Simulating Two Pushdown Stores by One Tape in $O(n^{1.5}\sqrt{\log n})$ Time.
(Preliminary Version)

Ming Li*

TR 85-667
March 1985

Department of Computer Science
Cornell University
Ithaca, New York 14853

* This work was supported in part by an NSF grant DCR-8301766.

Simulating Two Pushdown Stores by One Tape in $O(n^{1.5}\sqrt{logn})$ Time

(Preliminary Version)

Ming Li*
Department of Computer Science
Cornell University
Ithaca, New York 14853

Abstract

Based on two graph separator theorems, one old (the Lipton-Tarjan planar separator theorem) and one new, we present two unexpected upper bounds and resolve several open problems for on-line computations:

(1) 1 tape nondeterministic machines can simulate 2 pushdown stores in time $O(n^{1.5}\sqrt{logn})$ (true for both on-line and off-line machines). Together with the $\Omega(n^{1.5}/logn)$ lower bound by the author [L1], this solved the open problem 1 in [DGPR] for the 1 tape vs. 2 pushdown case. It also disproves the commonly conjectured $\Omega(n^2)$ lower bound. (Note, the $\Omega(n^2)$ lower bound has been proved for the deterministic case [M, L, V].)

(2) The languages define by [M] and [F], aimed to obtain optimal lower bound for 1 tape nondeterministic machines, can be accepted in $O(n^2 loglogn/\sqrt{logn})$ and $O(n^{1.5}\sqrt{logn})$ time by a 1 tape TM, respectively.

(3) 3 pushdown stores are better than 2 pushdown stores. This answers open problem 3 of [DG]. (An $\Omega(n^{4/3}/log^c n)$ lower bound is also obtained.)

(4) 1 tape can nondeterministically simulate 1 queue in $O(n^{1.5}\sqrt{logn})$ time. (The lower bounds were: $\Omega(n^2)$ for deterministic case [V] and $\Omega(n^{4/3}/logn)$ [L1] for nondeterministic case.)

* This work was supported in part by an NSF grant DCR-8301766.

1. An Overview

The theory of computational complexity has been developed based on various models of Turing machines. It is essential to understand the precise relationship among those computing models, *e.g.*, with or without nondeterminism and/or some more tapes (or pushdown stores). In this paper we develop a new simulation technique to resolve some remaining major open problems for on-line computations, basically completing our knowledge on the effects of the different number of tapes/pushdowns for on-line nondeterministic machines. Moreover, our result extends to off-line computations.

There are 2 major models of Turing machines for time complexity: on-line and off-line. A k tape TM has k work tapes and an extra input tape. If the input tape is 1-way, the machine is *on-line*; otherwise it is *off-line*.

Over the past 2 decades, the question of the relative power of TM's with different number of tapes has been extensively studied by many researchers.

Upper Bounds (for both on-line and off-line):

(1) Hartmanis-Stearns [HS]: 1 (det. or nondet.) tape can simulate k tapes/pushdowns (of same type) in $O(t^2)$ time;

(2) Hennie-Stearns [HS1]: 2 deterministic tape can simulate k tapes/pushdowns in $O(t\log t)$ time;

(3) Book-Greibach-Wegbreit [BGW]: 2 (3) nondeterministic tape (pushdowns) can simulate k tapes/pushdowns in same amount of time.

It is widely believed that above simulations are optimal. To obtain matching lower bounds, serious effort have been made, and following lower bounds (ONLY for on-line!) were obtained:

(1) Rabin [R]: 2 deterministic tapes are better than 1;

(2) Aanderaa [A] and [PSS]: k deterministic tapes are better than $k-1$, generalized by [PSS];

(3) Duris-Galil [DG]: 2 nondeterministic tapes (pushdowns) are better than 1;

(4) Paul [P]: simulating k deterministic tapes by $k-1$ requires $\Omega(n \log^{1/k} n)$ time;

(5) Duris-Galil-Paul-Reischuk [DGPR]: simulating 2 pushdowns by one nondeterministic tape requires $\Omega(nlogn)$ time; simulating k tapes by k pushdown stores requires $\Omega(nlog^{1/k+1}n))$ time;

(6) Vitanyi [V1]: simulating 2 tapes by 1 oblivious tape requires $\Omega(n^{3/2})$ time;

(7) Maass, Li, Vitanyi [M,L,V]: simulating 2 pushdowns by 1 tape requires $\Omega(n^2)$ time in deterministic case;

(8) Maass [M]: simulating 2 tapes (not pushdowns!) by 1 tape requires $\Omega(n^2/log^2nloglogn)$ time for nondeterministic machines; (A minor improvement by the current author [L]: $\Omega(n^2/lognloglogn)$;)

(9) Li [L]: nondeterministically simulating 2 pushdowns by 1 tape requires $\Omega(n^{1.5}/logn)$ time;

(10) Vitanyi [V]: deterministically simulating 1 queue by 1 tape requires $\Omega(n^2)$ time in the strong on-line model with output each step.

