Lecture 13

Time-Space Tradeoff Lower Bounds Using Branching Program

May 13, 2008
Lecturer: Paul Beame
Notes: Dang-Trinh Huynh-Ngoc

From last class, we know that any function on n bits computable by a Turing machine (or RAM, respectively) in time T and space S can be computed by a branching program (BP) (or R-way BP, respectively) of length T and size at most $2^{S+\log_2 n}$ (or $2^{S+\log_2 n+\log_2 T}$ for a levelled BP). In this lecture we will see how to use branching programs to prove time-space tradeoff lower bounds.

13.1 Time-space trade-off lower bounds for multi-output problems

In this lecture we also consider multi-output functions. The standard space-bounded model for computing multi-output functions (the analog of a log-space transducer) has a separate read-only input tape a separate write-only output tape as well as working tapes. Only the working tapes are counted towards the amount of space used by an algorithm, and the output tape is written from left to right. For the above simulation to continue to hold, we need to make a small modification to the BP model to avoid it having its nodes represent the contents of the output tape. In particular, we allow some of the edges to be labeled by output values. That is, any edge can have (multiple) labels of the form "$y_i = a$", where y is the output string, which means that a is the i-th value of the output string.

We consider the problem of sorting N integers from $[N]$. Some of the basic sorting algorithms that apply to this case include the bucket sort algorithm, which gives us time complexity of $O(N)$ and space complexity of $O(N \log N)$, since we need only $O(\log N)$ bits to represent the number of occurrences of any item, or selection sort, which gives us time complexity $O(N^2)$ but space complexity of $O(\log N)$. These algorithms gives time-space product of $O(N^2 \log N)$. It is easy to produce algorithms achieving time-space product of $O(N^2 \log N)$ for any time bound between linear and quadratic. By some clever ideas, Pagter and Rauhe gave a sorting algorithm with time-space product of $O(N^2)$ for time $\Omega(N \log N)$. Borodin-Cook proved a lower bound of $\Omega(N^2/\log N)$ on the time-space product for sorting and introduced the technique we will use below.

In this lecture we will prove a lower bound of $\Omega(N^2)$ on the time-space product for problems related to sorting. Similar lower bounds can be proved for many other problems include computing $A \cdot x$ for fixed matrices A whose submatrices have full rank, computing universal hash functions, and pattern matching. We give just one example to illustrate some general features of the technique.

We define the *UNIQUE-ELEMENTS* problem as follows: Given an input of N integers $x_1, \ldots, x_N$ in $[N]$, output all i such that x_i appear exactly once in the sequence. We will prove the following theorem.

Theorem 13.1 Any algorithm computing *UNIQUE-ELEMENTS* that runs in time at most T and uses space at most S must have that $T \cdot S$ is $\Omega(N^2)$.

Corollary 13.2 Any RAM algorithm sorting N elements in $[N]$ that outputs elements in sequence, runs in time at most T and uses space at most S must have $T \cdot S = \Omega(N^2)$.

Proof This follows because *UNIQUE-ELEMENTS* can be reduced to sorting if we assume that the output tape is written in left-to-right order. $\square$

Since any RAM algorithm computing *UNIQUE-ELEMENTS* in time at most T and uses space at most $S \geq \log_2 N$, there a corresponding there is an N-way levelled BP that computes *UNIQUE-ELEMENTS* of length T and size at most $2^{S+\log_2 N+\log_2 T}$ which is $2^{O(S)}$ for relevant values of T.

Theorem 13.3 If an N-way levelled branching program with length T and size at most 2^S computes *UNIQUE-ELEMENTS* then $T \cdot S$ is $\Omega(N^2)$.

Proof Let the input to the *UNIQUE-ELEMENTS* problem be given by the uniform distribution over $[N]^N$. Since for x_i to be unique, each of the other $N-1$ elements most avoid its value, the expected output size for *UNIQUE-ELEMENTS* would then be

$$E[\#\text{ outputs for } UNIQUE\text{-}ELEMENTS] = N(1-1/N)^{N-1} > N/e$$

since $(1-1/N)^N < 1/e < (1-1/N)^{N-1}$ for every integer $N \geq 1$.

