 2 has cost $\langle iv_{1j}, \emptyset, \{iv_{1j} \leq 1\}\rangle$
- Create a new intermediate variable $iv_3 = \sum_{j=1}^{\#c_2} iv_{1j}$
- The cost of $(2)^+$ is represented by iv_3 — Bound?
- Heuristics and templates to generate constraints for iv_3:
  - **Inductive Sum**: Farkas' Lemma with linear template $L(i,n,r)$ and constraint set φ_2 of CE 2: obtain a symbolic expression that satisfies: $\varphi_2 \Rightarrow (L(i,n,r) \geq 1 \wedge L(i,n,r) \geq 1 + L(i',n',r'))$

$\boxed{C_2 : \langle iv_1, \emptyset, \{iv_1 \leq 1\}\rangle} \longrightarrow \boxed{C_{(2)^+} : \langle \mathbf{iv_3}, \emptyset, \{\mathbf{iv_3} \leq |\mathbf{r}|\}\rangle}$

$\boxed{C_3 : \langle iv_2, \emptyset, \{iv_2 \leq 1\}\rangle} \longrightarrow \boxed{C_{(3)^+} : \langle \mathbf{iv_4}, \emptyset, \{\mathbf{iv_4} \leq |\mathbf{n} - \mathbf{i}|\}\rangle}$

$\boxed{C_1 : \langle 0, \emptyset, \emptyset\rangle}$

Computing Cost Structures: Chains

$\mathbf{1}: wh(i,n,r) = 0 \quad \{i \geq n\}$
$\mathbf{2}: wh(i,n,r) = 1 + wh(i',n,r') \; \{i < n,\ r > 0,\ 0 \leq i' < n,\ r' = r - 1\}$
$\mathbf{3}: wh(i,n,r) = 1 + wh(i',n,r) \; \{i < n,\ r \leq 0,\ i' = i + 1\}$

- Add cost of the phases in a chain

$C_{(2)^+} : \langle iv_3, \emptyset, \{iv_3 \leq |r|\}\rangle$

$C_{(3)^+} : \langle iv_4, \emptyset, \{iv_4 \leq |n - i|\}\rangle$

$C_1 : \langle 0, \emptyset, \emptyset\rangle$

$C_{(2)^+(3)^+1}$:
$\langle \mathbf{iv_3} + \mathbf{iv_4},\ \emptyset,\ ?\rangle$

Computing Cost Structures: Chains

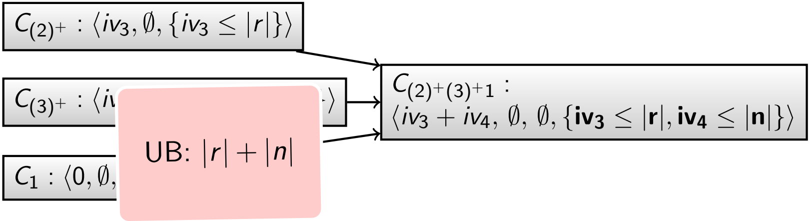
$$\mathbf{1} : wh(i, n, r) = 0 \quad \{i \geq n\}$$
$$\mathbf{2} : wh(i, n, r) = 1 + wh(i', n, r') \; \{i < n,\ r > 0,\ \mathbf{0} \leq \mathbf{i}' < \mathbf{n},\ r' = r - 1\}$$
$$\mathbf{3} : wh(i, n, r) = 1 + wh(i', n, r) \; \{i < n,\ r \leq 0,\ i' = i + 1\}$$

- Add cost of the phases in a chain
- To obtain bounds: express constraints in terms of initial variables
  - in phase $(2)^+$ variable i is set to arbitrary value between 0 and $n - 1$
  - worst case: expression $n - i$ in phase $(3)^+$ has value n

$$\boxed{C_{(2)^+} : \langle iv_3, \emptyset, \{iv_3 \leq |r|\}\rangle}$$
$$\boxed{C_{(3)^+} : \langle iv_4, \emptyset, \{iv_4 \leq |n - i|\}\rangle}$$
$$\boxed{C_1 : \langle 0, \emptyset, \emptyset\rangle}$$
$$\longrightarrow \boxed{C_{(2)^+(3)^+1} : \langle iv_3 + iv_4,\ \emptyset,\ \emptyset,\ \{\mathbf{iv_3} \leq |\mathbf{r}|, \mathbf{iv_4} \leq |\mathbf{n}|\}\rangle}$$

$$
\begin{aligned}
&\mathbf{1} : wh(i,n,r) = 0 \quad \{i \geq n\}\\
&\mathbf{2} : wh(i,n,r) = 1 + wh(i',n,r') \; \{i < n,\ r > 0,\ 0 \leq i' < n,\ r' = r - 1\}\\
&\mathbf{3} : wh(i,n,r) = 1 + wh(i',n,r) \; \{i < n,\ r \leq 0,\ i' = i + 1\}
\end{aligned}
$$

- Add cost of the phases in a chain
- To obtain bounds: express constraints in terms of initial variables
  - in phase $(2)^+$ variable i is set to arbitrary value between 0 and $n - 1$
  - worst case: expression $n - i$ in phase $(3)^+$ has value n



Amortized cost

```
while (l!=[]) {
 s=Cons(head(l),s);
 if (*)
    s=popSome(s);
 l=tail(l);
}
```

```
popSome(List s) {
 if (s==[] || *)
    return s;
 else
    popSome(tail(s));
}
```

- While-loop adds elements of list l to list s
  - In worst case loop can iterate l times
- Occasionally, popSome() removes elements from s
  - In worst case popSome() is called l times
- All elements on s not already present at start come from l:
  - Total cost is $\mathcal{O}(l)$ and not $\mathcal{O}(l^2)$
- Example involves amortized cost

```
while (l!=[]) {
 s=Cons(head(l),s);
 if (*)
    s=popSome(s);
 l=tail(l);
}
```

```
popSome(List s) {
  if (s==[] || *)
     return s;
  else
     popSome(tail(s));
}
```

$$
\begin{array}{l}
4 : whA(l,s) = 0 \quad \{l = 0\} \\
5 : whA(l,s) = 1 + popSome(s'',s') + whA(l',s') \\
\qquad\qquad \{l \geq 1,\, s \geq 0,\, s'' = s+1,\, l' = l-1\} \\
6 : whA(l,s) = 1 + whA(l',s')\ \{l \geq 1,\, s \geq 0,\, s' = s+1,\, l' = l-1\} \\
\hline
7 : popSome(s,so) = 0 \quad \{s = so\} \\
8 : popSome(s,so) = 1 + popSome(s',so)\ \{s \geq 1,\, s' = s-1\}
\end{array}
$$

