10. Project 1: Syntax and axiomatic semantics

Wed. 1 October

Due date: Wed. 15 October

Informal language definition

a. Consider a simple block structured language that provides the following constructs:

   - simple integer identifiers
   - arithmetic expressions involving only addition and subtraction over integers (when used as predicates a strictly positive value is treated as *true* and zero or negative values as *false*)
   - assignment statement to a single variable
   - statement sequence
   - conditional **if-then-else-fi**
   - parallel execution construct **cobegin-coend** (e.g., **cobegin** x:=x+1 **co** y:=y-z **coend**) which allows each branch to execute on a separate processor in parallel and without any interactions among the branches; the construct is exited when all branches terminate; **cobegin-coend** constructs cannot be nested but may have arbitrary number of branches

b. Assume the availability of a function τ which, when presented with an expression (e.g., x+y-3) returns a count of the number of additions and subtractions appearing in the expression (e.g., 2 for the earlier expression).

c. Assume that the time it takes to execute a strictly sequential program equals the number of additions, subtractions, tests, and assignments performed during its execution. The number of processors available to execute a **cobegin-coend** block is assumed to equal the number of branches in the construct.

Assignment

a. Develop an axiomatic semantic model that allows you to reason both about the computation and the time it takes to execute it. Use T to denote the current execution time.

b. Follow the format indicated below.

c. If you are unable to complete the full assignment, solve the problem using a subset of the full language described above and adjust the program used in the verification section accordingly. You will receive partial credit.

Homework format

A. Cover Page (1 page)

- class
- project number
- project name
- date
- name
- statement (optional)

B. Language Syntax (1 page)

- brief informal overview of the language
- abstract syntax
- additional constraints not captured by the syntax
- example program

C. Language Semantics (2 pages)

- briefoverview identifying the model type and the general modeling strategy you plan to pursue; focus the presentation on the more subtle aspects of the problem (e.g., time, cobegin-coend)
- show formally how τ is computed for a given expression
- explain how to compute the *wp* for each construct in the language

D. Program Verification (1 page)

- formally derive the initial condition *P* for which you can prove

```
{ P ^ T=2 }
    cobegin
        if y-x then x:=y fi
    co
        if v-u then u:=v+0 fi
    coend
    if u-x then x:=x-u+x fi
{ T=6 ^ x=0 }
```

- T denotes the execution time
- for each statement simply show its precondition, you do not need to prove formally that the derivation of the precondition is correct