Bounds on the Separation of Two Parallel Disk Models

Chris Armen *

University of Hartford

Abstract

The single-disk, D-head model of parallel I/O was introduced by Agarwal and Vitter to analyze algorithms for problem instances that are too large to fit in primary memory. Subsequently Vitter and Shriver proposed a more realistic model in which the disk space is partitioned into D disks, with a single head per disk. To date, each problem for which there is a known optimal algorithm for both models has the same asymptotic bounds on both models. Therefore, it has been unknown whether the models are equivalent or whether the single-disk model is strictly more powerful.

In this paper we provide evidence that the single-disk model is strictly more powerful. We prove a lower bound on any general simulation of the single-disk model on the multi-disk model and establish randomized and deterministic upper bounds. Let N be the problem size and let T be the number of parallel I/Os required by a program on the single-disk model. Then any simulation of this program on the multi-disk model will require $\Omega\left(T\frac{\log(N/D)}{\log\log(N/D)}\right)$ parallel I/Os. This lower bound holds even if replication is allowed in the multi-disk model. We also show an $O\left(\frac{\log D}{\log\log D}\right)$ randomized upper bound and an $O\left(\log D(\log\log D)^2\right)$ deterministic upper bound. These results exploit an interesting analogy between the disk models and the PRAM and DCM models of parallel computation.

*Department of Computer Science, University of Hartford, 200 Bloomfield Avenue, W. Hartford, CT 06117-1599. Email: armen@hartford.edu. This work was done while the author was a graduate student at Dartmouth College.

1 Introduction

The interest in out-of-core computation in recent years has been a result of increased sequential and parallel processing power outstripping the price/performance yield of primary memory. Since transfer of data to and from disk is orders of magnitude slower than access to primary memory, algorithms that minimize disk access are of increased importance. We consider here two models of parallel disks that have been proposed for the study of these algorithms.

Of these two models, described below, it seems obvious that one is the more powerful. However, every problem for which there is a known optimal algorithm has matching bounds on both models. Are the two models asymptotically equivalent? In order to prove that they are not, we would need to show a lower bound for a problem on one model that is asymptotically greater than an upper bound on the other model. In this paper, we provide evidence of a separation; we prove a lower bound on any *simulation* of the (seemingly) more powerful model by the (seemingly) less powerful one.

The single-disk, D-head model ($\mathcal{SD}$) [1] consists of a finite primary memory, a processing component, and a single disk with D read/write heads. The parameters of the model are

- N, the total number of records in the input;
- M, the size of the RAM, in records;
- D, the number of read/write heads;
- B, the block size, in records.

A single parallel read or write consists of D blocks of B contiguous records each. The processing element is assumed to be arbitrarily powerful, and the measure of complexity is the number of such parallel I/O operations.

The multi-disk model ($\mathcal{MD}$) [10] has the same parameters as the $\mathcal{SD}$ model. The difference is that the N records are distributed across D disks instead of being on a single disk, and the BD records in a parallel I/O must consist of D sets of B contiguous records, one set from each disk. Thus the $\mathcal{MD}$ model is clearly the more restrictive one.

The algorithms that have been studied on these models all have matching upper and lower bounds on both models. For example, the I/O bounds for sorting, FFT, and general permuting are

$$\Theta\left(\frac{N}{BD}\frac{\lg(N/B)}{\lg(M/B)}\right)$$

parallel I/Os. These bounds are tight. But is there a problem for which the lower bound on the $\mathcal{MD}$ model is asymptotically greater than an upper bound established on the $\mathcal{SD}$ model? Or are the two models equally powerful?

The $\mathcal{SD}$ model is at least as powerful as the $\mathcal{MD}$ model, as evidenced by the following simple simulation. We divide the single $\mathcal{SD}$ disk into D regions, each corresponding to a disk of the $\mathcal{MD}$ model. Each read or write in the multi-disk algorithm is simulated by one of the D heads moving to the appropriate block in each region of the single disk.

Our main result is a lower bound on the simulation in the other direction. We give a worst-case lower bound on the number of steps required to simulate an $\mathcal{SD}$ algorithm on the $\mathcal{MD}$ model. This analysis is modeled after that of Upfal and Widgerson [8], which pertains to the EREW PRAM and DCM models of parallel computation. In Section 3 we see that the simulation algorithms for these analogous models also apply to the $\mathcal{SD}$ and $\mathcal{MD}$ models, giving us upper bounds as well.

2 A Lower Bound on the Separation

We show a lower bound on any simulation of an $\mathcal{SD}$ algorithm on the $\mathcal{MD}$ model. The main problem encountered by such a simulation algorithm is that the set of BD records required by a particular $\mathcal{SD}$ instruction might be located on a set S of disks such that $|S| = o(D)$. This clustering of records will require the $\mathcal{MD}$ simulation algorithm to perform many $\mathcal{MD}$ instructions in order to simulate a single $\mathcal{SD}$ instruction. It might help the simulation algorithm to replicate some or all of the original N records. The bound that we show in this section allows for replication of the N records, whether the replication is uniform or not. First we require the following definitions, adapted from [8]:

- A *simulation scheme* consists of
  1. A mapping of records from addresses on the single $\mathcal{SD}$ disk to addresses on the multiple $\mathcal{MD}$ disks. The mapping may be one-to-many, allowing for replication of the original ($\mathcal{SD}$) records on the D disks of the $\mathcal{MD}$ model.
  2. A read protocol. When an $\mathcal{SD}$ instruction calls for a read from disk, the read protocol dictates which copy(ies) of a record are to be read on the $\mathcal{MD}$ model, and in what order.
  3. A write protocol. When an $\mathcal{SD}$ instruction calls for a write to disk, the write protocol dictates which copy(ies) of a record are to be updated (see note in the next item).
- The degree of replication ρ is the average number of up-to-date copies of each record. Note that a particular write protocol may not require updating all copies of a record, but a copy that is not updated no longer contributes to ρ.
- The maximum delay l is the worst–case number of simulation steps required for any read.