- Lists are abstracted to their length

```
while (l!=[]) {
 s=Cons(head(l),s);
 if (*)
    s=popSome(s);
 l=tail(l);
}
```

```
popSome(List s) {
  if (s==[] || *)
      return s;
  else
      popSome(tail(s));
}
```

$$
\begin{array}{l}
4 : whA(l,s) = 0 \quad \{l = 0\} \\
5 : whA(l,s) = 1 + popSome(s'', s') + whA(l', s') \\
\qquad\qquad \{l \geq 1,\ s \geq 0,\ s'' = s + 1,\ l' = l - 1\} \\
6 : whA(l,s) = 1 + whA(l', s')\ \{l \geq 1,\ s \geq 0,\ s' = s + 1,\ l' = l - 1\} \\
\hline
7 : popSome(s, so) = 0 \quad \{s = so\} \\
8 : popSome(s, so) = 1 + popSome(s', so)\ \{s \geq 1,\ s' = s - 1\}
\end{array}
$$

- Lists are abstracted to their length
- Essential to consider the return value of popSome()

Recall Control Flow Refinement: popSome()

$$
\begin{aligned}
&4 : whA(l,s) = 0 \quad \{l = 0\}\\
&5 : whA(l,s) = 1 + popSome(s'', s') + whA(l', s')\\
&\qquad\qquad \{l \geq 1,\ s \geq 0,\ s'' = s + 1,\ l' = l - 1\}\\
&6 : whA(l,s) = 1 + whA(l', s')\ \{l \geq 1,\ s \geq 0,\ s' = s + 1,\ l' = l - 1\}\\
\hline
&7 : popSome(s, so) = 0 \quad \{s = so\}\\
&8 : popSome(s, so) = 1 + popSome(s', so)\ \{s \geq 1,\ s' = s - 1\}
\end{aligned}
$$

$$
\begin{tikzcd}
8 \arrow[loop above] \arrow[r] & 7
\end{tikzcd}
$$

Recall Control Flow Refinement: popSome()

$$
\begin{array}{l}
4: whA(l,s) = 0 \quad \{l = 0\} \\
5: whA(l,s) = 1 + popSome(s'', s') + whA(l', s') \\
\qquad\qquad \{l \geq 1,\ s \geq 0,\ s'' = s + 1,\ l' = l - 1\} \\
6: whA(l,s) = 1 + whA(l', s')\ \{l \geq 1,\ s \geq 0,\ s' = s + 1,\ l' = l - 1\} \\
\hline
7: popSome(s, so) = 0 \quad \{s = so\} \\
8: popSome(s, so) = 1 + popSome(s', so)\ \{s \geq 1,\ s' = s - 1\}
\end{array}
$$

Phases
$(8)^+$
7

$$
\begin{tikzcd}
8 \arrow[loop above] \arrow[r] & 7
\end{tikzcd}
$$

Recall Control Flow Refinement: popSome()

$4: whA(l,s) = 0 \quad \{l = 0\}$
$5: whA(l,s) = 1 + popSome(s'', s') + whA(l', s')$
$\quad \{l \geq 1,\ s \geq 0,\ s'' = s+1,\ l' = l-1\}$
$6: whA(l,s) = 1 + whA(l', s')\ \{l \geq 1,\ s \geq 0,\ s' = s+1,\ l' = l-1\}$

$7: popSome(s, so) = 0 \quad \{s = so\}$
$8: popSome(s, so) = 1 + popSome(s', so)\ \{s \geq 1,\ s' = s-1\}$

$$\begin{tikzcd} 8 \arrow[loop above] \arrow[r] & 7 \end{tikzcd}$$

Phases

$(8)^+$
7

Chains

Non-terminating	Terminating
$(8)^+$	$(8)^+7$
	7

Recall Control Flow Refinement: popSome()

$4 : whA(l, s) = 0 \quad \{l = 0\}$
$5 : whA(l, s) = 1 + popSome(s'', s') + whA(l', s')$
$\qquad \{l \geq 1,\ s \geq 0,\ s'' = s + 1,\ l' = l - 1\}$
$6 : whA(l, s) = 1 + whA(l', s')\ \{l \geq 1,\ s \geq 0,\ s' = s + 1,\ l' = l - 1\}$

$7 : popSome(s, so) = 0 \quad \{s = so\}$
$8 : popSome(s, so) = 1 + popSome(s', so)\ \{s \geq 1,\ s' = s - 1\}$

$$\begin{tikzcd} 8 \arrow[loop above] \arrow[r] & 7 \end{tikzcd}$$

Phases	**Ranking functions**
$(8)^+$	s
7	

Chains

Non-terminating	Terminating
$\cancel{(8)^+}$	$(8)^+7$
	7

Recall Control Flow Refinement: popSome()

$4 : whA(l,s) = 0 \quad \{l = 0\}$
$5 : whA(l,s) = 1 + popSome(s'',s') + whA(l',s')$
$\qquad \{l \geq 1,\ s \geq 0,\ s'' = s+1,\ l' = l-1\}$
$6 : whA(l,s) = 1 + whA(l',s')\ \{l \geq 1,\ s \geq 0,\ s' = s+1,\ l' = l-1\}$

$7 : popSome(s,so) = 0 \quad \{s = so\}$
$8 : popSome(s,so) = 1 + popSome(s',so)\ \{s \geq 1,\ s' = s-1\}$

$$\begin{tikzcd} 8 \arrow[loop above] \arrow[r] & 7 \end{tikzcd}$$

Phases	**Ranking functions**
$(8)^+$	s
7	

Chains

Non-terminating	Terminating	**Invariants/Summaries**
$\cancel{(8)^+}$	$(8)^+7$	$s \geq 1 \land s > so$
	7	$s = so$

