Lower bounds for padded sorting and approximate counting

R.Sarnath
Deptt of Computer Science
226, Bell Hall
SUNY at Buffalo
Amherst NY 14260
email: sarnath@cs.buffalo.edu
Phone (716) 636-3774
Keywords: lower bounds, sorting, circuit complexity

Abstract

We examine the relationship between running time and error of parallel sorting algorithms. This is done by applying Hastad's main lemma to relate the size depth and error of simple circuits, that sort an input of 0's and 1's. As a consequence, we obtain lower bounds for approximate counting as well.

1 Introduction

Recently, attempts have been made to sidestep the lowerbound of Beame and Hastad [1], by separating the sorting problem from the list ranking problem. These attempts have been of two kinds - *chain sorting*, i.e., sorting the elements into a linked list [2], and *padded sorting*, i.e., sorting into a larger array [3,5,6]. The algorithms in [5] require the elements to be uniformly distributed in a certain range, whereas the algorithms in [3,6] are for the general case. The question we examine here, is *what must the size of the output array be, if the lowerbound of [1] must be circumvented?* In other words, if an algorithm sorts n items using polynomially many processors in time $o(\frac{\log n}{\log\log n})$, what is the minimum required size of the ouput array? A lowerbound of $n + f(n)$, where $f(n)$ is $o(n^\epsilon)$ for any constant $\epsilon > 0$, on the size of the output array is trivial - *if such a padded-sort were to be accomplished, the "extra" $f(n)$ elements could then be chained and eliminated in time $o(\log n/\log\log n)$, thus accomplishing a conventional sort. This implies that a running time of $\frac{\log n}{c+\log\log n}$ is unavoidable if the size of the final array is to be $n + o(n^\epsilon)$.* In this paper, we prove a better lowerbound - we have shown, as a consequence of the main lemma in [5], that any circuit of size polynomial in the size of the input, that sorts n items into an array of size $n+n^\epsilon-1$, $0<\epsilon<1$, requires depth at least $\Omega(\frac{(1-\epsilon)\log n}{\log\log n+c})$, for some constant c.

In order to prove the lowerbound for padded sorting, we shall restrict ourselves to the problem of sorting an array of 0's and 1's. Clearly, any lowerbound derived for this problem, applies to the general problem as well. Before we proceed further, we shall parametrize the "error" in the padded sort, i.e, the increase in the size of the output array, by a function f, as follows:

Definition : *A padded sort of n items is $f(n)$-approximate, if the size of the output array is $\leq (n + \frac{n}{f(n)} - 1)$.*

The $f(n)$-approximate padded-sorting problem is formally defined as follows:

Input : *A vector $A = (a_0, a_1, ..., a_n)$, $A \in \{0,1\}^n$. A_0 denotes $|\{a_i : a_i = 0, 1 \leq i \leq n\}|$, and A_1 denotes $|\{a_i : a_i = 1, 1 \leq i \leq n\}|$.*

Output : *A vector $B = (b_0, b_1, ..., b_m)$, $m = n+\frac{n}{f(n)}-1$, $B \in 0^k 1^{m-k}$, such that $A_1 \leq (m-k) \leq (m - A_0)$.*

Note that as we vary $f(n)$ from 1 to n, the size of the output array varies from $2n-1$ to n. To get an output of size n, we have to do a conventional sort. It is easily shown that there exists a depth-1 circuit that produces an output of size $2n-1$. If the size of the output is $> 2n-1$, there is a trivial solution to the problem, and is therefore uninteresting.

We can now make the connection between the above problem and that of *$f(n)$-approximate counting*. Given input as specified above, the output of an $f(n)$-approximate counting would be the binary representation of $(m-k)$. *It is easy to see that there exists a depth-1, logspace-uniform circuit family with unbounded fan-in, that transforms the output of an $f(n)$-approximate counting circuit, into the output of the corresponding $f(n)$-approximate padded sorting circuit and vice-versa.* Thus, a depth-k lowerbound on circuits for one of these problems, implies a depth-$(k-1)$ lowerbound for the other. As it turns out, it is easier to derive the lowerbound for the case of padded sorting.

