Homework 7

Michael Anderson

May 18, 2011

CS517

Prof. Cull

1

By Savitch's Theorem, if it takes $O(c^n)$ space to recognize a set for some constant positive value of c, Then it takes $O((c^n)^2)$ space to simulate non-determinism for that set. Let $\hat{c} = c^2$:

$$O((c^n)^2) = O((c^2)^n) = O(\hat{c}^n)$$

Since $\hat{c}$ is also a constant positive value, we are still in ESPACE, and $N - ESPACE = ESPACE$.

2

The possible configurations of some Turing Machine are no more than the Cartesian Product its state space, its possible tape contents, and the position of the read/write head. If a Turing Machine enters the same configuration twice while executing, then it is stuck in a dead loop since it must continue entering that configuration again and again. Therefore, the runtime is either infinite or bounded by $|Q| \cdot |\Gamma|^{S(n)} \cdot S(n)$. Since Q and Γ are both constants for any particular TM, we have that the runtime of any nondeterministic program is either infinite or exponential in the amount of space it uses.

3

By Problem 2 and Savitch's Theorem, we have the runtime as either infinite or $O(c^{S(n)^2})$ for some constant c.

4

a) Long division:

$$\begin{array}{r} 3 \\ 12\overline{)36} \\ \underline{36} \\ 0 \end{array}$$

b) $36/12 = (3 \cdot 3 \cdot 2 \cdot 2)/(3 \cdot 2 \cdot 2) = 3$

c) Suppose we code algorithms for the above methods, and give each method its own subroutine, and name them $DIV1$ and $DIV2$. Then given some function that requires division among natural numbers we can code the division as a nondeterministic instruction that either calls $DIV1$ or $DIV2$.

5

The arguments for all three parts (a), (b), and (c) are very similar. We have trivially that $N - RE \supseteq RE$, $N - coRE \supseteq coRE$, and $N - Recursive \supseteq Recursive$, because a deterministic acceptor, rejector, or recognizer is also a nondeterministic acceptor, rejector, or recognizer respectively.

Any nondeterministic program or part of a program that finishes in a finite amount of time can be simulated in a finite amount of time by a deterministic program. To make the conversion to a deterministic program, at each nondeterministic instruction simply use fork to create a thread for each possible path that the program could take, and execute each thread, thereby taking all of the possible paths.

Deterministic versions of nondeterministic programs are likely to have more, but still finite, runtimes. Since no particular runtime (except for finite when applicable) is specified for the classes RE, coRE, or Recursive, we then have that $N - RE \subseteq RE$, $N - coRE \subseteq coRE$, and $N - Recursive \subseteq Recursive$; and that $N - RE = RE$, $N - coRE = coRE$, and $N - Recursive = Recursive$.

6

(a) Each transition in a DM represents all and only all of the possible transitions that could have been taken non-deterministically in M. So if the DM ends up in an accepting state after reading a string, M could also end up in an accepting state after reading the same string, and will with magic guessing.

(b) Each possible state transition in M is encoded into DM by definition, and any set of M states that contain an accepting M state are consider an accepting state in the DM, so any path taken through M that leads to an accepting state must correspond to a path through DM that leads to an accepting state.

(c) Any time DM rejects a string, M must also reject it because any possible path to an accepting state in M would be represented within DM. Any time M rejects a string, DM must reject it also, because DM cannot take paths to accepting states that do not exist in M. Since M and DM must both accept and reject the same sets, they recognize the same set.

7

DM	0	1
q_0	$q_{1,2}$	q_1
q_1	q_1	q_3
$q_{1,2}$	$q_{1,2}$	q_3
q_3	q_3	q_3

This DM has only four states and it accepts strings of length ≥ 2, if a '1' symbol appears somewhere after the first digit of the string.

8

Such a nondeterministic machine would simply stay in the start state until the kth to the end position is reached, and then if a '1' symbol was found in that position transition into the first state of a chain of k states, the last of which is the only accepting state. After reaching the end of this chain, any additional symbols read in would lead to a trap state, so that the accepting state would only be reached if exactly k-1 symbols were read in after the '1' symbol in the kth to the last position. All of this requires $k + 2 = O(k)$ states, an initial state, the "chain" of k states, and the trap state.

(a) Let q_a be the only accepting state in M, and q_t be the trap state:

M	Not 1	1
q_0	q_0	q_0, q_1
q_1	q_a	q_a
q_a	q_t	q_t
q_t	q_t	q_t

DM	Not 1	1
q_0	q_0	$q_{0,1}$
$q_{0,1}$	$q_{0,a}$	$q_{0,1,a}$
$q_{0,a}$	$q_{0,t}$	$q_{0,1,t}$
$q_{0,t}$	$q_{0,t}$	$q_{0,1,t}$
$q_{0,1,a}$	$q_{0,a,t}$	$q_{0,1,a,t}$
$q_{0,1,t}$	$q_{0,a,t}$	$q_{0,1,a,t}$
$q_{0,a,t}$	$q_{0,t}$	$q_{0,1,t}$
$q_{0,1,a,t}$	$q_{0,a,t}$	$q_{0,1,a,t}$

Here there are 7 states in the DM, and only 4 states in M, not too significant of a difference.

(b) Use the same defs for q_a and q_t.

M	Not 1	1
q_0	q_0	q_0, q_1
q_1	q_2	q_2
q_2	q_a	q_a
q_a	q_t	q_t
q_t	q_t	q_t

M	Not 1	1
q_0	q_0	$q_{0,1}$
$q_{0,1}$	$q_{0,2}$	$q_{0,1,2}$
$q_{0,2}$	$q_{0,a}$	$q_{0,1,a}$
$q_{0,a}$	$q_{0,t}$	$q_{0,1,t}$
$q_{0,t}$	$q_{0,t}$	$q_{0,1,t}$
$q_{0,1,2}$	$q_{0,2,a}$	$q_{0,1,2,a}$
$q_{0,1,a}$	$q_{0,2,t}$	$q_{0,1,2,t}$
$q_{0,1,t}$	$q_{0,2,t}$	$q_{0,1,2,t}$
$q_{0,2,a}$	$q_{0,a,t}$	$q_{0,1,a,t}$
$q_{0,2,t}$	$q_{0,a,t}$	$q_{0,1,a,t}$
$q_{0,a,t}$	$q_{0,t}$	$q_{0,1,t}$
$q_{0,1,2,a}$	$q_{0,2,a,t}$	$q_{0,1,2,a,t}$
$q_{0,1,2,t}$	$q_{0,2,a,t}$	$q_{0,1,2,a,t}$
$q_{0,1,a,t}$	$q_{0,2,t}$	$q_{0,1,2,t}$
$q_{0,2,a,t}$	$q_{0,a,t}$	$q_{0,1,a,t}$
$q_{0,1,2,a,t}$	$q_{0,2,a,t}$	$q_{0,1,2,a,t}$

Here there are 5 states in M, and 16 states in DM. The number of states in M is growing linearly, while the number of states in DM is growing exponentially. This makes sense, because any DM for this type of problem must remember which of the last k digits read were 1's, so that at any time when the string ends the DM will know whether the kth to last digit was a 1. It takes 2^k many states to represent all of those possible combinations.