- Now invariants constitute summaries (inclusion of return variable)

Refinement Propagation

Propagate control flow refinement from popSome() to whA

- Substitute calls to popSome() by calls to refined chains of popSome()
- Add summaries of called chain to constraint set of caller

$4 : whA(l, s) = 0 \quad \{l = 0\}$

$\mathbf{5} : \mathbf{whA(l, s) = 1 + popSome(s'', s') + whA(l', s')}$

$\quad \{\mathbf{l \geq 1,\ s \geq 0,\ s'' = s + 1,\ l' = l - 1}\}$

$\underline{6 : whA(l, s) = 1 + whA(l', s')\ \{l \geq 1,\ s \geq 0,\ s' = s + 1,\ l' = l - 1\}}$

Refinement Propagation

Propagate control flow refinement from popSome() to whA

- Substitute calls to popSome() by calls to refined chains of popSome()
- Add summaries of called chain to constraint set of caller

$4: whA(l,s) = 0 \quad \{l = 0\}$

$\mathbf{5}: \mathbf{whA(l,s) = 1 + popSome(s'',s') + whA(l',s')}$
$\quad \{\mathbf{l \geq 1,\ s \geq 0,\ s'' = s+1,\ l' = l-1}\}$

$6: whA(l,s) = 1 + whA(l',s')\ \{l \geq 1,\ s \geq 0,\ s' = s+1,\ l' = l-1\}$

$$\Downarrow$$

$5.1: whA(l,s) = 1 + popSome[\mathbf{7}](s'',s') + whA(l',s')$
$\quad \{l \geq 1,\ s \geq 0,\ s'' = s+1,\ l' = l-1,\ \mathbf{s'' = s'}\}$

$5.2: whA(l,s) = 1 + popSome[(\mathbf{8})^{+}\mathbf{7}](s'',s') + whA(l',s')$
$\quad \{l \geq 1,\ s \geq 0,\ s'' = s+1,\ l' = l-1,\ \mathbf{s'' > s'}\}$

Refinement Propagation

Propagate control flow refinement from popSome() to whA

- Substitute calls to popSome() by calls to refined chains of popSome()
- Add summaries of called chain to constraint set of caller

$4 : whA(l, s) = 0 \quad \{l = 0\}$

$\mathbf{5} : \mathbf{whA(l, s) = 1 + popSome(s'', s') + whA(l', s')}$
$\quad \{\mathbf{l \geq 1,\ s \geq 0,\ s'' = s + 1,\ l' = l - 1}\}$

$6 : whA(l, s) = 1 + whA(l', s')\ \{l \geq 1,\ s \geq 0,\ s' = s + 1,\ l' = l - 1\}$

$\Downarrow$

$5.1 : whA(l, s) = 1 + popSome[\mathbf{7}](s'', s') + whA(l', s')$
$\quad \{l \geq 1,\ s \geq 0,\ s'' = s + 1,\ l' = l - 1,\ \mathbf{s'' = s'}\}$

$5.2 : whA(l, s) = 1 + popSome[(\mathbf{8})^{+}\mathbf{7}](s'', s') + whA(l', s')$
$\quad \{l \geq 1,\ s \geq 0,\ s'' = s + 1,\ l' = l - 1,\ \mathbf{s'' > s'}\}$

Again refine resulting CEs to obtain chains $(5.1 \vee 5.2 \vee 6)^{+}4$ and 4

Bound Computation

$4 : whA(l, s) = 0 \quad \{l = 0\}$

$5.1 : whA(l, s) = 1 + popSome[7](s'', s') + whA(l', s')$
$\{l \geq 1,\ s \geq 0,\ s'' = s + 1,\ l' = l - 1,\ s'' = s'\}$

$5.2 : whA(l, s) = 1 + popSome[(8)^{+}7](s'', s') + whA(l', s')$
$\{l \geq 1,\ s \geq 0,\ s'' = s + 1,\ l' = l - 1,\ s'' > s'\}$

$6 : whA(l, s) = 1 + whA(l', s')\ \{l \geq 1,\ s \geq 0,\ s' = s + 1,\ l' = l - 1\}$

$7 : popSome(s, so) = 0 \quad \{s = so\}$

$8 : popSome(s, so) = 1 + popSome(s', so)\ \{s \geq 1,\ s' = s - 1\}$

Bound Computation

$$4 : whA(l,s) = 0 \quad \{l = 0\}$$

$$5.1 : whA(l,s) = 1 + popSome[7](s'',s') + whA(l',s')$$
$$\{l \geq 1,\ s \geq 0,\ s'' = s+1,\ l' = l-1,\ s'' = s'\}$$

$$5.2 : whA(l,s) = 1 + popSome[(8)^{+}7](s'',s') + whA(l',s')$$
$$\{l \geq 1,\ s \geq 0,\ s'' = s+1,\ l' = l-1,\ s'' > s'\}$$

$$6 : whA(l,s) = 1 + whA(l',s')\ \{l \geq 1,\ s \geq 0,\ s' = s+1,\ l' = l-1\}$$

$$7 : popSome(s,so) = 0 \quad \{s = so\}$$

$$8 : popSome(s,so) = 1 + popSome(s',so)\ \{s \geq 1,\ s' = s-1\}$$

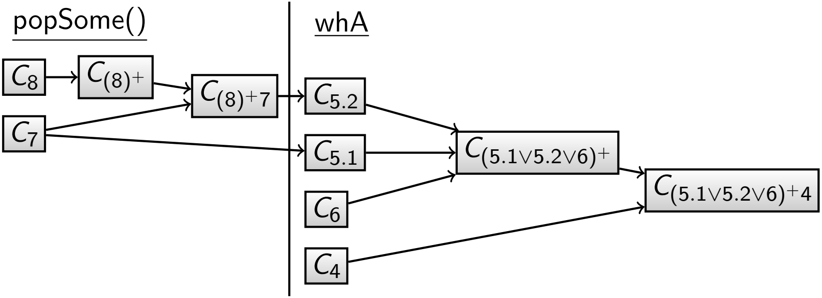
Again, peformed bottom-up (C_c denotes chain c)

