CS350 – Nondeterminism

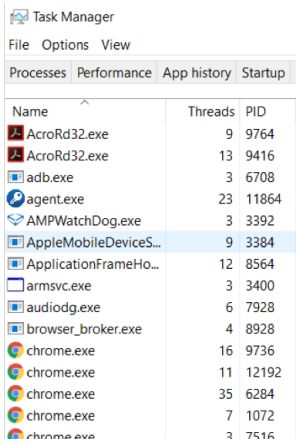
Purpose: Nondeterminism is a powerful tool that captures the notion of multitasking or performing multiple computations in parallel. One use of this tool is to try multiple possible actions at the same time and see which, if any, lead to fruitful results. The purpose of this module is for you to use nondeterminism to achieve tangible results.

Knowledge: This module will help you become familiar with the following content knowledge:

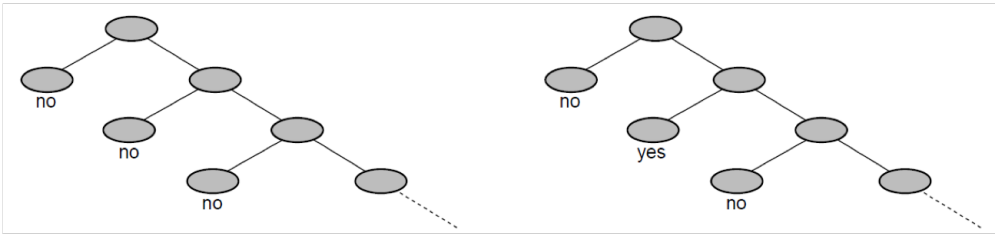
- How to use multiple threads to achieve nondeterminism in python.
- How to view nondeterminism as a computation tree.
- How to use nondeterminism with a Turing machine.

Activity: With your group perform the following tasks and answer the questions. You will be reporting your answers back to the class in 30 minutes.

1. Nondeterminism is present in most modern computers. Hundreds of processes and threads are running "simultaneously" on your laptop. (In fact, only a few are genuinely simultaneous. The operating system rapidly switches between threads to give the appearance of simultaneously working on dozens of tasks). What makes this overall behavior nondeterministic?



2. Computation trees are a useful way of modeling nondeterministic computations. Branches in the trees indicate when multiple threads are spawned. What is the output of each of the four trees? Remember that in nondeterministic processes we are seeing if some choice leads to success and/or a result.



3. Nondeterminism does not allow us to solve more problems. That is because any problem computed with a nondeterministic python program can also be computed by a deterministic one. Sketch out how we could get a deterministic program, given a nondeterministic one.

4. Nondeterminism, however, allows us to show that unrecognizable problems exist:

   Claim: The complement of a recognizable, undecidable decision problem is unrecognizable.

   Sketch of proof: Given an undecidable decision problem D, suppose that D and its complement are both recognizable. How can we use nondeterminism to get a contradiction?

Complete the following assignments for grading. Each should be done individually but you may consult with a classmate to discuss strategies.

Assignment 1:
Complete exercise 8.1 on p160 in the text. You can parse the input string by using index and substring operations. For example, the following code will extract the value of k:

```
i = inString.index(';')
k = int(inString[:i].strip())
inString = inString[i+1:]
```

Criteria for Success: The input "823;18910 5235 3422" produces no since none of the values are multiples of 823 and the input "823;18910 5235 3422 1646" produces yes.

Assignment 2:
Complete exercise 8.2 on p160 in the text.

Criteria for Success: The input "823;18910 5235 3422" produces no since none of the values are multiples of 823 and the input "823;18910 5235 3422 1646" produces 1646.

Assignment 3:
Complete exercise 8.4 on p161 in the text.

Criteria for Success: For each of the computation trees you have either given the output or stated that the result is undefined.

Assignment 4:
Complete exercise 8.9 on p162 in the text.

Criteria for Success: You have a completed a) through d). For d) you have a clear explanation of why you will hit the nondeterministic points a finite number of times for any input.

Assignment 5:
In JFLAP create a nondeterministic Turing Machine which accepts only strings of A's surrounding by x's on each side if the number of A's is either a multiple of 2 or a multiple of 3.

Criteria for Success: The strings xAAAAx and xAAAAAAAAAx would both be accepted since the first is a multiple of 2 and the second is a multiple of 3. But the string xAAAAAx would not be accepted.

Submit your files in Canvas for grading.