For nondeterministic on-line computations, the remaining open questions are:

(1) (Open problem in [L], originally in [DGPR]) Close the gap of $O(n^2)$ and $\Omega(n^{1.5}/logn)$ for 1 tape simulating 2 pushdowns. This is the only non-polylog gap left open. It is generally believed that an $\Omega(n^2)$ should hold, but despite some effort, the author was only able to obtain the $\Omega(n^{1.5}/logn)$ lower bound, although the $\Omega(n^2)$ lower bound is implied by a claim (without proof) of [F] and it is also proved by [V] in a modified nondeterministic model.

(2) (Open question 3 listed in [DG]) Are 3 pushdown stores better than 2 pushdown stores? In [DG], Duris and Galil listed three open questions for on-line computations. They solved the first two, and left this one open.

(3) Similar question as (1) for 1 tape vs. 1 queue. (Lower bound: $\Omega(n^{4/3}/logn)$ [L1]; upper bound $O(n^2)$.)

(4) Can we obtain $\Omega(n^2)$ for 1 tape machines by the language of [M] or language of [F] (as he claimed)?

(5) Does $\Omega(n^2)$ in (4) exist at all?

In this paper we answer all these questions except the last. Surprisingly, the first problem is not solved by the commonly conjectured n^2 lower bound. Instead, we present a new powerful simulation that allows 1 nondeterministic tape simulate 2 pushdowns fast for both on-line and off-line machines. Then we apply (1) to solve (2) and (3) and other related problems. We will also argue that 1 nondeterministic tape may be able to simulate 2 tapes faster than $O(t^2)$ time by studying a class of special graphs (To be added in the final paper). At least, the ingenious languages used by [M] and [F] (whose Theorem 2, if true, would imply the $\Omega(n^2)$ lower bound) will not yield $\Omega(n^2)$ lower bound.

2. The Graph Separators

The results in this paper rely on graph separators. In this section we present 2 separator theorems that we need, one old and one new.

The first theorem, which is very well-known as the planar graph separator theorem, is due to Lipton and Tarjan [LT]. The planar graph separator theorem have had many very nice applications [LT, LT1, G]. Theorem 1 in the next section can be considered as another application of it.

Theorem A (Planar Graph Separator [LT]): If graph G is planar, then it has a $\sqrt{n}$-separator.

The second theorem is a separator theorem about a special class of graphs. The combinatorial version of these graphs was defined by Maass [M]. We call undirected graph $G=<V,E>$ a *doubling transformation graph* if it is isomorphic to undirected $\overline{G}=<\overline{V}, \overline{E}>$, where $\overline{V}=\{0,1, \cdots, |V|-1\}$, and $\overline{E}=\{(i,j) \mid i=2j \ mod(|V|),\ i=2j+1 \ mod(|V|),$ or $i=j+1\}$. These graphs are important since should they not have a nontrivial separator, we can obtain a tight $\Omega(n^2)$ lower bound on 1 tape simulating 2 tapes for on-line machines. The Theorem 3.1 in [M] implies that these graphs do not have separators less than $n/logn$ and by which Maass obtained an $\Omega(n^2/log^2nloglogn)$ lower bound for 1 tape simulating 2 tape. In next theorem we show that these graphs do have nontrivial separators, and by which we show in Section 4 that the languages used in [M] and [F] can not yield $\Omega(n^2)$ lower bound.

Theorem B (Doubling Transformation Graph Separator): If G is a doubling transformation graph, then it has an $\frac{n}{\sqrt{logn}}$-separator.

Proof of Theorem B: Let $G=<V,E>$ where $V=\{0,1, \cdots ,k-1\}$ where $k=2^l$ for for some integer l and E as defined above. Therefore we have for s_1 and s_2 in V, $(s_1, s_2) \in E$ if

(1) $s_1=2*s_2(modk)$ or $s_1=2*s_2+1(modk)$, or

(2) $s_1=s_2+1$, or

(3) $(s_2, s_1) \in E$.

We show that there exists a partition of S into two sets S_1 and S_2 such that,

(a) $|S_1|=|S_2|$.

(b) $S_1 \cap S_2=\phi$,

(c) The separator N satisfies: $|N|=O(k/\sqrt{logk})$, where $N=\{s_1 \in S_1 | (s_1, s_2) \in E$ for some $s_2 \in S_2 \}$. (N is the set of elements in S_1 that are 'neighbors' of some elements in S_2.)