popSome()

$$\begin{tikzcd}
\boxed{C_8} \arrow[r] & \boxed{C_{(8)^+}} \arrow[dr] & \\
& & \boxed{C_{(8)^+7}} \\
\boxed{C_7} \arrow[urr] & &
\end{tikzcd}$$

$$
\begin{aligned}
&4 : whA(l,s) = 0 \quad \{l = 0\}\\
&5.1 : whA(l,s) = 1 + popSome[7](s'', s') + whA(l', s')\\
&\qquad\qquad \{l \geq 1,\ s \geq 0,\ s'' = s+1,\ l' = l-1,\ s'' = s'\}\\
&5.2 : whA(l,s) = 1 + popSome[(8)^{+}7](s'', s') + whA(l', s')\\
&\qquad\qquad \{l \geq 1,\ s \geq 0,\ s'' = s+1,\ l' = l-1,\ s'' > s'\}\\
&6 : whA(l,s) = 1 + whA(l', s')\ \{l \geq 1,\ s \geq 0,\ s' = s+1,\ l' = l-1\}\\
\hline
&7 : popSome(s, so) = 0 \quad \{s = so\}\\
&8 : popSome(s, so) = 1 + popSome(s', so)\ \{s \geq 1,\ s' = s-1\}
\end{aligned}
$$

Again, peformed bottom-up (C_c denotes chain c)



First Step: Bound Computation of popSome()

- List s might not be completely consumed in popSome():
  - to obtain amortized cost, express $C_{(8)^+}$ in terms of s, so **and** $\mathbf{s_f}$, $\mathbf{so_f}$

Chain: $(8)^{+}7$

$$
\begin{array}{l}
\text{popSome}(\mathbf{s}, \mathbf{so}) \\
\quad \searrow \text{CE } 8 \\
\quad 1 + \text{popSome}(s_2, so_2) \\
\qquad \searrow \text{CE } 8 \\
\qquad \dots \\
\qquad\quad \searrow \text{CE } 8 \\
\qquad\quad 1 + \text{popSome}(\mathbf{s_f}, \mathbf{so_f}) \\
\qquad\qquad \searrow \text{CE } 7 \\
\qquad\qquad 0
\end{array}
$$

First Step: Bound Computation of popSome()

- List s might not be completely consumed in popSome():
  - to obtain amortized cost, express $C_{(8)^+}$ in terms of s, so **and** $\mathbf{s_f}$, $\mathbf{so_f}$
- Define intermediate variable $iv_2 = \sum_{j=1}^{\#c_8} (iv_{1_j})$

$$7 : \text{popSome}(s, so) = 0 \quad \{s = so\}$$
$$8 : \text{popSome}(s, so) = 1 + \text{popSome}(s', so) \ \{s \geq 1,\ s' = s - 1\}$$

$\boxed{C_8 : \langle iv_1, \emptyset, \{iv_1 \leq 1\}\rangle}$

$\boxed{C_7 : \langle 0, \emptyset, \emptyset\rangle}$

First Step: Bound Computation of popSome()

- List s might not be completely consumed in popSome():
  - to obtain amortized cost, express $C_{(8)^+}$ in terms of s, so **and** $\mathbf{s_f}$, $\mathbf{so_f}$
- Define intermediate variable $iv_2 = \sum_{j=1}^{\#c_8}(iv_{1j})$
- Apply "Inductive Sum" heuristics to obtain bound $iv_2 \leq |s - s_f|$
  - $\#$ of iterations of $(8)^+$ bound by difference initial/final value of s

$7 : \text{popSome}(s, so) = 0 \quad \{s = so\}$
$8 : \text{popSome}(s, so) = 1 + \text{popSome}(s', so)\ \{s \geq 1,\ s' = s - 1\}$

$\boxed{C_8 : \langle iv_1, \emptyset, \{iv_1 \leq 1\}\rangle} \longrightarrow \boxed{C_{(8)^+} : \langle iv_2, \emptyset, \{iv_2 \leq |s - s_f|\}\rangle}$

$\boxed{C_7 : \langle 0, \emptyset, \emptyset\rangle}$

First Step: Bound Computation of popSome()

- List s might not be completely consumed in popSome():
  - to obtain amortized cost, express $C_{(8)^+}$ in terms of s, so **and** $\mathbf{s_f}$, $\mathbf{so_f}$
- Define intermediate variable $iv_2 = \sum_{j=1}^{\#c_8} (iv_{1_j})$
- Apply "Inductive Sum" heuristics to obtain bound $iv_2 \leq |s - s_f|$
  - $\#$ of iterations of $(8)^+$ bound by difference initial/final value of s
- Final value of s $(= s_f)$ returned and unmodified in $(8)^+$:
  $s_f = so_f = so$

$7 : \text{popSome}(s, so) = 0 \quad \{\mathbf{s} = \mathbf{so}\}$
$8 : \text{popSome}(s, \mathbf{so}) = 1 + \text{popSome}(s', \mathbf{so}) \ \{s \geq 1,\ s' = s - 1\}$

$$\begin{tikzcd}
\boxed{C_8 : \langle iv_1, \emptyset, \{iv_1 \leq 1\}\rangle} \arrow[r] & \boxed{C_{(8)^+} : \langle iv_2, \emptyset, \{iv_2 \leq |s - s_f|\}\rangle} \arrow[d] \\
\boxed{C_7 : \langle 0, \emptyset, \emptyset\rangle} \arrow[r] & \boxed{C_{(8)^+7} : \langle iv_2, \emptyset, \{iv_2 \leq |s - so|\}\rangle}
\end{tikzcd}$$

