COMP 362 Assignment 4

Christopher Hundt 110220945

March 30, 2004

1. (a) Suppose $Ax \leq b$ for some $x \geq 0$. Then each component of Ax is less than or equal to the corresponding component of b. Then $(Ax)^T \leq b^T$, since the corresponding components are the same. That is, $x^T A^T \leq b^T$. Then, since $y \geq 0$, $x^T A^T y \leq b^T y$. But $A^T y \geq 0$ and $x \geq 0$, so $x^T \geq 0$, so $x^T A^T y \geq 0$. Then $b^T y \geq 0$, a contradiction. Thus our assumption that $Ax \leq b$ was wrong, and (LP1) is infeasible.

   (b) Suppose $Az \leq 0$ and $c^T z = \alpha > 0$. Then, for some feasible solution x and any supposed maximum M, let $k > \frac{M - c^T x}{\alpha}$. Then $A(x + kz) = Ax + kAz \leq Ax \leq b$ and $c^T(x + kz) = c^T x + kc^T z > M$. Thus no maximum could exist, and (LP1) is unbounded.

2. The dual of (LP1) is

$$\min b^T y \text{ subject to } A^T y \geq c,\ y \geq 0.$$

This is equivalent to

$$\max(-b^T)y \text{ subject to } (-A^T)y \leq -c,\ y \geq 0.$$

Then the dual of this is

$$\min(-c)^T z \text{ subject to } (-A^T)^T z = -Az \geq -b,\ z \geq 0,$$

which is equivalent to

$$\max c^T x \text{ subject to } Az \leq b,\ z \geq 0,$$

which, after replacing z by x, is (LP1).

3. Let the non-deterministic Turing machine be N. We will design a deterministic Turing machine M that decides $L(N)$. We assume that $f(|x|)$ can be computed easily by M and that M has access to N. We know that N's transition relation provides some number of possible transitions from each state for a given input. We let k be the maximum number of transitions possible from a single state for a single symbol. Then there are no more than $k^{f(|x|)}$ different computation paths of length no more than $f(x)$. So we establish an ordering of the possible computation paths for input x by doing a depth-first search and ordering paths in the order in which they are finished in the DFS. Then M uses the following algorithm:

for each path of computation on x
    do run N along computation path x until a halting state (for at most $f(|x|)$ steps)
        if in state TRUE
            then return TRUE
        "Back up" to last decision made where there was another untried choice
return FALSE

When we say "back up," we mean restore the tape to the point where the last decision was made between different possible transitions. This can be done if we store a list of the decisions made on the current path and a second list of the symbols that are written over, so they can be replaced. Then, for example, if we wrote a symbol u where another symbol v used to be and then moved to the right, we move to the left and replace u with v.

Regarding the time for this algorithm, each path is traversed at most once and takes time no more than $f(|x|)$, and there are no more than $k^{f(|x|)}$ paths. Thus a very rough upper bound for the time is $f(n)k^{(n)}$.

4. By definition of NTIME, Any machine in NTIME($f(n)$) can be considered as the machine N in the solution above. In the machine M described above, there is no more than $f(|x|)$ tape used in following a path in N, since each path has no more than $f(|x|)$ steps, so can only write to $f(|x|)$ tape cells, and the "backing up" process makes sure the same space on the tape is re-used. The only extra space required to store information is to store the current set of decisions made (of which there are no more than $f(|x|)$ since that is the limit of the path length) and the symbols overwritten for the current path (of which there are no more than $f(|x|)$ since there are only $f(|x|)$ symbols used in running a single path). Thus the total tape size for a deterministic machine M simulating the non-deterministic machine N is no more than $3f(|x|)$, so $N \in$ SPACE($f(n)$).