KOI - Lab 7

Anton Soderlund

February 12, 2020

1 Task A

Denna implementation finns även i Task B. Jag studerade exempelprogrammet och byggde min stackfunktion i *parser.ypp* i samma stil som printfunktionerna.

```
void build_stack_from_tree(TreeNode *p)
void build_stack_from_tree2(TreeNode *p,bool left)
```

Den första funktionen initierar stacken, startar den rekursiva traverseringen genom trädet och sen printar/kör stacken efteråt. Den andra funktion är den rekursiva funktionen som tar ett extra argument för att hålla koll på vilka värdet som är lvalues resp. rvalues. Sedan kontrolleras flödet med en switch-sats, liknande printfunktionen.
Varje case appendar den passande instruktionen och lägger den på stacken, och sedan rekurserar den längre ner i trädet. Det som var lite komplicerat var att se till att labels, jump och goto instruktioner hamnar i rätt positioner, i t.ex. while-satsen. Jag gjorde en testkörning med exemplet från lab 5 och det såg korrekt ut, förutom att när jag kör stacken med sm.run(), stackop_read verkar inte exekveras ordentligt. Om man byter ut read(n) med n=5, så bygger den stacken korrekt.

2 Task B

Denna uppgiften gjorde jag med en ny rekursiv funktion.

```
void init_optimizer(TreeNode *p)
TreeNode *optimize(TreeNode *p)
```

Den skapar en ny rotnod för det nya optimerade trädet *opti_p*, och skickar in rotnoden för det första trädet i den rekursiva funktionen. Den börjar med att traversera ner längst ner i trädet, vilket sedan ledar in i switch-satsen som styr ifall det finns något att optimera.På det sättet optimerar vi alltid längst ner först. Den nya noden returnas sedan som argument till den tidigare noden som argument.
Optimeringsfunktionen förenklar endast konstanta uttryck vilket inkluderar uttrycken, **[k + 0, k * 0, k * 1, k < k, k > k]** för en konstant k inkl. den

kommutativa motsvarigheten för dessa uttryck. If-satser använder nu dessa konstanta beräkningar (ifall de är konstanta) och byter if-noden till det argument som motsvarar till villkoret. Om villkoret är sant byts noden ut till arg[1], och till arg[2] ifall villkoret är falskt, dvs hela if-noden skärs bort ur trädet.

Utöver dessa funktioner finns det även ett par hjälpfunktioner så att det går att hämta värdet från symboltabellen för variabler, eller direkt från leaf_value ifall det är konstanta värden.