By Markov's inequality, we can bound the probability that the output size is at least $N/(2e)$:

$$\Pr[\#\text{ outputs } > \frac{N}{2e}] \geq \frac{1}{2e-1}.$$

(Let α be the probability in question. Then the expected number of outs would be at most $N \cdot \alpha + \frac{N}{2e}(1-\alpha)$ and hence $\alpha + (1-\alpha)\frac{1}{2e} \geq \frac{1}{e}$. Therefore $\alpha(1-\frac{1}{2e}) \geq \frac{1}{2e}$ which yields the claimed bound.)

Let $\mathcal{P}$ be the levelled branching program computing *UNIQUE-ELEMENTS*. It is clear that $\mathcal{P}$ must query all input values, i.e $T \geq N$. For some $h \leq N/2$ to be chosen later, we partition $\mathcal{P}$ into T/h *layers* of height h each. (Each layer begins at a level that is a multiple of h, called a *boundary* and includes h levels of edges.) We call an input on which *UNIQUE-ELEMENTS* outputs at least $N/(2e)$ output values a "good input". It follows that $\mathcal{P}$ run on any good input must produce at least

$$\frac{N}{2e(T/h)} = \frac{Nh}{2eT}$$

output values in some layer. Define $m = \frac{Nh}{2eT}$. Thus, for any good input, there exists a node v at a boundary level such that the sub-program $\mathcal{P}'$ rooted at v in that layer produces at least m correct output values on this input. The following claim says that this happens with very small probability.

Claim: If $m \leq N/4$ and $h \leq N/4$, then for any $\mathcal{P}'$ of height at most h,

$$\Pr[\mathcal{P}' \text{ outputs at least } m \text{ correct values }] \leq e^{-m/2},$$

where the probability is taken over the uniform distribution on $[N]^N$.
Proof Since

$$\begin{aligned}&\Pr[\mathcal{P}' \text{ outputs } \geq m \text{ correct values }] \\ &= \sum_{\text{paths } \pi} \Pr[\text{ the execution follows } \pi] \cdot \Pr[\mathcal{P}' \text{ outputs } \geq m \text{ correct values } \mid \text{ the execution follows } \pi].\end{aligned}$$

it suffices that for each path π from the root to a sink in $\mathcal{P}'$ we prove that

$$\Pr[\mathcal{P}' \text{ outputs } \geq m \text{ correct values } \mid \text{ the execution follows } \pi] \leq e^{-m/2}.$$

Let s be the number of input variables that are not mentioned along π. Hence $s \geq N-h-m \geq N/2$. We also let t to be the number of variables that are output but are not queried along π. Thus the number of variable queries is $N-s-t$, and the number of input consistent with π is N^{s+t}. Therefore, the total number of inputs such that $\geq m$ output values on π are correct is at most $(N-m)^s \cdot N^t$, where the first factor is the number of assignments for the s variables that avoid the $\geq m$ output values.

Thus the claimed probability is at most

$$\frac{(N-m)^s N^t}{N^{s+t}} = (1-\frac{m}{N})^s \leq e^{-ms/N} \leq e^{-m/2}.$$

$\square$

Choosing $h = N/4$, we have $m = \frac{\lceil N^2 \rceil}{8eT} \leq N/4$ since $T \geq N$. By the claim, there is at most an $e^{-m/2}$ fraction of inputs that is mapped to any node in any boundary level such that the subprogram of height h rooted at this node outputs at least m correct outputs. Since the fraction of "good inputs" is at least $\frac{1}{2e-1}$, we need

$$2^S \cdot e^{-m/2} \geq \frac{1}{2e-1}.$$

Therefore $S \geq Cm$ for some constant $C > 0$. Since $m = C'N^2/T$ for some constant C', we have $S \geq CC'N^2/T$ so ST is $\Omega(N^2)$. $\square$

13.2 Time-space trade-off lower bounds for decision problems

In this case we cannot use the progress measure of the number of output values produced to analyze the complexity. Instead ideas from communication complexity are used. We sketch the approach in the context of oblivious branching programs using a variation of a method due to Alon and Maass.