Second Step: (1) Bound Computation of CEs of whA()

$$\begin{aligned}
5.1: & \; whA(l,s) = 1 + \text{popSome}[7](s'',s') + whA(l',s') \\
& \quad \{l \geq 1,\ s \geq 0,\ s'' = s+1,\ l' = l-1,\ s'' = s'\} \\
5.2: & \; whA(l,s) = 1 + \text{popSome}[(8)^{+}7](s'',s') + whA(l',s') \\
& \quad \{l \geq 1,\ s \geq 0,\ s'' = s+1,\ l' = l-1,\ s'' > s'\} \\
6: & \; whA(l,s) = 1 + whA(l',s')\ \{l \geq 1,\ s \geq 0,\ s' = s+1,\ l' = l-1\}
\end{aligned}$$

$\boxed{C_{5.1} : \langle\rangle}$

$\boxed{C_7 : \langle 0, \emptyset, \emptyset \rangle}$ $\boxed{C_{5.2} : \langle\rangle}$

$\boxed{C_{(8)^{+}7} : \langle iv_2, \emptyset, \{iv_2 \leq |s - so|\} \rangle}$

$\boxed{C_6 : \langle\rangle}$

Second Step: (1) Bound Computation of CEs of whA()

- Add cost of calls computed in first step to cost relation of caller

$$5.1 : whA(l, s) = 1 \underline{+ \text{popSome}[7](s'', s')} + whA(l', s')$$
$$\{l \geq 1,\ s \geq 0,\ s'' = s + 1,\ l' = l - 1,\ s'' = s'\}$$

$$5.2 : whA(l, s) = 1 \underline{+ \text{popSome}[(8)^{+}7](s'', s')} + whA(l', s')$$
$$\{l \geq 1,\ s \geq 0,\ s'' = s + 1,\ l' = l - 1,\ s'' > s'\}$$

$$6 : whA(l, s) = 1 + whA(l', s')\ \{l \geq 1,\ s \geq 0,\ s' = s + 1,\ l' = l - 1\}$$

$\boxed{C_{5.1} : \langle iv_3, \emptyset, \{iv_3 \leq 1\}\rangle}$

$\boxed{C_7 : \langle 0, \emptyset, \emptyset\rangle}$ $\boxed{C_{5.2} : \langle iv_2 + iv_4, \emptyset, \{iv_4 \leq 1\rangle}$

$\boxed{C_{(8)^{+}7} : \langle iv_2, \emptyset, \{iv_2 \leq |s - so|\}\rangle}$

$\boxed{C_6 : \langle iv_5, \emptyset, \{iv_5 \leq 1\}\rangle}$

- Add cost of calls computed in first step to cost relation of caller
- $|s - so|$ corresponds to $|s'' - s'|$ in CE 5.2

$$5.1 : whA(l, s) = 1 + \text{popSome}[7](s'', s') + whA(l', s')$$
$$\{l \geq 1,\ s \geq 0,\ s'' = s + 1,\ l' = l - 1,\ s'' = s'\}$$

$$5.2 : whA(l, s) = 1 + \text{popSome}[(8)^{+}7](\underline{s'', s'}) + whA(l', s')$$
$$\{l \geq 1,\ s \geq 0,\ s'' = s + 1,\ l' = l - 1,\ s'' > s'\}$$

$$6 : whA(l, s) = 1 + whA(l', s')\ \{l \geq 1,\ s \geq 0,\ s' = s + 1,\ l' = l - 1\}$$

$\boxed{C_{5.1} : \langle iv_3, \emptyset, \{iv_3 \leq 1\}\rangle}$

$\boxed{C_7 : \langle 0, \emptyset, \emptyset\rangle}$

$\boxed{C_{5.2} : \langle iv_2 + iv_4, \emptyset, \{iv_4 \leq 1\}\rangle}$

$\boxed{C_{(8)^{+}7} : \langle iv_2, \emptyset, \{iv_2 \leq \underline{|s - so|}\}\rangle}$

$\boxed{C_6 : \langle iv_5, \emptyset, \{iv_5 \leq 1\}\rangle}$

Second Step: (1) Bound Computation of CEs of whA()

- Add cost of calls computed in first step to cost relation of caller
- $|s - so|$ corresponds to $|s'' - s'|$ in CE 5.2
- Express bounds in terms of initial variables and variables of recursive call (to obtain amortized cost)

$$5.1 : whA(l, s) = 1 + \text{popSome}[7](s'', s') + whA(l', s')$$
$$\{l \geq 1,\ s \geq 0,\ s'' = s + 1,\ l' = l - 1,\ s'' = s'\}$$

$$5.2 : whA(l, s) = 1 + \text{popSome}[(8)^{+}7](s'', s') + whA(l', s')$$
$$\{l \geq 1,\ s \geq 0,\ \mathbf{s''} = \mathbf{s} + \mathbf{1},\ l' = l - 1,\ s'' > s'\}$$

$$6 : whA(l, s) = 1 + whA(l', s')\ \{l \geq 1,\ s \geq 0,\ s' = s + 1,\ l' = l - 1\}$$

$$\boxed{C_{5.1} : \langle iv_3, \emptyset, \{iv_3 \leq 1\}\rangle}$$

$$\boxed{C_7 : \langle 0, \emptyset, \emptyset\rangle}$$

$$\boxed{C_{5.2} : \langle iv_2 + iv_4, \emptyset, \{iv_4 \leq 1,\ \mathbf{iv_2} \leq |\mathbf{s} + \mathbf{1} - \mathbf{s'}|\}\rangle}$$

$$\boxed{C_{(8)^{+}7} : \langle iv_2, \emptyset, \{iv_2 \leq |s - so|\}\rangle}$$

$$\boxed{C_6 : \langle iv_5, \emptyset, \{iv_5 \leq 1\}\rangle}$$

$$5.1 : \mathit{whA}(l, s) = 1 + \text{popSome}[7](s'', s') + \mathit{whA}(l', s')$$
$$\{l \geq 1,\ s \geq 0,\ s'' = s + 1,\ l' = l - 1,\ s'' = s'\} = \varphi_{5.1}$$

$$5.2 : \mathit{whA}(l, s) = 1 + \text{popSome}[(8)^{+}7](s'', s') + \mathit{whA}(l', s')$$
$$\{l \geq 1,\ s \geq 0,\ s'' = s + 1,\ l' = l - 1,\ s'' > s'\} = \varphi_{5.2}$$

$$6 : \mathit{whA}(l, s) = 1 + \mathit{whA}(l', s')\ \{l \geq 1,\ s \geq 0,\ s' = s + 1,\ l' = l - 1\} = \varphi_6$$

$$\boxed{C_{5.2} : \langle iv_2 + iv_4, \emptyset, \{iv_4 \leq 1, iv_2 \leq |s + 1 - s'|\}\rangle} \quad \boxed{C_{5.1} : \langle iv_3, \emptyset, \{iv_3 \leq 1\}\rangle}$$

$$\boxed{C_6 : \langle iv_5, \emptyset, \{iv_5 \leq 1\}\rangle}$$

$$\boxed{C_{(5.1 \vee 5.2 \vee 6)^{+}} : \langle\rangle}$$

Second Step: (2) Bound Computation of Phase of whA()

