SEMANTYKA I WERYFIKACJA - ĆW. 14

(wersja bez Haha)

Kontynuujemy dowodzenie poprawności częściowej programów.

1. **Zadanie:** Przeprowadź dowód poprawności częściowej programu:

   $\{n \geq 1 \wedge (\forall x.1 \leq x < n \Longrightarrow A[x] \leq A[x+1]) \wedge (\exists x.1 \leq x < n \wedge A[x] = v)\}$

   ```
   i := 1; j := n;
   while i < j do
     k := (i + j) div 2;
     if A[k] < v then
       i := k + 1
     else
       j := k
   ```

   $\{A[i] = v\}$

   Rozwiązanie: Wstępna wersja niezmiennika pętli to:

   $$\phi \quad \equiv \quad i \leq j \wedge (\exists x.i \leq x \leq j \wedge A[x] = v)$$

   Nie jest jasne co wpisać po przypisaniu na `k`; pomysł to

   $$\phi \wedge (i \leq k \leq j)$$

   Niestety to nie działa! W niezmienniku musimy jeszcze przepychać dwie informacje:

   - że tablica jest posortowana, i
   - że i i j mieszczą się w zakresie tablicy.

2. **Zadanie:** Dana jest tablica dwuwymiarowa $A[1..n, 1..n]$ zawierająca liczby 0 i 1, reprezentująca relację binarną *zna* między n osobami. Powiemy, że i jest *znakomitością* jeśli:

   - każdy oprócz i zna i,
   - i nie zna nikogo poza i.

   Napisz program znajdujący znakomitość, o ile istnieje, i dowiedź częściowej poprawności.

   Rozwiązanie: W czasie liniowym:

   ```
   i := 1; j := n;
   while i <> j do
     if A[i, j] = 1 then
       i := i + 1
     else
       j := j - 1
   ```

Niezmiennik:

$i \leq j \wedge (\exists k. 1 \leq k \leq n \wedge k \text{ jest znakomitością}) \Longrightarrow (\exists k. i \leq k \leq j \wedge k \text{ jest znakomitością})$

Uwaga: Wszelkie nietrywialne rozumowania najlepiej umieszczać jawnie jako pary asercji między tymi samymi instrukcjami i korzystać z reguły osłabiania, np:

$\{N \wedge i \neq j \wedge i \ \ zna \ \ j\}$
$\Downarrow$
$\{N[i \mapsto i+1]\}$
`i := i+1`
$\{N\}$

3. Oto insertion sort:

```
i := 1;
while i <= n do
  j := 1;
  while j <= i do
    if A[i]>A[j] then
      t := A[i]; A[i] := A[j]; A[j] := t
```

Uwaga: aby udowodnić, że tablica po posortowaniu zawiera te same elementy co na początku, należy dodać niejawną zmienną tablicową `B` i dodać precondition: $\forall i = 1..n. A[i] = B[i]$.

A oto incredible sort:

```
i := 1;
while i <= n do
  j := 1;
  while j <= n do
    if A[i]<A[j] then
      t := A[i]; A[i] := A[j]; A[j] := t
```

4. **Zadanie:** Napisz program obliczający największy wspólny dzielnik dwóch liczb i przeprowadź dowód poprawności częściowej.