Let $bin(i)$ denote the binary representation of integer i, and $\#bin(i)$ denote the number of 1's in $bin(i)$. Without loss of generality, let l be odd. Define

$S_1=\{s \,|\#bin(s)<l/2 \}$, and

$S_2=\{s \,|\#bin(s)>l/2 \}$.

(Note: If l even, put half of those s's with $\#bin(s)=l/2$ into S_1 and the other half to S_2. This will not hurt the calculations below.) Obviously, $|S_1|=|S_2|$ and $S_1 \cap S_2=\phi$. We now calculate $|N|$ in the following. Define

$N_1=\{s_1 \in S_1 |$ for some $s_2 \in S_2$, s_1 and s_2 satisfy condition (1) in the lemma$\}$,

$N_2=\{s_1 \in S_1 |$ for some $s_2 \in S_2$, $s_1+1=s_2\}$, and

$N_3=\{s_1 \in S_1 |$ for some $s_2 \in S_2$, $s_2+1=s_1\}$.

It is clear that $|N| \leq 2|N_1|+|N_2|+|N_3|$. Since rule (1) in the lemma can at most change 1 in $\#bin(i)$ for any i, N_1 contains only those s_1's in S_1 such that $\#bin(s_1)=l/2$. Therefore

$$|N_1| \leq \binom{l/2}{l}=O\left(\frac{\sqrt{2\pi l}\, e^{-l} l^l}{\pi l e^{-l} (l/2)^l}\right)=O(2^l/\sqrt{l}).$$

Similarly, since $\#bin(s+1) \leq \#bin(s)+1$, $|N_2| = O(2^l/\sqrt{l})$.

Consider N_3. For $s_2 \in S_2$, if $\#bin(s_2) = \frac{l}{2} + i$ and $s_2 + 1 \in S_1$, then the last $i+1$ bits of bin(s_2) must be 1's. Fix i, we have at most $\sum_{i=1}^{l/2} \binom{(l/2)-1}{l-i-1}$ such elements. So,

$$|N_3| \leq \sum_{i=1}^{l/2} \binom{(l/2)-1}{l-i-1},$$

$$= \sum_{i=1}^{l/2} \binom{(l/2)-i}{l-i-1}$$

$$\leq \sum_{i=1}^{i/2} \binom{(l/2)-\lceil i/2 \rceil}{l-i+1}$$

$$= O\left(\sum_{i=1}^{l/2} \frac{2^{l-i+1}}{\sqrt{l}}\right)$$

$$= O(2^l/\sqrt{l}).$$

This concludes $|N| = O(k/\sqrt{logk})$. □ (Theorem B)

Remark: Theorem B gives an upper bound on Theorem 3.1 of [M].

Remark: Assertion (c) in the proof of Theorem B is true for any partition as long as for all $s_1 \in S_1$ and $s_2 \in S_2$ we have $\#bin(s_1) \leq \#bin(s_2)$. It is this property, which will be simply referred as Theorem B, which is actually used in the proof of Theorem 5.

3. The Fast Simulation

For the sake of clarity, we first give an $O(n^{1.5}\log^2 n)$ time simulation, then improve it to $O(n^{1.5}\sqrt{logn})$.

Theorem 1: An on-line nondeterministic two pushdown store machine can be simulated by an on-line one tape nondeterministic machine in time $O(n^{1.5}\log^2 n)$.

Proof of Theorem 1: Fix nondeterministic machine M with 2 pushdown stores, and name the 2 pushdown stores $S1$ and $S2$, 2 work heads h_1 and h_2. We will design a machine M' with only one work tape to simulate M in $O(n^{1.5}\log^2 n)$ time.

Lemma 1: M can be changed so that each of its work head either stay stationary without reading or seeing anything or it *pops* (then 'see') or *pushes*.

Proof of Lemma 1: If M reads a symbol on the top of a pushdown store without ***popping*** it, we can simply make M to ***pop*** it then ***push*** it back. While the ***push*** command is performed, the other work head just stay doing nothing. □ (Lemma 1)

First we define computation graphs for M with respect to different inputs. On a fixed input w, for each computation path P, an undirected graph $G^P = < V,E >$ is defined as below. To define G^P, only the computation along path P will be considered.

For V,

(1) 2 special vertices s_1 and s_2 are in V, each representing an empty stack;

(2) if a symbol a is ***pushed*** onto pushdown store Si at step t, then (a,i,t) is a vertex in V.

For E,

(3) if a of (a,i,t) is pushed immediately upon b of (b,i,t'), then edge $((a,i,t),(b,i,t'))$ is in E;

(4) if at some step, a of $(a,1,t)$ is pushed or popped, and simultaneously, b of $(b,2,t')$ is pushed or popped, then the edge $((a,1,t),(b,2,t'))$ is in E.

Lemma 2: For each computation path P of M, G_P is planar.

