Assignment

CS 766 Programming Assignment

Assignment : Verifying program under sequential consistency

Deadline : 18 Feb 2022, 11:00 PM

1 Background

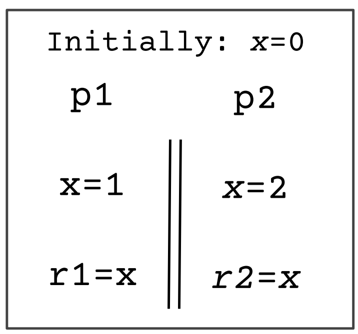

Figure 1: A sample program

Concurrent Program: Concurrent program **P** contains fixed number of processes $p_1, p_2, ...p_n$.. Each process contains `write` or `read` instructions. `write` instructions are of form $x = d$, where x is global variable and d is an integer. `read` instructions are of form $r = x$, where r is local register and x is global variable.

Sequential Execution An execution is sequence of instructions. An execution is sequential iff all read instructions read values from latest write instructions. For an execution E $e_1, e_2, ..., e_m$, where $e_i = \{$`write` or `read`$\}$ instruction , $in_E(e)$ denotes the index of instruction e in the Execution E.

An execution E is sequential iff (i) every read instruction, $r = x$, reads value of global variable x from the write instruction $x = d_1$ (this can be initial value, assume that all global variables are initialized with 0) where $in_E(r = x) < in_E(x = d_1)$, and (ii) if $r = x$ reads from $x = d_1$ then for all write instructions $x = d_2$ we have either $in_E(x = d_2) < in_E(x = d_1)$ or $in_E(r = x) < in_E(x = d_2)$.

```
Execution E
    x=2
    x=1
    r2=x//1
    r1=x//1
```

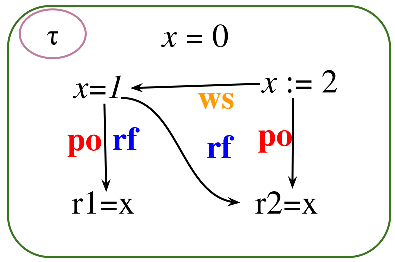

Figure 2: An execution and corresponding trace

Trace We can represent executions using traces. A trace τ is tuple (S_I, po, rf, ws), where S_I is set of instructions, po (program order), rf(read-from) and ws(write serialization also called coherence order) are binary relations on instructions in S_I. These relations are defined as follows:

- $e[po]e'$: instructions e and e' belong to same process p and e appear before e' in p. po totally orders instructions of each individual process.
- $e[rf]e'$: e is write instruction and e' is read instruction on the same variable and e' obtains its value form e. rf^{-1} is inverse of rf relation.
- $e[ws]e'$: ws(write serialization) relates two write instructions on same variable. $e[ws^x]e'$ denotes that e and e' are write instructions x. If $e[ws^x]e'$, then all processes agree that e happened-before e'. $ws = \bigcup_x ws^x$.
- We also define relation $fr = \bigcup_x fr^x$ where $fr^x = (rf^x)^{-1}; ws^x$.

Valid trace A trace $\tau = (S_I, po, rf, ws)$ is valid trace iff $(po \cup rf \cup ws \cup fr)$ is acyclic. Figure 2 gives an sequential execution and corresponding valid trace for the program given in figure 1.

2 Assignment

Your task will be to develop a tiny software tool that takes program as input and outputs all valid traces.

Input program can have assert statement in it. If there exists a sequential execution(valid trace) that violates the assert statement, then your tool should stop exploring traces and output the result as "Assertion violation" and display this trace.

Extras, Not mandatory: [You can optimize your algorithm by exploring only one trace per $rf-$equivalence class of traces. Two traces $\tau_1 = (S_I, po_1, rf_1, ws_1)$ and $\tau_2 = (S_I, po_2, rf_2, ws_2)$ are $rf-$equivalent if $po_1 = po_2$ and $rf_1 = rf_2$.]

