Sequential Nondeterminism

CS 536: Science of Programming, Fall 2021

A. *Why*

- Nondeterminism can help us avoid unnecessary determinism.
- Nondeterminism can help us develop programs without worrying about overlapping cases.

B. *Objectives*

At the end of these practice questions you should

- Be able to evaluate nondeterministic conditionals and loops.

C. *Nondeterminism*

1. Let $IF \equiv branch\ \{e_1 \rightarrow s_1 \square e_2 \rightarrow s_2 \square \ldots \square e_n \rightarrow s_n\}$ and $BB \equiv e_1 \vee e_2 \vee \ldots e_n$.
   a. What property does BB have to have for us to avoid a runtime error when executing IF?
   b. Does it matter if we reorder the guarded commands? (E.g., if we swap $e_1 \rightarrow s_1$ and $e_2 \rightarrow s_2$.)

2. Let $DO \equiv while\ \{e_1 \rightarrow s_1 \square e_2 \rightarrow s_2 \square \ldots \square e_n \rightarrow s_n\ \}$ and $BB \equiv e_1 \vee e_2 \vee \ldots e_n$. What property does BB have to have for us to avoid an infinite loop when executing DO?

3. Consider the loop $i := 0;\ while\ \{\ i < 1000 \rightarrow s_1;\ i := i+1 \square i < 1000 \rightarrow s_2;\ i := i+1\ \}$ (where neither s_1 nor s_2 modifies i). Do we know anything about how many times or in what pattern we will execute s_1 vs s_2?

4. Consider the loop $x := 1;\ while\ \{\ x \geq 1 \rightarrow x := x+1 \square x \geq 2 \rightarrow x := x-2\ \}$. Can running it lead to an infinite loop?

5. What is $M(s, \{x = 1\})$ where s $\equiv$ $while\ \{\ x \leq 20 \rightarrow x := x*2 \square x \leq 20 \rightarrow x := x*3\ \}$?

Solution to Practice 7 (Nondeterministic Sequential Programs)

1. (Basic properties of nondeterministic if)
   a. We need $\sigma \vDash BB$, because if $\sigma \vDash \neg BB$, then $M(IF, \sigma) = \{\perp_e\}$. (In English: At least one guard must be true; if none of them are true, we get a runtime error.)
   b. The order of the guarded commands doesn't matter: If more than one guard is true, we nondeterministically choose one element from the set of corresponding statements, and in a set, the elements aren't ordered.
2. The nondeterministic *while* loop halts if *BB* is false at the top of the loop; an infinite loop occurs when *BB* is always true at the top of the loop.
3. Say S_1 is run m times and S_2 is run n times. We know $0 \le m, n \le 1000$ and $m+n = 1000$, but that's all. At each iteration, the choice is nondeterministic (i.e., unpredictable). The choice does not have to be random (like with a coin flip), and the sequence of choices don't have to follow an pattern or distribution or be fair, etc. We can't even assign a probability to any particular sequence of choices (like "always choose S_1").
4. It's possible that the loop could run forever. There's no guaranteed fairness in nondeterministic choice, so we could increment x by 1 many more times than we decrement it by 2.
5. $\{\{x = 24\}, \{x = 27\}, \{x = 32\}, \{x = 38\}, \{x = 48\}\}$

Notes - do not publish

#7 {{x = 24}, {x = 32}, {x = 36}} get to 24 via x = 12 * 2, x = 32 via x = 16*2, x = 36 via x = 18*2, x = 27 via x = 9*3

m	n	2^ m	3^n	2^m * 3^n	
0	0	1	1	1	
1	0	2	1	2	
2	0	4	1	4	
3	0	8	1	8	
4	0	16	1	16	2^4 * 3^0
5	0	32	1	32	2^5 * 3^0
0	1	1	3	3	
1	1	2	3	6	
2	1	4	3	12	2^2 * 3*1
3	1	8	3	24	2^3 * 3^1
0	2	1	9	9	
1	2	2	9	18	2^1 * 3^2
2	2	4	9	36	2^2 * 3^2
0	2	1	1	0	
0	3	1	1	0	