Proof of Lemma 2: If we delete from G^P all the edges added by rule (4) (edges linking 2 pushdowns), then the resulting graph is a pair of trees, each for one pushdown. This is clear from the following observation. For each pushdown store, a set of consecutive pushing commands creates a linearly linked list (by rule (3)). The popping commands do not create any new node. And each time the command sequence switches from ***pop*** to ***push*** for a stack, a new branch on the corresponding tree grows.

Now arrange the 2 trees in the following way. The tree for $S1$ is put on the left, and it is oriented so that whenever a new branch grow, it grows to the left. That is, a right-to-left depth-first search would coincide the path P. The tree for $S2$ is put on the right and it is oriented so that the new branches grow only to the right. That is, a left-to-right depth-first search would coincide the path P. In Figure 1, the envelops around the 2 trees (each for a pushdown store) indicate the time flow, or the pushdown store head traversal paths on their corresponding trees. h_1 goes from right to left, and h_2 left to right.

From Figure 1, we 'straighten' the envelop of the time flow. Now the edges of rule (4) can be added. They are added according to the time flow. No two cross because time does not 'flow' back. After adding all the edges, we carefully fold our envelops back surrounding the trees, resulting the planar embedding we needed. □ (Lemma 2)

Lemma 3: G_P can be made constant degree bounded.

Proof of Lemma 3: There is only one way to make a node having high degree — to use rule (3): if M alternatively performs the commands ***pop*** and ***push***, a picture like the one in Figure 2(a) will be formed. To overcome this difficulty, we can simply modify M so that every time M switches from ***pop*** to ***push***, it first ***pushes*** in a dummy symbol, then does its regular ***push*** operation. And later on, when M pops out this dummy symbol, it just ignores the symbol. After this modification, the graph in Figure 2(a) would look like Figure 2(b), with at most $O(n)$ dummy nodes. The resulting G_P (for any P) for the modified machine has bounded degree and is still planar. □ (Lemma 3)

Now we are ready to do our simulation. Since G_P is planar, it has $\sqrt{n}$-separator by Theorem A. We can lay G_P on the only work tape of M', so that

(1) each vertex occupies $O(logn)$ space to specify the relative distance from itself to its neighbors (constant number of edges) on G_P;

(2) $O(nlogn)$ space for the whole graph G_P;

(3) (a) The $\sqrt{n}$-separator of G_P is in the middle, a set A of roughly half of the vertices of G_P is on the left, and a set B of the other half on the right.

(b) Recursively, for a non-separator-set C generated by (a) or (b), let $C = D \cup E \cup \sqrt{|C|}$-***separator*** where sets D and E each contains roughly $|C|/2$ elements. D and E should be separated by the $\sqrt{|C|}$-separator on the tape.

Notice that we do not have the graph G_P or even P at all. Even the above arrangement can not be computed efficiently, since otherwise one deterministic tape can simulate 2 deterministic pushdown stores fast which is impossible [M,V,L]. However, we can ***guess***. To simulate M, M' keeps a description of M under its work head (or in its finite control). If M accepts the input, then there is an accepting computation path P, and there is a planar computation graph of bounded degree G_P. M' guesses G_P, and then arrange G_P as above by ***guessing***. Now M' would simply simulate M (guess P). Each time M needs to ***push*** or ***pop*** a symbol, M' would simply follow the edges of G_P to find the position of the needed vertex. (If the guess was incorrect, then M' does not accept for this path.)

The total time used for this simulation is $O(nlogn*f(n)*logn)$, where $nlogn$ is the length of the tape used, and the second $logn$ is costed by the $logn$-sized counter dragged by the work head, and $f(n)$ is the maximum number of edges crossed at some cell on the work tape. But each vertex, by above arrangement (3), can be crossed by at most

$$\sum_{i=0}^{\infty}\sqrt{\frac{n}{2^i}}=O(\sqrt{n})$$

edges. Therefore the total simulation time is $O(n^{1.5}\log^2 n)$. □ (Theorem 1)

Theorem 1': 1 tape can nondeterministically simulate 2 pushdown stores in $n^{1.5}\sqrt{logn}$) time.