- Define intermediate variables representing the various sums:

$$iv_6 = \sum_{j=1}^{\#c_{5.2}} (iv_{2j}) \quad iv_7 = \sum_{j=1}^{\#c_{5.1}} (iv_{3j}) \quad iv_8 = \sum_{j=1}^{\#c_{5.2}} (iv_{4j}) \quad iv_9 = \sum_{j=1}^{\#c_6} (iv_{5j})$$

$$5.1 : whA(l, s) = 1 + \text{popSome}[7](s'', s') + whA(l', s')$$
$$\{l \geq 1,\ s \geq 0,\ s'' = s + 1,\ l' = l - 1,\ s'' = s'\} = \varphi_{5.1}$$

$$5.2 : whA(l, s) = 1 + \text{popSome}[(8)^{+}7](s'', s') + whA(l', s')$$
$$\{l \geq 1,\ s \geq 0,\ s'' = s + 1,\ l' = l - 1,\ s'' > s'\} = \varphi_{5.2}$$

$$6 : whA(l, s) = 1 + whA(l', s')\ \{l \geq 1,\ s \geq 0,\ s' = s + 1,\ l' = l - 1\} = \varphi_6$$

$$\boxed{C_{5.2} : \langle iv_2 + iv_4, \emptyset, \{iv_4 \leq 1, iv_2 \leq |s + 1 - s'|\}\rangle} \quad \boxed{C_{5.1} : \langle iv_3, \emptyset, \{iv_3 \leq 1\}\rangle}$$

$$\boxed{C_6 : \langle iv_5, \emptyset, \{iv_5 \leq 1\}\rangle}$$

$$\boxed{C_{(5.1 \vee 5.2 \vee 6)^+} : \langle \mathbf{iv_6} + \mathbf{iv_7} + \mathbf{iv_8} + \mathbf{iv_9}, \emptyset, \{\}\rangle}$$

- Define intermediate variables representing the various sums:

$$iv_6 = \sum_{j=1}^{\#c_{5.2}} (iv_{2j}) \quad iv_7 = \sum_{j=1}^{\#c_{5.1}} (iv_{3j}) \quad iv_8 = \sum_{j=1}^{\#c_{5.2}} (iv_{4j}) \quad iv_9 = \sum_{j=1}^{\#c_{6}} (iv_{5j})$$

- Use heuristics to obtain constraints for these variables:
  - Inductive sum for iv_6: obtain symbolic linear expression satisfying:
    $\varphi_{5.2} \Rightarrow L(i,n,r) \geq (\mathbf{s}+\mathbf{1}-\mathbf{s}') \wedge L(i,n,r) \geq (\mathbf{s}+\mathbf{1}-\mathbf{s}') + L(i',n',r')$

$$5.1 : whA(l,s) = 1 + \text{popSome}[7](s'',s') + whA(l',s')$$
$$\{l \geq 1,\ s \geq 0,\ s'' = s+1,\ l' = l-1,\ s'' = s'\} = \varphi_{5.1}$$

$$5.2 : whA(l,s) = 1 + \text{popSome}[(8)^{+}7](s'',s') + whA(l',s')$$
$$\{\mathbf{l} \geq \mathbf{1},\ \mathbf{s} \geq \mathbf{0},\ \mathbf{s}'' = \mathbf{s}+\mathbf{1},\ \mathbf{l}' = \mathbf{l}-\mathbf{1},\ \mathbf{s}'' > \mathbf{s}'\} = \varphi_{\mathbf{5.2}}$$

$$6 : whA(l,s) = 1 + whA(l',s')\ \{l \geq 1,\ s \geq 0,\ s' = s+1,\ l' = l-1\} = \varphi_6$$

$\boxed{C_{5.2} : \langle iv_2 + iv_4, \emptyset, \{iv_4 \leq 1, iv_2 \leq |s+1-s'|\}\rangle}$ $\boxed{C_{5.1} : \langle iv_3, \emptyset, \{iv_3 \leq 1\}\rangle}$

$\boxed{C_6 : \langle iv_5, \emptyset, \{iv_5 \leq 1\}\rangle}$

$\boxed{C_{(5.1 \vee 5.2 \vee 6)^+} : \langle iv_6 + iv_7 + iv_8 + iv_9, \emptyset, \{\}\rangle}$

Second Step: (2) Bound Computation of Phase of whA()

- Define intermediate variables representing the various sums:

$$iv_6 = \sum_{j=1}^{\#c_{5.2}} (iv_{2j}) \quad iv_7 = \sum_{j=1}^{\#c_{5.1}} (iv_{3j}) \quad iv_8 = \sum_{j=1}^{\#c_{5.2}} (iv_{4j}) \quad iv_9 = \sum_{j=1}^{\#c_6} (iv_{5j})$$