Example 1 : Consider that program shown in figure 1 is input program. Output for this input program can be:
1: Trace: $[x = 1, r1 = x, x = 2, r2 = x]$, rf relation:$[[x = 1, r1 = x], [x = 2, r2 = x]]$, co relation:$[[x = 2, x = 1]]$
2: Trace: $[x = 1, r1 = x, x = 2, r2 = x]$, rf relation:$[[x = 1, r1 = x], [x = 2, r2 = x]]$, co relation:$[[x = 1, x = 2]]$
3: Trace: $[x = 1, x = 2, r1 = x, r2 = x]$, rf relation:$[[x = 2, r1 = x], [x = 2, r2 = x]]$, co relation:$[[x = 1, x = 2]]$
4: Trace: $[x = 2, x = 1, r1 = x, r2 = x]$, rf relation:$[[x = 1, r1 = x], [x = 1, r2 = x]]$, co relation:$[[x = 2, x = 1]]$
You can output traces and rf, ws relations in any order. But, ensure that each trace is valid trace and you do not repeat same trace(having same rf, ws relations).

Example 1 : Consider that program shown in figure 1 is input program and it has an assert statement, `assert(r2 != 1)`. Then your tool should output following:
`Assertion Violation :`
Violating Trace: $[x = 2, x = 1, r1 = x, r2 = x]$, rf relation:$[[x = 1, r1 = x], [x = 1, r2 = x]]$, co relation:$[[x = 2, x = 1]]$

2.1 Input Format:

First line of input is number of processes, n. Then next n lines contain instructions form the each process. Then next line will be integer A. If A is 1 then the next line will contain assert statement.

If A is 0 then there will be no assert statement for the given program. Assert statement can have logical 'and', 'or' operators, e.g. `assert`(`r2` != 1 and $r4$!= 2).

Input Example 1: The program given in the figure 1 will be presented in following format:

2
x=1;r1=x;
x=2;r2=x;
0

Input Example 2: The program given in the figure 1 with the assert statement, `assert`(`r2` != 1) will be presented in following format:

2
x=1;r1=x;
x=2;r2=x;
1
assert($r2$!= 1)

Input Example 3:

4
x=2;y=1;
y=2;z=1;
z=2;x=1;
r1=x;r2=y;r3=z;
1
assert($r1$!= 1 or $r2$!= 1 or $r3$!= 1)

2.2 Output Format

Output 1 One of the correct outputs for Input example 1.
No. of traces = 4
1-: Trace: $[x = 1, r1 = x, x = 2, r2 = x]$, rf relation:$[[x = 1, r1 = x], [x = 2, r2 = x]]$, co relation:$[[x = 2, x = 1]]$
2-: Trace: $[x = 1, r1 = x, x = 2, r2 = x]$, rf relation:$[[x = 1, r1 = x], [x = 2, r2 = x]]$, co relation:$[[x = 1, x = 2]]$
3-: Trace: $[x = 1, x = 2, r1 = x, r2 = x]$, rf relation:$[[x = 2, r1 = x], [x = 2, r2 = x]]$, co relation:$[[x = 1, x = 2]]$

4-: Trace: $[x = 2, x = 1, r1 = x, r2 = x]$, rf relation:$[[x = 1, r1 = x], [x = 1, r2 = x]]$, co relation:$[[x = 2, x = 1]]$

Output 2 Output for sample input 2.
Error: Assertion Violation
Violating Trace: $[x = 2, x = 1, r1 = x, r2 = x]$, rf relation:$[[x = 1, r1 = x], [x = 1, r2 = x]]$, co relation:$[[x = 2, x = 1]]$

2.3 Input Constraints

1. $1 \leq n \leq 10$.
2. Maximum number of instruction per process = 4
3. Fixed global variables: x , y, and z.
4. Maximum number of local registers in the input program= 10
5. Each read instruction can obtain its value from max 4 write instruction. So, if there is read instruction, $r = x$, then it can obtain value of x from max 4 write instructions on variable x.
6. Initially x=0 , y=0 , z=0. (Global valuables are initialized with zero.)

2.4 Task

Students will develop the tool in python. **Students can form group of 2-3 students to complete the assignment**. Along with the code, students need to submit a document(pdf file), describing their algorithm and approach followed. If you complete the assignment in group then mention name and roll no. of all members in the document.

Deadline: 18 Feb 2022, 11:00 PM

Submission Format: Students should develop the tool in python language. The driver file(which accepts the input program and outputs the traces) should be named as `trace.py`. You can create other files as per your approach(algorithm).
You need to submit the solution to assignment as the zip file, which

contains all python files(including driver file, `trace.py`) and pdf file(document where you describe your approach and algorithm). Name the zip file as $<\texttt{your}-\texttt{roll}-\texttt{no}>$.zip (for groups with 2-3 students, mention roll no. of the any member.)

Marks: Working tool: 14 marks, Document(pdf): 6 marks.