Proof (*idea*): Instead of using 1 symbol per node in the graph, we divide the pushdown stores into blocks of size $logn$. Each $logn$-block will be a node in the graph. The graph is constructed as before. The bounded degree is taken care of as before. (Many different details will be presented in the final paper.) The resulting graph will be of size $\frac{n}{logn}$ and it has a $\sqrt{n/logn}$-separator. Do the simulation as before. Since the length of the tape this graph occupies is only n, the edge crossing at each cell is only $\sqrt{n/logn}$, the total running time is $O(n^{1.5}\sqrt{logn})$. (Notice that the inter-block computation requires only $\log^c n$ time for some c.) □ (Theorem 1')

4. On 1 tape vs. 2 tapes

Now we consider the original 1 tape vs. 2 tape question for on-line machines. The deterministic case is closed; the nondeterministic case is still open. Maass almost closed the gap for the later case with his $\Omega(n^2/log^2nloglogn)$ lower bound, and the author used his method to improve this to $\Omega(n^2/lognloglogn)$ [L1]. Recently the author found a paper by Freivalds who claimed a result (Theorem 2 in

[F]) which, if true, would immediately imply an $\Omega(n^2)$ lower bound, closing the gap. Now we are challenged by the following questions.

Question1. Can we really obtain the $\Omega(n^2)$ lower bound by the language of Maass or the language of Freivalds (as claimed in [F])?

Question2. Does the $\Omega(n^2)$ lower bound exist at all?

We will negatively answer Question 1 and discuss Question 2.

In [M], although a very general language L_I was introduced, only a simple subset, $\hat{L}$, of it was used. The language $\hat{L}$ can be defined as follows (w.l.g. let k be odd).

$$
\begin{aligned}
\hat{L}=\{ & b_0{}^1 b_1{}^1 \cdots b_k{}^1 \\
& b_0{}^2 b_0{}^3 b_1{}^2 b_2{}^2 b_1{}^3 b_3{}^2 \cdots b_{2i}{}^2 b_i{}^3 b_{2i+1}{}^2 \cdots b_{k-1}{}^2 b_{(k-1)/2}{}^3 b_k{}^2 \\
& b_0{}^4 b_{(k+1)/2}{}^3 b_1{}^4 b_2{}^4 b_{(k+3)/2}{}^3 b_3{}^4 \cdots b_{2j(mod k+1)}{}^4 b_j{}^3 b_{2i+1(mod k+1)}{}^4 \cdots b_{k-1}{}^4 b_k{}^3 b_k{}^4 \\
& \mid b_i{}^1 = b_i{}^2 = b_i{}^3 = b_i{}^4 \text{ for } i=0, \cdots, k \}
\end{aligned}
$$

The length of each $b_i{}^j$ (a binary string) may be different. We can also define a delimited version L^* of $\hat{L}$ where every $b_i{}^j$ in $\hat{L}$ is replaced by $*b_i{}^j*$ and they are of a fixed length.

The language, B, constructed in [F] is similar (but less complete) to $\hat{L}$ if we let $c(i)=a(i)b(i)$ and replace single $a(i)$ or $b(i)$ by $c(i)$ in the following. Here is the construction of [F]. Let B' consist of all strings

$$
\begin{aligned}
& a(1)b(1)a(2)b(2) \cdots a(2n)b(2n)2a(2n)b(2n)b(2n-1)a(2n-1)b(2n-2)b(2n-3) \cdots \\
& \cdots a(n+1)b(2)b(1)
\end{aligned}
$$

where all $a(i)$ and $b(i)$ are from $\{0,1\}$. B is defined to be the set of all strings $0x$ or $1y$, where $x \in B'$ and $y \in \overline{B'}$.

In [L1], based on [M]'s approach, we have shown that languages $\hat{L}$ and L^* requires $\Omega(\frac{n^2}{lognloglogn})$ time which implied the following theorem.

Theorem: It requires $\Omega(n^2/lognloglogn)$ time to nondeterministically simulate 2 tapes by 1 tape for on-line computation.

But the following theorem will show that one cannot do much better from this language, which negatively answers Question 1.

Theorem 2: $\hat{L}$ (L^* and B) can be accepted in $O(\frac{n^2 loglogn}{\sqrt{logn}})$ time by a 1-tape *nondeterministic* on-line machine.

Proof of Theorem 2: We design a 1-tape on-line nondeterministic machine to accept $\hat{L}$. According to (the remark of) Theorem B, we will distribute each sequence $\{b_i{}^j\}$ on a separate track of the work tape by the following rules:

(1) if $\#bin(b_p{}^j) < \#bin(b_q{}^j)$, then $b_p{}^j$ is put on the left of $b_q{}^j$;

(2) if $\#bin(b_p{}^j) = \#bin(b_q{}^j)$, and $b_p{}^j < b_q{}^j$ then $b_p{}^j$ is put on the left of $b_q{}^j$.

There is one more problem to be solved: M has to keep track which block it is reading and thereby decides where to put the block. It may 'drag' a counter under the work head, but this results an intolerable running time $O(\frac{n^2 logn}{\sqrt{logn}})$. The techniques for 'dragging' a counter that do not cause the *logn* delay developed in [P3] and [F1] cannot be used here, since only one work tape is available.

