Termination

BubbleSort: Partial Correctness

If BubbleSort terminates, then it returns a sorted array.

```
@pre ⊤
@post sorted(rv, 0, |a| − 1)
int[] BubbleSort(int[] a₀) {
  int[] a := a₀;
  for (int i := |a| − 1; i > 0; i := i − 1) {
    for (int j := 0; j < i; j := j + 1) {
      if (a[j] > a[j + 1]) {
        int t := a[j];
        a[j] := a[j + 1];
        a[j + 1] := t;
      }
    }
  }
  return a;
}
```

BubbleSort: Termination

Does BubbleSort always terminate?

```
@pre ⊤
int[] BubbleSort(int[] a_0) {
  int[] a := a_0;
  for (int i := |a| − 1; i > 0; i := i − 1) {
    for (int j := 0; j < i; j := j + 1) {
      if (a[j] > a[j + 1]) {
        int t := a[j];
        a[j] := a[j + 1];
        a[j + 1] := t;
      }
    }
  }
  return a;
}
```

BubbleSort: Termination

Well-founded $(\underbrace{\mathbb{Z}^+ \times \mathbb{Z}^+}_{S},\ \underbrace{<}_{\prec})$

```
@pre ⊤
int[] BubbleSort(int[] a₀) {
  int[] a := a₀;
  for @L₁ :    (i + 1 ≥ 0) ↓ (i + 1, i + 1)
  (int i := |a| − 1; i > 0; i := i − 1)
    for @L₂ :    (i − j ≥ 0 ∧ i + 1 ≥ 0) ↓ (i + 1, i − j)
    (int j := 0; j < i; j := j + 1)
      if (a[j] > a[j + 1]) {
        int t := a[j];
        a[j] := a[j + 1];
        a[j + 1] := t;
      }
  return a;
}
```

L_1: $\underbrace{(i+1 \geq 0)}_{\mu_1} \downarrow \underbrace{(i+1,\ i+1)}_{\delta_1}$

L_2: $\underbrace{(i-j \geq 0\ \wedge\ i+1 \geq 0)}_{\mu_2} \downarrow \underbrace{(i+1,\ i-j)}_{\delta_2}$

μ_i: supporting invariant

δ_i: ranking function

BubbleSort: Supporting Invariants

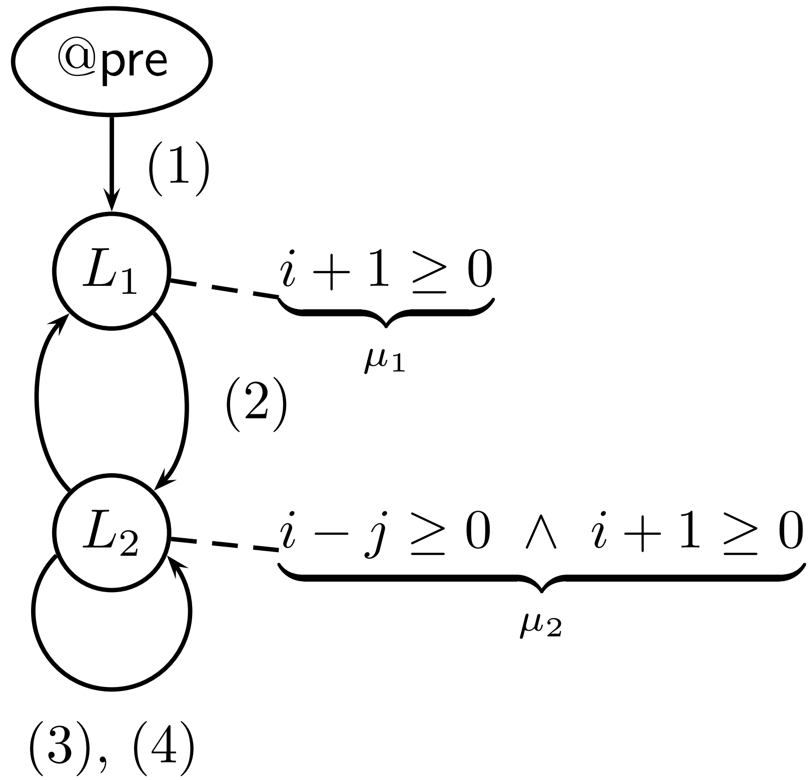
Path (5)

$@L_2 : i{-}j \geq 0 \wedge i{+}1 \geq 0$

assume $j \geq i$;

$i := i - 1$;

$@L_1 : i + 1 \geq 0$



$$\underbrace{i - j \geq 0 \ \wedge \ i + 1 \geq 0}_{\mu_2} \wedge j \geq i \ \Rightarrow \ \underbrace{(i - 1) + 1 \geq 0}_{\mu_1'}$$