Theorem 13.4 Let $f : [R]^n \to \{0,1\}$. Suppose that there is a constant $c > 0$ such that for any set of variables S with $|S| = 2m$ for some m there is a restriction ρ with $unset(\rho) = C$ such that for any partition of C into two sets $A, B \subseteq [n]$ of size m, $D^{cc}(f|_\rho(A,B)) \geq cm$ then if there is an oblivious R-way branching program of length T and size at most 2^S computing f then T is $\Omega(n \log(n/S))$.

Proof Let $\mathcal{P}$ be such a branching program. Define $k = \lceil T/n \rceil$. Since $\mathcal{P}$ is oblivious we can define a count, q_i, of the number of times that variable x_i is queried along any path. Since $T \leq kn$ the average value of q_i is at most k. Therefore, at least $n/2$ variables x_i have $q_i \leq 2k$.

We now divide the branching program into r layers of height $h = T/r = kn/r$ for some r that we will fix later. We want to argue that there is an assignment of the layers to the players so that each player receives layers containing all queries the branching program makes to a subset of variables of size m, each of which is queried at most $2k$ times. We will do this using the probabilistic method.

We randomly assign each layer to one of the two players independently. Since each remaining variable x_i is queried at most $2k$ times, the probability that all of the up to $2k$ layers in which it is queried are assigned to a given player is at least 2^{-2k}. Therefore the expected number of variables that are only assigned to a given player is at least $\mu = n2^{-2k=1}$.

Although the expected value is large this is not enough to say that both can be achieve simultaneously. To do this we use the second moment method. The assignment of variables read in the same layer is highly correlated. However since each of the remaining variables is queried in at most $2k$ layers each of height h its assignment is correlated to that of at most $2kh = 2k^2n/r$ other variables. If we choose r to be $\Omega(k^2 2^{2k})$ then this is at most $\epsilon n/2^{2k} \leq \epsilon\mu$ variables for some constant $\epsilon > 0$. Using Chebyshev's inequality one can then show that there is positive probability that for both players at least $m = \mu/2 = n/2^{2k+2}$ variables are only assigned to that player. Let A and B be the first m such variables for each player respectively and C be the union of A and B. Let ρ be the restriction given by the conditions of the lemma.

We now show how both players can use $\mathcal{P}$ to compute $f|_\rho$ with inputs partitioned into A and B using small communication. They will start at the root and simulate the execution of $\mathcal{P}$. Alice will simulate $\mathcal{P}$ on her layers and Bob will simulate $\mathcal{P}$ on his layers. Whenever the layer assignment switches from one player to the other, that player will communicate the name of the boundary node in $\mathcal{P}$ between the two layers so that the other player can continue the simulation. Each such communication takes at most S bits and there will be a total of at most r names sent.

Therefore by the hypothesis we have $Sr \geq cm = cn/2^{2k+2}$. Given the value of r we have $S \geq c'n/(2^{4k}k^2) \geq c''n/2^{6k}$ for some constants $c', c'' > 0$ so $2^{6k} \geq c''n/S$. Therefore using $T \leq kn$ we have $2^{6T/n} \geq c''n/S$ so T/n is $\Omega(\log(n/S))$ and therefore T is $\Omega(n\log(n/S))$. $\square$

Although we won't go into details one can produce such decision problems f. For oblivious branching programs, the best lower bound we know for an explicit decision problem is $T = \Omega(n \log^2 \frac{n}{S})$ due to Babai, Nisan and Szegedy and uses multi-party communication complexity.

For the general R-way branching program model, the best we know for decision problems is $T = \Omega(n \log(\frac{n \log n}{S}))$ due to Beame, Jayram, and Saks and extends methods used for read-k branching programs due to Borodin, Razborov, and Smolensky. The argument can no longer directly use communication complexity since the variable partition can now be input dependent.

Ajtai was the first to produce superlinear time-space tradeoffs for Boolean branching programs and for R-way programs for *ELEMENT-DISTINCTNESS*. The best current lower bound for Boolean branching programs is $T = \Omega(n\sqrt{\frac{\log}{\log\log}(\frac{n}{S})})$. Also in this R-way model, we know the same lower bound of $T = \Omega(n\sqrt{\frac{\log}{\log\log}(\frac{n}{S})})$ for the *ELEMENT-DISTINCTNESS* problem. These results are due to Beame, Saks, Sun, and Vee.