Surprisingly, the major part of the counter never need to be moved at all. We divide the work tape into four tracks: track i, i=1,2,3,4. Each track is further divided into 4 subtracks: subtrack i, i=1,2,3,4. For each track, we will keep a static counter $C0$ (of length *logn*) which never moves, and we keep two other counters $C1$ and $C2$ of length *loglogn* moving with the work tape head. $C1$ holds the most recent value of the last *loglogn* bits of $C0$. $C2$ holds the number of 1's in the $C0$ (last *loglogn* bits are taken from $C1$). $C0$ is modified every time $C1$ overflows. The algorithm of M is outlined as below.

(1) M lays the following frame on each track ($C0=C1=C2=0$):

Subtrack 1: $\# \ 1 \ \# \ space_1 \ \# \ 2 \ \# \ space_2 \ \# \ \cdots \ \# \ logn \ \# \ space_{logn} \ \#$,

Subtrack 2: $C0$ (size *logn*),

Subtrack 3: $C1$ (size *loglogn*),

Subtrack 4: $C2$ (size *loglogn*).

The segment $\# \ m \ \# \ space_m \ \#$ on track j is explained as '$space_m$ will hold all blocks $b_i{}^j$'s such that $\#bin(i)=m$.' The size of each $space_m$ can be guessed. The size of each $\# \ m \ \#$ is *loglogn* where n can be guessed also.

(2) Sequence $\{b_i^j\}$ will be put in track j, where j can be found out by the finite control of M. At the time M is about to read b_i^j, it adds 1 to $C1$ in track j. If $C1$ overflows, M goes back to change $C0$ in track j, and then bring back a number, of length at most $loglogn$, which is the number of 1's decreased or increased in $C0$. (We do not bring along $C1$ and $C2$ with the work head in this process.) Then modify $C2$ in track j accordingly.

(3) Locate the work head to $\# \; l \; \#$ where l is equal to $C2$. Copy b_i^j (whose length can be guessed) on to the first available place in $\# \; space_l \; \#$ in sub-track 1 of track j. In this process $C1$ and $C2$ are moved along with the work head.

(4) After the whole input is read, check if $b_i^1 = b_i^2 = b_i^3 = b_i^4$, for all i, in the obviously way.

The time complexity of this algorithm is analyzed as follows. Step (1) takes no more than $O(nlogn)$ time. In step (2), the overflow can happen at most $O(n/logn)$ times. Each time the overflow happens, modifying $C0$ and $C2$ requires at most $O(nloglogn)$ time. Thus step (2) takes $O(\frac{n^2loglogn}{logn})$ time. By Lemma 3.2.6C, step (3) takes $O(\frac{n^2loglogn}{\sqrt{logn}})$ time. Step (4) takes only $O(n)$ time. Thus the total time is $O(\frac{n^2loglogn}{\sqrt{logn}})$. The proof is therefore complete. □ (Theorem 2)

Corollary 2: Language L^* (and B) can be accepted by a 1 tape deterministic on-line machine in $O(n^2loglogn/\sqrt{logn}\,)$ time. □ (Corollary 2)

The author is currently working on Question 2 and he believes that it can be answered negatively. Other upper bounds have been obtained. For example, 1 non-deterministic tape can probabilistically simulate 2 tapes in less than square time with any fixed small error ϵ (*i.e.* reject with $Pr(exist\ accepting\ path) \leq \epsilon$ accept if there is a path P, $Pr(P\ accepts) > 1-\epsilon$). Also W. Ruzzo showed that a multitape Σ_k ATM running in time T can be simulated by a 1 tape $\Sigma_{k'}$ ATM in time $O(TlogT)$ where $k' = k+1$ if k is odd, $k' = k$ otherwise [R2], the $k=1$ case has been proved by N. Pippenger [R2].

5. Applications of Theorem 1

Application 1: 3 pushdowns vs. 2 pushdowns

In [DG], Duris and Galil listed 3 open questions. They answered two of them and left one open: Are three pushdown stores better than two (for nondeterministic machines)? Since we know that 3 pushdowns are as powerful as k tapes for any k, and 1 pushdown can only accept CFL's, this question is really the last unknown question to us. Using above powerful simulation we now answer this question positively.

Theorem 5: Three pushdown stores are better than two pushdown stores for nondeterministic machines.

Proof of Theorem 5: The following lemma is needed, after the current author proved this lemma, he found out that this is an old result proven in [BG] and actually mentioned in [DG]. We certainly refer the reader to [BG] for a proof.

Lemma 5[BG]: For nondeterministic machines, 3 pushdown stores are equivalent to 2 (therefore k) tapes. □ (Lemma 5)

Suppose Theorem 5 is not true. Then 2 pushdown stores can simulate 3 pushdown stores in linear time which in turn, by Lemma 5, can simulate 2 (hence k) tapes in linear time. By Theorem 1, this implies that 1 tape can simulate 2 pushdowns and therefore 2 tapes in $O(n^{1.5}\sqrt{logn})$ time. However, this contradicts to the Maass' lower bound [M]. Hence the theorem. □ (Theorem 5)

A careful calculation in above proof yields the following.