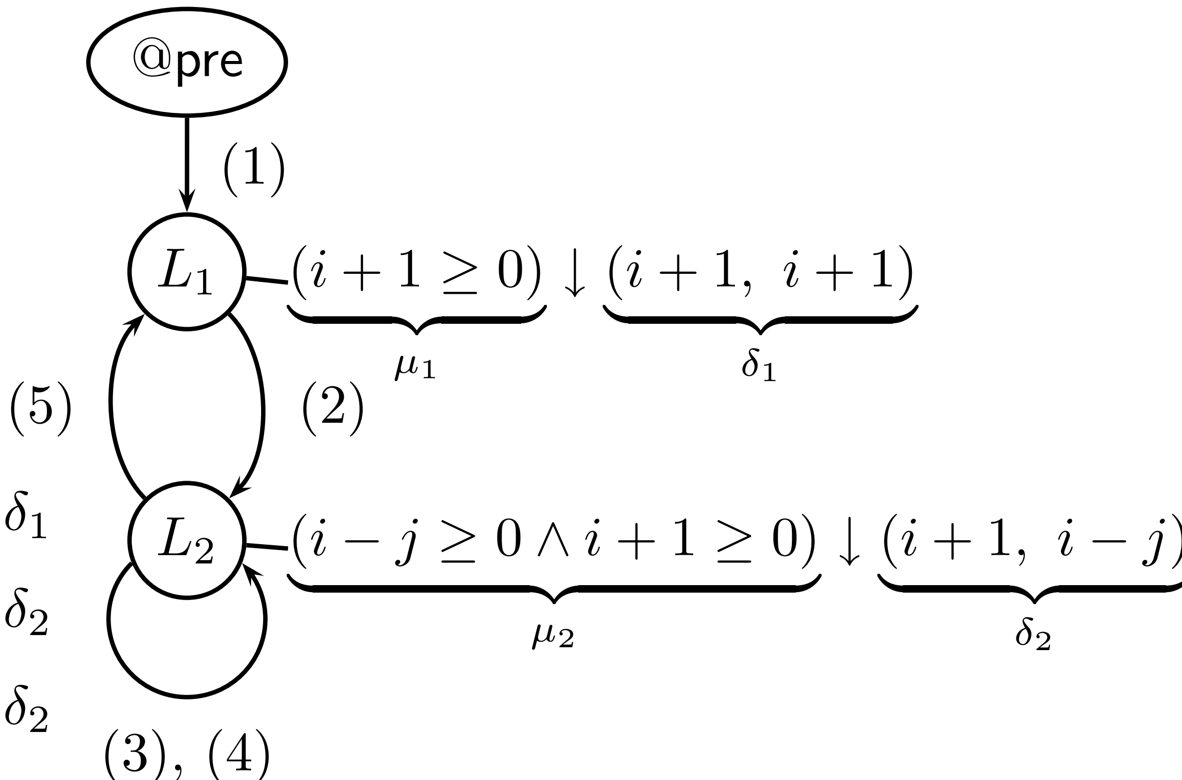
BubbleSort: Ranking Functions

Bounded below:

- $\mu_1 \;\Rightarrow\; \delta_1 \geq 0$
- $\mu_2 \;\Rightarrow\; \delta_2 \geq 0$

Decreasing:

- $\mu_1 \wedge \pi_2 \;\Rightarrow \delta_2' < \delta_1$
- $\mu_2 \wedge \pi_3 \;\Rightarrow \delta_2' < \delta_2$
- $\mu_2 \wedge \pi_4 \;\Rightarrow \delta_2' < \delta_2$
- $\mu_2 \wedge \pi_5 \;\Rightarrow \delta_1' < \delta_2$



BubbleSort: Bounded Below

- $\underbrace{i+1 \geq 0}_{\mu_1} \Rightarrow \underbrace{(i+1,\ i+1)}_{\delta_1} \geq 0$
- $\underbrace{i-j \geq 0 \ \wedge \ i+1 \geq 0}_{\mu_2} \Rightarrow \underbrace{(i+1,\ i-j)}_{\delta_2} \geq 0$

BubbleSort: Decreasing

Path 2

$@L_1 : (i+1 \geq 0) \downarrow (i+1,\ i+1)$

assume $i > 0$;

$j := 0$;

$@L_2 : (i - j \geq 0\ \wedge\ i+1 \geq 0) \downarrow (i+1,\ i-j)$

$$\underbrace{i+1 \geq 0}_{\mu_1} \wedge i > 0 \ \Rightarrow\ \underbrace{(i+1,\ i-0)}_{\delta_2'} < \underbrace{(i+1,\ i+1)}_{\delta_1}$$

