NTIME vs DTIME

Suppose $M$ is nondeterministic, $T(n)$ time bounded and accepts $L$. (Recall that every path of $M$ (even non-accepting ones) must be $T(n)$ time bounded).

Number of different IDs of $M$ (reachable within $T(n)$ steps from starting ID)
$\leq s \times (1 + T(n))^k \times r^k T(n)$,
where $s$ is the number of states of $M$,
$k$ is the number of tapes and
$r$ is the number of symbols used by $M$.

$s \times (1 + T(n))^k \times r^k T(n) \leq d T(n)$ for some constant $d$. 
$M'$ constructs a list of reachable IDs in a BFS manner starting from initial ID of $M$.

This list can be constructed in time polynomial in number of IDs and max length of IDs.

$M'$ can then search the list to see whether it contains an accepting ID.

Thus, total time required is bounded by $c^{T(n)}$ for some constant $c$.

Note that $c$ depends on $M$ (and thus $L$).
NSPACE vs DSPACE: Savitch’s Theorem

Trivial simulation (as for time) would give exponential bounds.

**Theorem (Savitch):** Suppose $S(n)$ is fully space constructible and $S(n) \geq \log n$. Then $NSPACE(S(n)) \subseteq \text{DSPACE}([S(n)]^2)$.

Proof: Suppose $S(n)$ is fully space constructible and $M$ is a $S(n)$ space bounded machine which accepts $L$. Wlog assume that $M$ has only one work tape. Alphabet size of $M$: $r$
Number of states of $M$: $s$

Number of different IDs of $M$ on inputs of length $n$:
$\leq s(n + 2)(S(n))(r)^{S(n)} \leq c^{S(n)}$, for some constant $c$.
Thus if $M$ accepts $x$, then it must do so within $c^{S(n)}$ steps.
(Note that $S(n) \geq \log n$; this is why we needed $S(n) \geq \log n$).
$I_1 \Rightarrow_i I_2$: denotes the fact that $M$ can reach from ID $I_1$ to ID $I_2$ in atmost $i$ steps.

We construct $M'$ as follows:

$M'(x)$

Let $n = |x|$. Let $I_0$ be the initial ID of $M$ on input $x$.

If there exists an accepting ID $I_f$ of $M$, of length atmost $S(n)$, such that $\text{TEST}(I_0, I_f, c^{S(n)})$ is true, then accept.

Else reject.

End $M'$
TEST($I_1, I_2, t$)

If $I_1 = I_2$, then return true.

ElseIf $t < 1$, then return false.

ElseIf $t \geq 1$ and one can reach $I_2$ from $I_1$ in one step, then return true.

ElseIf there exists an ID, $I'$, of length at most $S(n)$, such that 

$\text{TEST}(I_1, I', \lfloor t/2 \rfloor)$ and $\text{TEST}(I', I_2, \lceil t/2 \rceil)$,

then return true.

Else return false.

End TEST
Clearly, $M'$ accepts $x$ iff $M$ does.

Space needed:
Each TEST routine needs about $O(S(n))$ local space.
The depth of recursive calls to TEST is atmost $O(S(n))$.
Thus the space used is atmost $O([S(n)]^2)$.
The implementation of the above recursive routine TEST on a TM can be done by separating the different recursive calls by using special markers and doing a stacklike implementation.
Suppose $X$ is a class of languages.
Then
\[ coX = \{ \overline{L} : L \in X \}. \]

Nondeterminism:
Guess (a proof, certificate ....) and Verify the correctness.
Closure of NSPACE under complementation:
Immerman-Szelepscenyi Result

\[ \text{DST} \text{Conn} = \{(G, s, t) : \text{there is a path from } s \text{ to } t \text{ in } G\}. \]

\( G \) is a directed graph.

Proposition: \( \text{DST} \text{Conn} \in N \text{LogSpace} \).
Proof: Suppose \( n \) is the number of nodes in the graph \( G \).
Starting with \( s_0 = s \).

At stage \( r \):

- If \( s_r = t \), then accept.
- ElseIf \( r > n \), then abort.
- Else: Guess \( s_{r+1} \), verify that there is an edge \((s_r, s_{r+1})\). If fail, then abort. Otherwise, continue to next stage.
**Theorem:** \( DSTConn \in coNLogSpace. \)

Proof: \( count(i) \): gives the number of nodes in \( G \) which can be reached from \( s \) in at most \( i \) steps.

\[ \text{CannotReach}(s,t) \]

\[ c = 0 \]

For each \( v \in V(G) \) do:

Guess and verify that \( v \) is reachable from \( s \) using path of length at most \( n \).

if successful, then

let \( c = c + 1 \); if \( v = t \), then reject.

End For

If \( c = count(n) \), then accept; Else reject.

End

If \( t \) is not reachable, then the above algorithm can (non-deterministically) find \( count(n) \) other nodes which are reachable from \( s \), and accept.
We now show how to compute $\text{count}(\cdot)$; Note that $\text{count}(0) = 1$. We show how to compute $\text{count}(i + 1)$ using $\text{count}(i)$.

$\text{count}(i + 1)$

$c = 0$.

For each $v \in V(G)$ do

$d = 0$

For each $w \in V(G)$ do

Guess and verify a path from $s$ to $w$ of length at most $i$.

If successful in above, then

let $d = d + 1$;

if $w = v$ or $(w, v)$ is an edge, then let $c = c + 1$, and continue with next $v$.

End For

If $d \neq \text{count}(i)$, then reject the computation.

End For

$\text{count}(i + 1) = c$
Theorem: $\text{NLogSpace} = \text{coNLogSpace}$.
Theorem: Suppose $S(n)$ is fully space constructible, and $S(n) \geq \log(n)$. Then $\text{NSpace}(S(n)) = \text{coNSpace}(S(n))$. 