Corollary 5: It requires $\Omega(\frac{n^{4/3}}{\log^{\alpha} n})$ time to simulate 3 pushdown stores by 2 pushdown stores for nondeterministic machines.

Application 2: 1 tapes vs. 1 queue

The question of tapes vs. queues were investigated in [V]. He showed: for strong 'on-line' (with an output each step) machines, it requires $\Omega(n^2)$ time to deterministically simulate 1 queue (first in first out device) by 1 tape. He also modified the concept of nondeterminism to get an $\Omega(n^2)$ lower bound. In [L1], we showed that for the weak on-line model (where the output is produced only at the end), it required $\Omega(n^{4/3}/logn)$ to simulate 1 deterministic queue by 1 nondeterministic tape. It was also hoped that $\Omega(n^2)$ lower bound may be obtained. But the following theorem breaks this hope.

Theorem 6: A 1-queue machine running in time $O(t)$ can be simulated by a 1-tape nondeterministic machine in $O(t^{1.5}\sqrt{logt})$ time.

Proof (*idea*): We can give two proofs. The first one is based on the observation that a similarly constructed computation graph as in Theorem 1 for a 1 queue machine is also planar.

The second proof follows from the assertion that two pushdown stores can simulate 1 queue in linear time. □ (Theorem 6)

Open Question: What left here is a gap between $\Omega(n^{4/3}/logn)$ and $O(n^{1.5}\sqrt{logn})$. We believe it is the lower bound that should be significantly improved.

Application 3: off-line TM's

There has been very few results on off-line TMs (with 2-way input tape) regarding our tape reduction problem. No significant lower bound has ever been obtained. No progress in upper bounds is made for years. In next Theorem we take a first step towards closing the gaps for off-line machines.

Theorem 7: Theorems 1 and 3 are true for off-line machines.

The proof for this theorem follows directly from the former proofs.

Application 4: about language B

The language B in Section 4 was claimed to take $\Omega(n^2)$ time for a 1-tape nondeterministic on-line machine by [F] (Theorem 2 of [F]). This language allows us to further demonstrate the power of Theorem 1. Theorem 2 shows that B can be accepted by a 1-tape nondeterministic on-line machine in time $O(n^2 loglogn/\sqrt{logn})$, therefore Theorem 2 in [F] was not correct. Here, using Theorem 1, we show B is even much easier than that.

Lemma 8: B can be accepted by 2 pushdown stores in linear time.

Theorem 8: B can be accepted by a 1-tape nondeterministic on-line machine in time $O(n^{1.5}\sqrt{logn})$

6. Acknowledgements

Deep thanks to Professor Juris Hartmanis, my thesis advisor who so patiently guided my research, Professor John Gilbert, expert on graph separators who was a wonderful resource on graph theory, Professors Joel Seiferas and Yaccov Yesha for their encouragement, and Luc Longpre and Zhen Zhang, my fellow friends at Cornell who simplified my early proof of Theorem 1 [L^2] and helped me with the proof of Theorem B [Z^2].

7. References

[A] S.O. Aanderaa, On k-tape versus (k-1)-tape real time computation, in Complexity of Computation. R. Karp Ed. (1974) pp. 75-96.

[BGW] R.V. Book, S.A. Greibach, and B. Wegbreit, Time- and tape-bound Turing acceptors and AFL's, JCSS 4,6 (Dec. 1970) pp. 606-621.

[DGPR] P. Duris, Z. Galil, W.J. Paul, and R. Reischuk, Two nonlinear lower bounds, Proc. 15th ACM STOC (1983) pp. 127-132. (Revised June 1983)

[DG] P. Duris and Z. Galil, Two tapes are better than one for nondeterministic machines, Proc. 14th ACM STOC (1982) pp. 1-7.

[F] R. Freivalds, Probabilistic machines can use less running time, Info. Processing, 77 (1977) pp. 839-842.

[F1] M. Furer, The tight deterministic time hierarchy, Proc. 14th ACM STOC (1982) pp. 8-16.

[G] J.R. Gilbert, Graph separator theorems and sparse gaussian elimination, Ph.D. Thesis, Stanford Univ. (1980).

[HS] J. Hartmanis and R.E. Stearns, On the computational complexity of algorithms, Trans. Amer. math. Soc. 117 (1965) pp. 285-306.

[HS1] F.C. Hennie and R.E. Stearns, Two tape simulation of multitape Turing machines, J.ACM, 4 (1966) pp. 533-546.