- Use heuristics to obtain constraints for these variables:
  - Inductive sum for iv_6: obtain symbolic linear expression satisfying:
    $\varphi_{5.2} \Rightarrow L(i,n,r) \geq (\mathbf{s}+\mathbf{1}-\mathbf{s}') \wedge L(i,n,r) \geq (\mathbf{s}+\mathbf{1}-\mathbf{s}') + L(i',n',r')$
  - Solution: $\mathbf{l}+\mathbf{s}$ remains unchanged in CEs 5.1 and 6

$$5.1 : whA(l,s) = 1 + \text{popSome}[7](s'',s') + whA(l',s')$$
$$\{l \geq 1,\ s \geq 0,\ \mathbf{s}'' = \mathbf{s}+\mathbf{1},\ \mathbf{l}' = \mathbf{l}-\mathbf{1},\ \mathbf{s}'' = \mathbf{s}'\} = \varphi_{\mathbf{5.1}}$$

$$5.2 : whA(l,s) = 1 + \text{popSome}[(8)^{+}7](s'',s') + whA(l',s')$$
$$\{l \geq 1,\ s \geq 0,\ s'' = s+1,\ l' = l-1,\ s'' > s'\} = \varphi_{5.2}$$

$$6 : whA(l,s) = 1 + whA(l',s')\ \{l \geq 1,\ s \geq 0,\ \mathbf{s}' = \mathbf{s}+\mathbf{1},\ \mathbf{l}' = \mathbf{l}-\mathbf{1}\} = \varphi_{\mathbf{6}}$$

$\boxed{C_{5.2} : \langle iv_2 + iv_4, \emptyset, \{iv_4 \leq 1, iv_2 \leq |s+1-s'|\}\rangle}$ $\boxed{C_{5.1} : \langle iv_3, \emptyset, \{iv_3 \leq 1\}\rangle}$

$\boxed{C_6 : \langle iv_5, \emptyset, \{iv_5 \leq 1\}\rangle}$

$\boxed{C_{(5.1 \vee 5.2 \vee 6)^+} : \langle iv_6 + iv_7 + iv_8 + iv_9, \emptyset, \{iv_6 \leq |l+s|\}\rangle}$

- Define intermediate variables representing the various sums:

$$iv_6 = \sum_{j=1}^{\#c_{5.2}} (iv_{2j}) \quad iv_7 = \sum_{j=1}^{\#c_{5.1}} (iv_{3j}) \quad iv_8 = \sum_{j=1}^{\#c_{5.2}} (iv_{4j}) \quad iv_9 = \sum_{j=1}^{\#c_6} (iv_{5j})$$

- Use heuristics to obtain constraints for these variables:
  - Inductive sum for iv_7 yields $\mathbf{l}$ as bound

$$\begin{aligned} 5.1: &\ whA(l,s) = 1 + \text{popSome}[7](s'',s') + whA(l',s') \\ &\ \{\mathbf{l \geq 1, s \geq 0, s'' = s+1, l' = l-1, s'' = s'}\} = \varphi_{\mathbf{5.1}} \end{aligned}$$

$$\begin{aligned} 5.2: &\ whA(l,s) = 1 + \text{popSome}[(8)^{+}7](s'',s') + whA(l',s') \\ &\ \{l \geq 1,\ s \geq 0,\ s'' = s+1,\ l' = l-1,\ s'' > s'\} = \varphi_{5.2} \end{aligned}$$

$$6: whA(l,s) = 1 + whA(l',s')\ \{l \geq 1,\ s \geq 0,\ s' = s+1,\ l' = l-1\} = \varphi_6$$

$$\boxed{C_{5.2} : \langle iv_2 + iv_4, \emptyset, \{iv_4 \leq 1, iv_2 \leq |s+1-s'|\}\rangle} \quad \boxed{C_{5.1} : \langle iv_3, \emptyset, \{\cancel{iv_3 \leq 1}\}\rangle}$$

$$\boxed{C_6 : \langle iv_5, \emptyset, \{iv_5 \leq 1\}\rangle}$$

$$\boxed{C_{(5.1 \vee 5.2 \vee 6)^+} : \langle iv_6 + iv_7 + iv_8 + iv_9, \emptyset, \{iv_6 \leq |l+s|\}\rangle}$$

- Define intermediate variables representing the various sums:

$$iv_6 = \sum_{j=1}^{\#c_{5.2}} (iv_{2j}) \quad iv_7 = \sum_{j=1}^{\#c_{5.1}} (iv_{3j}) \quad iv_8 = \sum_{j=1}^{\#c_{5.2}} (iv_{4j}) \quad iv_9 = \sum_{j=1}^{\#c_{6}} (iv_{5j})$$

- Use heuristics to obtain constraints for these variables:
  - Inductive sum for iv_7 yields $\mathbf{l}$ as bound
  - Same bound obtained for iv_8, iv_9

$$5.1 : whA(l,s) = 1 + \text{popSome}[7](s'', s') + whA(l', s')$$
$$\{l \geq 1,\ s \geq 0,\ s'' = s+1,\ l' = l-1,\ s'' = s'\} = \varphi_{5.1}$$

$$5.2 : whA(l,s) = 1 + \text{popSome}[(8)^{+}7](s'', s') + whA(l', s')$$
$$\{\mathbf{l} \geq \mathbf{1},\ \mathbf{s} \geq \mathbf{0},\ \mathbf{s}'' = \mathbf{s}+\mathbf{1},\ \mathbf{l}' = \mathbf{l}-\mathbf{1},\ \mathbf{s}'' > \mathbf{s}'\} = \varphi_{\mathbf{5.2}}$$

$$6 : whA(l,s) = 1 + whA(l', s')\ \{\mathbf{l} \geq \mathbf{1},\ \mathbf{s} \geq \mathbf{0},\ \mathbf{s}' = \mathbf{s}+\mathbf{1},\ \mathbf{l}' = \mathbf{l}-\mathbf{1}\} = \varphi_{\mathbf{6}}$$

$$\boxed{C_{5.2} : \langle iv_2 + iv_4, \emptyset, \{iv_4 \leq 1, iv_2 \leq |s+1-s'|\}\rangle} \quad \boxed{C_{5.1} : \langle iv_3, \emptyset, \{iv_3 \leq 1\}\rangle}$$

$$\boxed{C_6 : \langle iv_5, \emptyset, \{iv_5 \leq 1\}\rangle}$$

$$\boxed{C_{(5.1\vee 5.2\vee 6)^+} : \langle iv_6 + iv_7 + iv_8 + iv_9, \emptyset, \{iv_6 \leq |l+s|\}\rangle}$$

Second Step: (2) Bound Computation of Phase of whA()

