Werkcollege Compilerconstructie
Dinsdag 11 november 2014

1. We bekijken een algoritme om het minimale element in een array met integers te verplaatsen naar positie 0 in het array[1]:

```
i = 0;
min = a[i];
while (i < n-1)
{ i = i+1;
  if (a[i] < min)
  { a[0] = a[i];
    a[i] = min;
    min = a[0];
  }
}
```

   Als we aannemen dat een integer vier bytes in beslag neemt, kan een rechttoe-rechtaan vertaling van dit algoritme in drie-adres code het volgende opleveren:

```
(1)  i = 0
(2)  t1 = 4*i
(3)  min = a[t1]
(4)  t2 = n-1
(5)  if i < t2 goto 7
(6)  goto 21
(7)  i = i+1
(8)  t3 = 4*i
(9)  t4 = a[t3]
(10) if t4 < min goto 12
(11) goto 20
(12) t5 = 4*0
(13) t6 = 4*i
(14) t7 = a[t6]
(15) a[t5] = t7
(16) t8 = 4*i
(17) a[t8] = min
(18) t9 = 4*0
(19) min = a[t9]
(20) goto 4
(21) ...
```

   (a) Welke instructies in bovenstaande drie-adres code zijn de leaders?

[1] Dit algoritme is bijvoorbeeld te gebruiken voor SelectionSort.

(b) Teken de *flow graph* met de basic blocks bij regels 1–20 van bovenstaande drie-adres code. Nummer de basic blocks $B_1, B_2, \ldots$.

(c) Je moet de drie-adres code in regels 1–20 hierboven nu stapsgewijs gaan optimaliseren. Bij iedere stap moet je kort aangeven wat je doet, en moet je voor de basic blocks die bij die stap veranderen de complete nieuwe blokken geven. Je mag gebruik maken van de volgende soorten transformaties (voor zover ze te gebruiken zijn):

- *constant folding*
- *local common-subexpression elimination*
- *global common-subexpression eliminiation*
- *copy propagation*
- *dead-code elimination*
- *code motion*
- *reduction in strength*
- *induction-variable elimination*

Noem bij iedere stap het soort transformatie dat je gebruikt hebt. Geef ook de complete nieuwe flow graph die het eindresultaat is van de optimalisatiestappen.

(d) Zijn er voor de drie-adres code uit deze opgave nog andere soorten optimalisaties mogelijk dan degene die je bij het vorige onderdeel mocht gebruiken? Zo ja, welke en waar in de code.