[HU] J.E. Hopcroft and J.D. Ullman, Introduction to automata theory, languages, and computation, Addison-Wesley (1979).

[L] M. Li, On 1 tape versus 2 stacks, TR-84-591, Dept. of Comp. Sci., Cornell University (Jan. 1984).

[L1] M. Li, Lower bounds by Kolmogorov-complexity, to appear in ICALP'85, Nafplion, Greece. Also TR-85-666, Comp. Sci. Dept., Cornell University.

[L2] M. Li, Lower bounds in computational complexity, Ph.D. Thesis, TR-85-663, Dept. Comp. Sci., Cornell University (Jan. 1985).

[LT] R.J. Lipton and R.E. Tarjan, A separator theorem for planar graphs, SIAM J. App. Math. 36 (1979) pp. 177-189.

[LT1] R.J. Lipton and R.E. Tarjan, Applications of a planar separator theorem, SIAM. J. Comp. 9 (1980) pp. 615-627.

[M] W. Maass, Quadratic lower bounds for deterministic and nondeterministic one-tape Turing machines, Proc. 16th ACM STOC (May 1984) (See revised version).

[P2] W.J. Paul, On-line simulation of $k+1$ tapes by k tapes requires nonlinear time, Proc. 23rd IEEE FOCS (1982) pp. 53-56.

[P3] W.J. Paul, On time hierarchies, Proc. 9th ACM STOC (1977) pp. 218-222.

[PSS] W.J. Paul, J.I. Seiferas, and J. Simon, An information-theoretic approach to time bounds for on-line computations, Proc. 12th ACM STOC (1980) pp. 357-367.

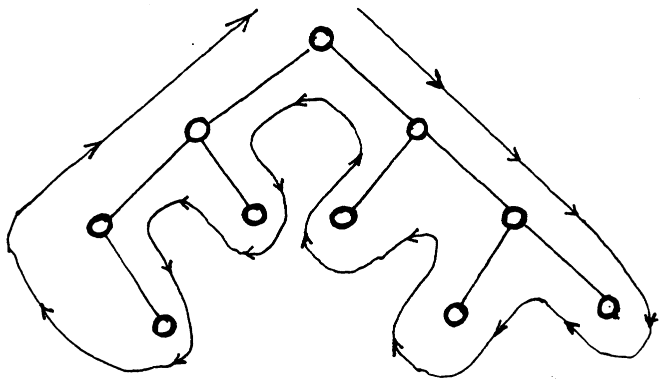
[R] M.O. Rabin, Real time computation, Israel J. of Math, 1,4 (1963) pp. 203-211.

[R2] W. Ruzzo, *Private communication*. (1984)

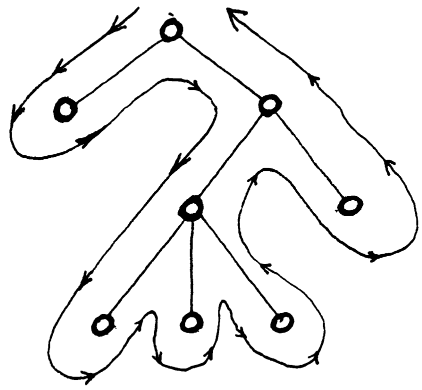
[V] P.M.B. Vitanyi, One queue or two pushdown stores take square time on a one-head tape unit, Center for Math. Comp. Sci., Amsterdam (Revised Version, 1984).

[V1] P.M.B. Vitanyi, On the simulation of many storage tapes by one, 10th ICALP'83, Lecture Notes in Computer Science 154, Spring Verlag, pp. 687-694.

List of Figures

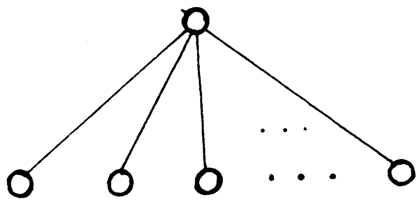


(a) tree for S1

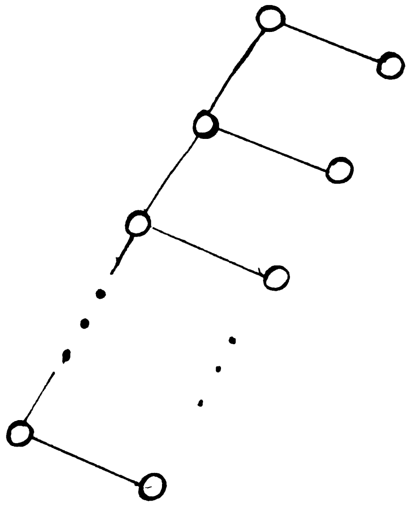


(b) tree for S2

Figure 1



(a) unbounded degree



(b) bounded degree

Figure 2