- Define intermediate variables representing the various sums:

$$iv_6 = \sum_{j=1}^{\#c_{5.2}} (iv_{2j}) \quad iv_7 = \sum_{j=1}^{\#c_{5.1}} (iv_{3j}) \quad iv_8 = \sum_{j=1}^{\#c_{5.2}} (iv_{4j}) \quad iv_9 = \sum_{j=1}^{\#c_6} (iv_{5j})$$

- Use heuristics to obtain constraints for these variables:
  - Inductive sum for iv_7 yields $\mathbf{l}$ as bound
  - Same bound obtained for iv_8, iv_9 and even sum of all three

$$5.1 : whA(l,s) = 1 + \text{popSome}[7](s'', s') + whA(l', s')$$
$$\{l \geq 1,\ s \geq 0,\ s'' = s+1,\ l' = l-1,\ s'' = s'\} = \varphi_{5.1}$$

$$5.2 : whA(l,s) = 1 + \text{popSome}[(8)^{+}7](s'', s') + whA(l', s')$$
$$\{l \geq 1,\ s \geq 0,\ s'' = s+1,\ l' = l-1,\ s'' > s'\} = \varphi_{5.2}$$

$$6 : whA(l,s) = 1 + whA(l', s')\ \{l \geq 1,\ s \geq 0,\ s' = s+1,\ l' = l-1\} = \varphi_6$$

$$\boxed{C_{5.2} : \langle iv_2 + iv_4, \emptyset, \{iv_4 \leq 1, iv_2 \leq |s+1-s'|\}\rangle} \quad \boxed{C_{5.1} : \langle iv_3, \emptyset, \{iv_3 \leq 1\}\rangle}$$

$$\boxed{C_6 : \langle iv_5, \emptyset, \{iv_5 \leq 1\}\rangle}$$

$$\boxed{C_{(5.1 \vee 5.2 \vee 6)^+} : \langle iv_6 + iv_7 + iv_8 + iv_9, \emptyset, \{iv_6 \leq |l+s|,\ iv_7 + iv_8 + iv_9 \leq |l|\}\rangle}$$

- Define intermediate variables representing the various sums:

$$iv_6 = \sum_{j=1}^{\#c_{5.2}} (iv_{2j}) \quad iv_7 = \sum_{j=1}^{\#c_{5.1}} (iv_{3j}) \quad iv_8 = \sum_{j=1}^{\#c_{5.2}} (iv_{4j}) \quad iv_9 = \sum_{j=1}^{\#c_{6}} (iv_{5j})$$

- Use heuristics to obtain constraints for these variables:
  - Inductive sum for iv_7 yields **l** as bound
  - Same bound obtained for iv_8, iv_9 and even sum of all three

$$5.1 : whA(l,s) = 1 + \text{popSome}[7](s'', s') + whA(l', s')$$
$$\{l \geq 1,\ s \geq 0,\ s'' = s+1,\ l' = l-1,\ s'' = s'\} = \varphi_{5.1}$$

$$5.2 : whA(l,s) = 1 + \text{popSome}[(8)^+7](s'', s') + whA(l', s')$$
$$\{l \geq 1,\ s \geq 0,\ s'' = s+1,\ l' = l-1,\ s'' > s'\} = \varphi_{5.2}$$

$$6 : whA(l,s) = 1 + whA(l', s')\ \{l \geq 1,\ s \geq 0,\ s' = s+1,\ l' = l-1\} = \varphi_6$$

$\boxed{C_{5.2} : \langle iv_2}$ ... $\boxed{+1 - s'|\}\rangle}$ $\boxed{C_{5.1} : \langle iv_3, \emptyset, \{iv_3 \leq 1\}\rangle}$

$\boxed{C_6 : \langle iv_5, \emptyset}$

UB: $|l+s| + |l|$

$\boxed{C_{(5.1 \vee 5.2 \vee 6)^+} : \langle iv_6 + iv_7 + iv_8 + iv_9, \emptyset, \{iv_6 \leq |l+s|,\ iv_7 + iv_8 + iv_9 \leq |l|\}\rangle}$

Implementation and Experiments

- All techniques implemented in CoFloCo ($\Rightarrow$ complexity competition)
- Automatically obtained precise bounds for executable, concurrent model (in ABS) of a distributed genetic algorithm
- Essential features to deal with realistic code:
  - Control flow refinement of cost relations
  - Ability to maintain multiple bound candidates in a cost structure
- In many applications amortized bounds very useful: code over (functional) data structures
- Drawback of control flow refinement: exponential blow-up possible
  - Employ path/chain merging to mitigate that problem

Main Elements of the Presented Approach

Control Flow Refinement

- Separate bounds for computation paths with differing behavior
- Derive explicit preconditions for these: document, optimize

Propagation of Invariants/Summaries, Bottom-Up Strategy

- Strengthen application conditions
- Take return values into account: amortized cost

Separation of Concerns in Cost Structures, Intermediate Variables

- Maintain multiple candidates for bounds
- Enable incremental bound inference
- Plug in induction heuristics to infer closed bounds
- Represent non-linear bounds
- Handle loops with resets (see paper)

Summary

- Overcame existing limitations of resource analysis with cost relations:
  - Multi-phase loops
  - Loops with resets
  - Amortized cost
- Increase precision of derived bounds
- Presented techniques are language independent:
  imperative and functional programs
- Can also infer (best case) lower bounds
- Extendable to logarithmic or exponential bounds

References

Albert, E., Arenas, P., Genaim, S., Puebla, G., and Zanardini, D. (2007).
Cost analysis of Java Bytecode.
In *Proc. 16th European Symp. on Programming*, ESOP, pages 157–172. Springer.

Debray, S. K. and Lin, N. W. (1993).
Cost Analysis of Logic Programs.
ACM Transactions on Programming Languages and Systems, 15(5):826–875.

Vasconcelos, P. B. and Hammond, K. (2004).
Inferring cost equations for recursive, polymorphic and higher-order functional programs.
In *Proc. 15th Intl. Conf. on Implementation of Functional Languages*, IFL, pages 86–101. Springer.

Wegbreit, B. (1975).
Mechanical Program Analysis.
Communications of the ACM, 18(9):528–539.