NTM for composite numbers

$$L = \{1^n : n \text{ is a composite positive integer}\}$$

composite: product of two integers $\neq 1$.

e.g., $111111 \in L$

$11111 \notin L$

Can design a (deterministic) TM to multiply two positive integers in unary:

Input: $x\#y\#$, where $x = 1^k$ and $y = 1^m$, $k, m \in \mathbb{Z}^+$

Output: $x\#y\#z$, where $z = 1^{km}$

NTM for composite numbers

1. Write #11 after input x and enter "guess 1st factor" stage.
2. While in "guess 1st factor" stage, make a nondeterministic choice:
   - Write 1 and move R, remaining in this stage, or
   - Write #11, move R, and enter "guess 2nd factor" stage.
3. While in "guess 2nd factor" stage, make a nondeterministic choice:
   - Write 1 and move R, remaining in this stage, or
   - Write #, move R, and enter "multiply" stage.
4. While in "multiply" stage, multiply the two guessed factors in unary, writing the result after the last #; enter "verify" stage.
5. While in "verify" stage, compare the input x (the string of 1s before the first #) and the result of the multiplication in Step 4 (the string of 1s after the last #). If they are equal, accept; otherwise, reject.

Nondeterminism is useful because it allows the TM to "guess" something:

- Guess when to take one kind of action as opposed to another (e.g., prove closure of decidable languages under concatenation --- exercise!).
- Guess the lucky solution that is the key to answering a decision problem (as in testing if a number is composite).

"Guessing" really amounts to making nondeterministic moves.

For now, you must show how the NTM makes its guess (as in the preceding example).

Once you get more comfortable with the notion of nondeterminism we will allow looser uses of the phrase "guess..." (e.g., guess the factors of the input and verify that their product is equal to the input).

Simulation of NTM by DTM

Given a NTM M construct a DTM M' that simulates M:

M' accepts/rejects/loops on $x \Longrightarrow M$ accepts/rejects/loops on x

M' has two tapes

- Tape 1 contains a queue of configurations that have been discovered in the BFS of the computation tree of M on x.

$$\#C_0\#C_1\#C_2 \ldots *C_i \ldots \#C_j \ldots$$

  - Before *: "explored" configurations;
  - C_i: configuration currently being (or next to be) explored;
  - after C_i: discovered configurations yet to be explored.

- Tape 2 is a "work tape".

Simulation of NTM by DTM

Initially the input x is on tape 1; tape 2 is empty

1. Shift x on tape 1 two positions to the right and write $*q_0$ in the first two cells. Place the head of tape 1 over *. (Tape 1 = $* q_0 x$.)
2. Copy to tape 2 the configuration C that follows * on tape 1 (where the head of tape 1 is positioned) as many times as there are next moves of the NTM from C, each preceded by #.
3. For each configuration on tape 2, make appropriate changes to obtain the configuration after the corresponding step of the NTM; append the resulting configuration to tape 1, preceded by #.
   If copying an accepting configuration, accept. Erase tape 2 and move its head to the leftmost cell.
4. Move left the tape 1 head until it finds the *; replace * by #; move right the tape 1 head until the next #; if there is none, reject; else change the # to * and go to step 2.