1 Honors Theory: Hints for HW1

All machines here are deterministic.

1.1 Using a PM to simulate a TM

This is relatively easy. Let M be any k-tape TM. We construct a PM P to simulate M.

Let

$$\Delta = \{c, q, \widehat{q}, t_0, t_1, \ldots, t_k, h_0, \ldots, h_k, p\}$$

These symbols have the following interpretation:

c	Cell Contents
q	State
$\widehat{q}$	Current State
t_i	ith tape
h_i	ith head position
p	previous cell

Before describing how the symbols are used, we note some useful conventions/notations:

- In figures, when only draw those σ-edges ($\sigma \in \Delta$) which are not self-loops.
- If $w \in \Delta^*$, then let $[w]$ denote the node pointed to by w in the current Δ-structure.
- If $\sigma \in \Delta$, then σ^+ is the infinite set of words $\{\sigma, \sigma^2, \sigma^3, \ldots\}$. Combined with the previous notation, then $[\sigma^+]$ is the set of nodes $\{[\sigma], [\sigma^2], [\sigma^3], \ldots\}$. This set is always finite in our applications, and appears as a list that is terminated when $[\sigma^m] = [\sigma^{m+1}]$ for some m). See figure 1.

Thus we can use $[\sigma^+]$ to construct a finite list sequence of nodes:

1. If M has n states, then these are represented be a list of n nodes in the set $[q^+]$. The initial state q_0 and accepts state q_a are represented by $[\epsilon]$ (the center) and $[q]$, respectively.
2. If M has m tape symbols, then these are represented by a list of m in the set $[c^+]$.
3. The set of nodes $[t_i^+]$ represents contiguous cells on the ith tape. With some exceptions noted below, all the represented cells contain non-blank symbols.

   To facilitate moving to the next cell on the left, we use the p-symbol (the "previous" node): Thus $[t_i^j] = [t_i^{j+1} p]$ for all j such that $[t_i^j] \neq [t_i^{j+1}]$.

The configuration of M can now be represented:

- We represent the contents of the ith tape as follows: if the mth cell contains the jth tape symbol, then $[t_i^m c] = [c^j]$.

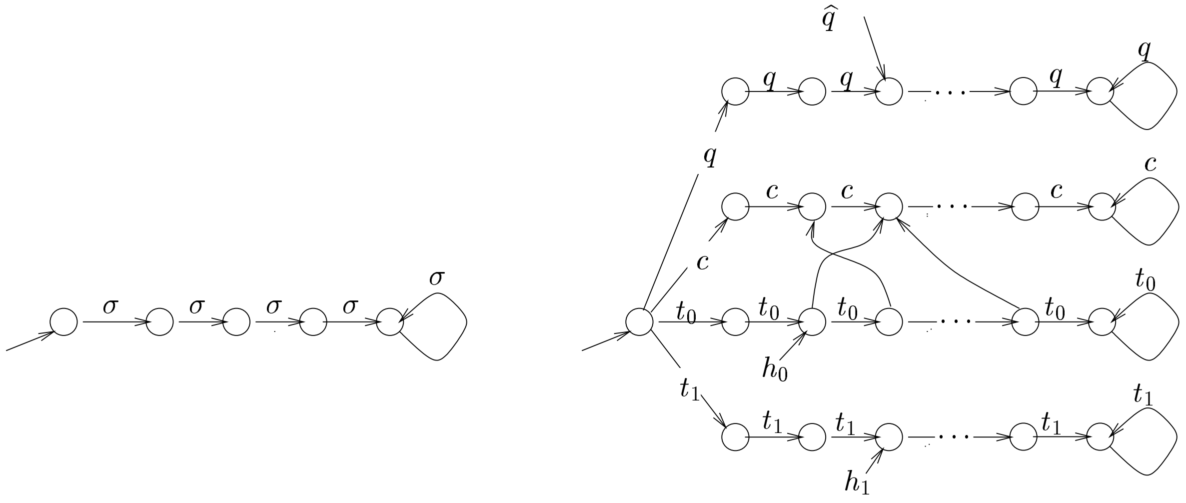

Figure 1: (a) σ-List. (b) Δ-structure to represent a 1-tape TM configuration.

If the mth cell has a blank, then $[t_i^m c] = [\epsilon]$. Actually, we only need to represent blanks at the start of the computation, when all work cells are blanks. Also, the first and last cells of the input tape are blanks. In all other cases, we may assume non-blank symbols. Note: the index m of a cell on a non-blank work tape is relative to the first non-blank symbol (which has the index $m = 1$) on that tape.

- The position of the ith head is given by $[h_i]$. Thus $[h_i] = [t_i^m]$ for some $m \geq 1$.
- The current state is $[\widehat{q}]$. Thus $[\widehat{q}] = [q^i]$ for some $i \geq 0$.