BubbleSort: Decreasing

Path 3/4

$@L_2 : (i - j \geq 0 \ \wedge \ i + 1 \geq 0) \downarrow (i + 1, \ i - j)$

$\mathsf{assume}\ j < i;$

$\vdots$

$j := j + 1;$

$@L_2 : (i - j \geq 0 \ \wedge \ i + 1 \geq 0) \downarrow (i + 1, \ i - j)$

$$\underbrace{i - j \geq 0 \ \wedge \ i + 1 \geq 0}_{\mu_2} \wedge j < i \ \Rightarrow$$

$$\underbrace{(i + 1, \ i - (j + 1))}_{\delta_2'} < \underbrace{(i + 1, \ i - j)}_{\delta_2}$$

BubbleSort: Decreasing

Path 5

$@L_2 : (i - j \geq 0 \;\wedge\; i + 1 \geq 0) \downarrow (i + 1,\; i - j)$

assume $j \geq i$;

$i := i - 1$;

$@L_1 : (i + 1 \geq 0) \downarrow (i + 1,\; i + 1)$

$$\underbrace{i - j \geq 0 \;\wedge\; i + 1 \geq 0}_{\mu_2} \wedge j \geq i \;\Rightarrow$$
$$\underbrace{((i - 1) + 1,\; (i - 1) + 1)}_{\delta_1'} < \underbrace{(i + 1,\; i - j)}_{\delta_2}$$

BinarySearch: Partial Correctness

If array a is sorted and BinarySearch terminates, then it returns $\top$ iff it contains e in positions $[\ell, u]$.

```
@pre 0 ≤ ℓ ∧ u < |a| ∧ sorted(a, 0, |a| − 1)
@post rv ↔ ∃i. ℓ ≤ i ≤ u ∧ a[i] = e
bool BSearch(int[] a, int ℓ, int u, int e) {
  if (ℓ > u) return false;
  else {
    int m := (ℓ + u) div 2;
    if (a[m] = e) return true;
    else if (a[m] < e) return BSearch(a, m + 1, u, e);
    else return BSearch(a, ℓ, m − 1, e);
  }
}
```

BinarySearch: Termination

Does BinarySearch terminate for ℓ, u, and a satisfying the precondition?

```
@pre 0 ≤ ℓ ∧ u < |a|
bool BSearch(int[] a, int ℓ, int u, int e) {
  if (ℓ > u) return false;
  else {
    int m := (ℓ + u) div 2;
    if (a[m] = e) return true;
    else if (a[m] < e) return BSearch(a, m + 1, u, e);
    else return BSearch(a, ℓ, m − 1, e);
  }
}
```

BinarySearch: Termination

Well-founded $(\mathbb{Z}^+, <)$

```
@pre u − ℓ + 1 ≥ 0
@post ⊤
↓ (u − ℓ + 1)
bool BSearch(int[] a, [int ℓ, int u], int e) {
  if (ℓ > u) return false;
  else {
    int m := (ℓ + u) div 2;
    if (a[m] = e) return true;
    else if (a[m] < e) return BSearch(a, [m + 1, u], e);
    else return BSearch(a, [ℓ, m − 1], e);
  }
}
```

BinarySearch: Termination

$$\ell \leq u \;\Rightarrow\; u - \underbrace{(((\ell + u) \texttt{ div } 2) + 1)}_{m+1} + 1 < u - \ell + 1$$

$$\begin{aligned} \ell &= (\ell + \ell) \texttt{ div } 2 \\ &\leq (\ell + u) \texttt{ div } 2 \qquad \text{since } \ell \leq u \\ &< (\ell + u) \texttt{ div } 2 + 1 \end{aligned}$$

BinarySearch: Termination

$$\ell \leq u \;\Rightarrow\; \underbrace{(((\ell + u) \texttt{ div } 2) - 1)}_{m-1} - \ell + 1 < u - \ell + 1$$

$$\begin{aligned} u &= (u + u) \texttt{ div } 2 \\ &\geq (\ell + u) \texttt{ div } 2 \qquad \text{since } \ell \leq u \\ &> (\ell + u) \texttt{ div } 2 - 1 \end{aligned}$$

BinarySearch: Termination

Path 1:

@pre $(u - \ell + 1 \geq 0) \downarrow (u - \ell + 1)$

assume $\ell \leq u$;

$m := (\ell + u)$ `div` 2;

assume $a[m] \neq e$;

assume $a[m] < e$;

@pre $(u - \ell + 1 \geq 0)\{\ell \mapsto m + 1\} \downarrow (u - \ell + 1)\{\ell \mapsto m + 1\}$

$$\ell \leq u \;\Rightarrow\; u - (((\ell + u) \texttt{ div } 2) + 1) + 1 < u - \ell + 1$$