2 A lowerbound for padded sorting

Our approach is very similar to the one in [4], and we shall use the main lemma therein, along with the method of restrictions, to obtain our result. However, the problem does not lend itself to a neat inductive solution, as is the case with parity, primarily because we are using an arbitrary function $f(n)$, to characterise the error. We shall therefore outline a step-by-step procedure to reduce a depth-k circuit, with bounded size and bottom fan-in, to a depth-2 circuit, which we shall show, does not exist. This would imply that the depth-k circuit that we started with, does not exist. For the sake of completeness, we shall reproduce here, some definitions, and the main lemma from [4].

Definition: *A* **restriction** ρ*, is a mapping of the set of variables* $\{x_1, x_2, ..., x_n\}$ *to the set of values* $\{0, 1, \star\}$.

$\rho(x_i) = 0$*, means* x_i *is replaced by* 0.

$\rho(x_i) = 1$*, means* x_i *is replaced by* 1.

$\rho(x_i) = \star$*, means* x_i *remains a variable.*

When we say that the set $\{x_1, x_2, ..., x_3\}$ *is* **hit** *by* ρ*, we mean that each* x_i *is replaced by* $\rho(x_i)$*. When we say that a function* G *is hit by* ρ*, we mean that the set of inputs to* G *is hit by* ρ.

Definition : *A* **random restriction** $\rho \in R_p$ *satisfies the following conditions:*

$\rho(x_i) = 0$*, with probability* $\frac{1}{2} - \frac{p}{2}$

$\rho(x_i) = 1$*, with probability* $\frac{1}{2} - \frac{p}{2}$

$\rho(x_i) = \star$*, with probability* p

independently for different x_i.

Main Lemma :*Let* G *be an* AND *of* OR*s, all of size* $\leq t$*, and* ρ *a random restriction from* R_p*. Then, the probability that* G*, after being hit by* ρ*, cannot be written as an* OR *of* AND*s, all of size* $< s$*, is bounded by* α^s*, where* α *is the unique positive root of the equation,*

$$(1 + \frac{4p}{1+p}\frac{1}{\alpha})^t = (1 + \frac{2p}{1+p}\frac{1}{\alpha})^t + 1 \tag{1}$$

We make the following observations about this result:

1. Provided that p is $o(1)$, an elementary argument shows that $\alpha < 5pt$.
2. The result also holds in a dual form, to write an OR of ANDs as an AND of ORs.
3. If $t = 1$, then $\alpha = \frac{2p}{1+p}$.

To clarify our circuit model further, what we have is a monotone boolean circuit, with alternating levels of AND and OR gates, n-inputs, and m ouputs. The inputs are at the lowest level of the circuit, and the outputs appear at the top. We can look at this m circuits, computing m boolean functions. This circuit computes an $f(n)$-approximate padded-sort. Our theorem holds for any function $f(n)$, *as long as* $\frac{1}{\log f(n)}$ **is** $o(1)$, *and* $\log(f(n))$ **is** $o(\frac{n}{f(n)})$.

We shall first prove the following lemma, which serves as the "base case".

Lemma 1: *Let C be a circuit of depth 2, with C_I inputs and C_O outputs, $2C_I \geq C_O \geq C_I$, that computes a padded sort. Then C must have bottom fan-in at least $\beta = C_I - C_0/2 - 2$.*

Proof : Let us assume that the gates at depth 2 are all OR gates. (A similar argument holds for the other case as well.) Suppose a circuit exists that has bottom fan-in $\beta - 1$. Consider the circuit that computes the $\lceil C_O/2 \rceil$-th bit of the output. By setting some $\beta - 1$ inputs to 0, we can force this bit of the output to be 0. But the remaining $C_I - \beta + 1$ bits of the input could be 1. This would imply that C does not compute a padded sort. $\Box$.

Theorem 1: *No circuit for computing an $f(n)$-approximate padded sort exists, that has depth less than $d_0 = \frac{\log f(n)}{d + \log\log f(n)} + 2$, for some $d > 0$, and atmost 2^s gates, where $s \leq c \log f(n)$, for some $c > 0$.*

Proof: Suppose, by way of contradiction, such a circuit does indeed exist. We shall reduce the depth of the circuit, one level at a time, until we are down to a circuit of depth 2.