Initialization: before the computation of P on any input x, it is assume that the Δ structure is properly initialized:

1. The states $[q^+]$ and tape symbols $[c^+]$ are correctly setup.
2. Initial state is start state: $[\widehat{q}] = [\epsilon]$.
3. The input tape contents is initialized to contain x.
4. The work tapes are initially empty: for $i = 1, \ldots, k$, $[t_i^2] = [t_i]$ (only one cell represented) and $[t_i c] = [\epsilon]$ (blank symbol).

A typical configuration is shown in figure 1(b). We leave the rest of the details to the student (in your homework, however, we expect you to spell out a bit more details).

The main result is this:

Theorem 1 *For any Turing machine* M *that accepts in time $t(n)$ and space $s(n)$, there is a pointer machine P that accepts $L(\mathrm{M})$ in time and space $O(t(n), s(n))$.*

1.2 Using a TM to simulate a PM

This is slightly harder.

Let P be any PM and M be a TM we will construct to simulate P.

We assume the input to P follows the convention in part (1) above. In particular, we initially have the input symbols $[c^+]$ and the input word is placed on tape 0, $[t_0^+]$.

We also assume that P never modify the initial input (to conform to our conventions for TMs). Thus, assignments of the form

$$t_0^m \leftarrow w, \quad t_0^m c \leftarrow w, \quad c^m \leftarrow w$$

is illegal for all $m \in \mathbb{N}$.

Let $d = |\Delta|$. The key issue is some efficient representation of an arbitrary Δ-structure used by P. The machine M has $d+2$ work tapes with the following functions.

Tape $d+1$: this stores in binary notation the value n equal to current number of nodes in the current Δ-structure. The length of n is $\lfloor \log_2 n \rfloor + 1$ is denoted by s.

NOTE: it is assume that the last bit in the binary notation has some additional mark to indicate that it is the last bit. E.g., if the last bit is 0, we use the symbol $0'$, and it is it 1, we use $1'$. This self-limiting notation is assumed in all our binary notations. Thus we can string together consecutive binary numbers without having to explicitly introduce markers.

Tape d+2: this is used as scratch tape.

Tapes 1 to tape d: these store a represent of the Δ-structure as follows. Assume each node is indexed from 1 to n. If u is the ith node represented on tape 2, let $\#(u)$ be the binary notation for i. All our binary numbers will use exactly s bits, zero padded if necessary.

On each of these d tapes stores a sequence of n binary numbers. Thus each tape uses ds cells.

Let u be a ith node. For $j = 1, \ldots, d$, if the j-edge from u goes into node u_i, then the ith binary number of tape j will contain the number $\#(u_i)$.

Our machine M has various states, but certain states are called DISTINGUISHED. There is a 1-1 correspondence between the distinguished states and the instructions of P. Whenever we enter a DISTINGUISHED state, the contents of tapes 1,2 and 3 will conform to the specifications above. In the non-distinguished states, the tape contents are in transition and may not conform to the specifications.

Each time a new node is created in the Δ-structure, we need to increment the value n on tape 1. If this causes the value s to increase, then we must first update the contents of tapes 2 and 3 in the natural way before proceeding.

One more detail: when n reaches a power of 2, then ALL the binary numbers representation must changed. This involves updating all the tape contents of tapes 1 to k.

We leave the remaining details to the reader (again, your homework answers should elaborate a bit more). The main result is this:

Theorem 2 *For any pointer machine* P *that accepts in time-space* $(t(n), s(n))$*, there is a Turing machine that accepts* $L(\mathrm{P})$ *in time and space* $O(ts\log s, s\log s)$.

The space of $s\log s$ is immediate.

This space $s\log s$ is actually optimal in the worst case: note that if a Δ-structure has n nodes, then there are $n^{n|\Delta|}$ possible configurations. We need at $\Omega(n\log n)$ tape cells to represent these configurations.

Why is the time $O(ts\log s)$? This is because simulating each PM instruction may take $O(s\log s)$ to sweep through the tapes to search for a particular node number.

QUESTION: Could the time complexity be improved?

1.3 Comments

Most students did the first part quite well, but some important details in the second part were sometimes missed. For example:

– We must use the PM to accept languages, so there must be some convention for inputing a word x. So, at the start of the PM's computation, this representation of x is already part of the Δ-structure!

– If you are a TM simulating a PM, you do not have to represent the instructions of the PM on your tapes – you can "remember" this finite information in your states!

– Some people did not carry out the time and/or space complexity analysis. A common mistake is to believe that each PM step can be simulated in $O(\log s)$ steps of a TM.