In the first step we shall transform this circuit into a depth d_0 circuit that has bottom fan-in bounded by s. To do this, we view the circuit as a depth $d_0 + 1$ circuit, with bottom fan-in 1. We then hit this circuit with a random restriction $\rho \in R_p$, $p = \frac{1}{10}$. The main lemma then assures us that we can rewrite all the CNF formulas that represent the output of gates at depth 2, (we can assume that these are CNF formulas, without any loss of generality) as DNF formulas, each having bottom fan-in bounded by s. This gives us two adjacent levels of OR gates, which can then be collapsed into one level. This yields a circuit that has $0.1n$ inputs, atmost 2^s gates at depth 2 or more, and bottom fan-in bounded by s. Let it be that ρ set x_0 bits to 0, and x_1 bits to 1. We then ignore the leftmost x_0 output bits, and the x_1 rightmost output bits. Note that this circuit no longer computes an $f(n)$-approximate padded sort, since the size of the input is $0.1n$, whereas the size of the output is $(0.1n) + \frac{n}{f(n)} - 1$.

In each subsequent step, we do the following:

- Hit the circuit with a restriction ρ from R_p, where $p = \frac{1}{10c \log f(n)}$.
- Assume, without loss of generality, that the gates at depth 2 are AND gates. We can rewrite the CNF formulas that represent the outputs of these AND gates as DNF formulas, such that the circuits corresponding to the DNF formulas also have bottom fan-in bounded by s. Since, by the assumption made earlier, p is $o(1)$, the probability that any such conversion cannot be carried out, is bounded above by $\frac{1}{2^s}$. Since we have atmost 2^s gates at depth 2, we can carry out the conversion for all these gates, with some finite probability of success. In particular, a restriction ρ exists, that enables to carry out these conversions. This gives us two adjacent levels of OR gates, that can be collapsed to into 1 level, without any increase in the number of gates at levels 2 or more.
- If ρ sets x_0 bits to 0, and x_1 bits to 1, we ignore the leftmost x_0 output bits, and the x_1 rightmost output bits.

Consider the circuit after $\frac{\log f(n)}{d+\log\log f(n)}$ steps. The number of inputs is now $n_0 = 0.1n[10c\log f(n)]^{\frac{-\log f(n)}{d+\log\log f(n)}}$, which is $\geq 4n/f(n)$, for large enough n. The number of outputs is $n_0 + \frac{n}{f(n)} - 1$. This circuit has depth 2, and fan-in bounded by $c\log f(n)$. By lemma 1, and the condition that $\log(f(n))$ is $o(\frac{n}{f(n)})$, such a circuit cannot exist. This concludes the proof. $\Box$

Since the function n^ϵ satisfies the conditions we imposed on f, we have the following result.

Corollary 1: *For any polynomial P, and any $\epsilon > 0$, no circuit of depth $d_0 = \frac{(1-\epsilon)\log n}{d+\log\epsilon+\log\log n}$ with atmost $P(n)$ gates exists, that can compute an n^ϵ-approximate padded sort.*

Corollary 2:*No circuit of depth $d_0 - 1$ exists, for n^ϵ-approximate counting, that satisfies the above constraints.*

3 Conclusion

The best upper bound on the size of the output array for general sorting is $O(n)$ [4]. Clearly some work needs to be done to close this gap.

If $f(n)$ were $\log^c n$, we have a randomised PRAM algorithm for $f(n)$-approximate counting that uses $o(n)$ processors and $O(\log\log n)$ time. This algorithm works by sampling $log^d n$ inputs (for suitable d) uniformly at random. Using only polylog(n) processors and $O(\log\log n)$ time, we can compute the number of 1's in the sample. A direct application of the Chernoff bounds, assures us that the count we estimate by sampling is within $n/f(n)$ of the actual count, with high probability.

We therefore conjecture, that if we were to restrict ourselves to the $\{0,1\}$ case, the bounds in this paper are tight.

References

[1] Beame P. and Hastad J., Optimal bounds for decision problems on the CRCW PRAM. J of the ACM, 36(3):643-670, July 1989.

[2] Hagerup T., Fast parallel space allocation, estimation and integer sorting, tech rep MPI-I-91-106, Max Planck Institut fur Informatik, June 1991.

[3] Hagerup T. and Raman R., Waste makes haste - tight bounds for loose parallel sorting, to appear in Proc of 33rd annual IEEE FOCS.

[4] Hastad J., Almost optimal lower bounds for small depth circuits, in proc of 18th annual ACM STOC, pp 6-20, 1986.

[5] Mackenzie P. and Stout Q., Ultra-fast expected time parallel algorithms. Proc. of symposium on discrete algorithms, 414-424, 1991.

[6] Sarnath R., Doubly logarithmic time parallel sorting, manuscript.