Given a simulation scheme S, an algorithm for the $\mathcal{SD}$ model induces a sequence of simulation steps on the $\mathcal{MD}$ model. In order to prove a lower bound on the number of parallel I/O operations required to simulate an algorithm, we will show that l must be large, as shown in the following lemma.

Lemma 2.1 *Let N be the number of records, D the number of read/write heads, and ρ the degree of replication for any simulation scheme from the $\mathcal{SD}$ to the $\mathcal{MD}$ model. Then the simulation scheme must have maximum delay $l \geq (N/D)^{1/2\rho}$.*

Proof. We say that a record is *nonubiquitous* if it has fewer than 2ρ copies. We will count the number of nonubiquitous records in two different ways, generating an expression in l, N, D and ρ which we then solve for l. Because there are on average ρ updated copies of each record, at least half of the records must have fewer than 2ρ copies. Therefore there are at least $N/2$ nonubiquitous records.

Alternatively we can produce an overestimate of the number of nonubiquitous records. First note that there can be no set of D records whose copies are concentrated in a set of fewer than D/l disks. If there were, it would take more than l steps to read the D records, contradicting the definition of the delay l.

There are $\binom{D}{D/l}$ sets of size D/l. By the discussion in the previous paragraph, each of these sets can contain all of the copies of no more than D records, so there can be no more than $D\binom{D}{D/l}$ occurrences of nonubiquitous records in sets of this size. If a record is nonubiquitous then all of its copies must be included in at least $\binom{D-2\rho}{D/l-2\rho}$ sets of this size. To see this, note that this is the number of sets of size $D/l-2\rho$ that definitely do not contain an occurrence of a given record; we can combine each of these sets with the set of at most 2ρ records that might contain an occurrence. We divide to get an overestimate of the number of nonubiquitous records; such an overestimate must be greater than the minimum number of nonubiquitous records. We have

$$\frac{\binom{D}{D/l} D}{\binom{D-2\rho}{D/l-2\rho}} \geq \frac{N}{2},$$

which implies that

$$\frac{D!\,(D/l-2\rho)!}{(D/l)!\,(D-2\rho)!} \geq \frac{N}{2D}.$$

Applying Stirling's approximation, taking logarithms and dropping lower-order terms, we get

$$2\rho \ln l \geq \ln N - \ln D,$$

which implies that

$$l \geq (N/D)^{1/2\rho}.$$

■

Recall that the delay l is the worst-case number of steps required by a simulation scheme for any read. If many of the reads in the $\mathcal{SD}$ algorithm require nearly l reads in the simulation, then the delay l will be related to the overall performance of the simulation.

Theorem 2.2 *Let S be any simulation scheme from the $\mathcal{SD}$ to the $\mathcal{MD}$ model. There exists an algorithm which runs in $T \geq (1+\epsilon)\frac{N}{D}$ parallel I/Os for some fixed $\epsilon > 0$ on the $\mathcal{SD}$ model but for which S requires $\Omega\left(T\frac{\lg(N/D)}{\lg\lg(N/D)}\right)$ I/Os.*

Proof. We construct the program for such an algorithm. Let $B = 1$ and $N > 2D$. We consider a *step* to consist of a *read* portion followed by a *write* portion, and we do not count computation steps. The program will consist of an initialization phase of length $\sigma = N/D$ steps, followed by a work phase of duration $T-\sigma$ steps. Let ρ_t be the degree of replication after the tth step, and let s_t be the number of parallel I/Os required on the $\mathcal{MD}$ to simulate the tth $\mathcal{SD}$ step.

The input contains N records with no replication. The simulation may introduce some degree of replication in the initialization phase. At the end of that phase, the average degree of replication can be no greater than the average number of simulation steps per $\mathcal{SD}$ step:

$$\rho_\sigma \leq \frac{\sum_{1\leq t\leq\sigma} s_t}{\sigma} = \frac{D}{N}\sum_{1\leq t\leq\sigma} s_t \qquad (1)$$

For the remaining $T-\sigma$ steps of the computation, each step will consist of the following:

1. Read D records, including at least one that achieves the maximum delay l.
2. (a) Write to D records that have at least ρ_{t-1} updated copies, if there are are D such records, or

(b) Write to all of the records that have at least ρ_{t-1} updated copies, if there are D or fewer of such records.

We wish to bound the degree of replication ρ_t, for some $t > \sigma$. Our approach is to bound the *increase* in replication from step $t-1$ to step t; that is, we know that ρ_t is ρ_{t-1}, adjusted by some positive or negative quantity that we estimate as follows. Simulating step t takes s_t parallel I/Os. At least $(N/D)^{1/2\rho_{t-1}}$ of these I/Os must be reads, by Lemma 2.1. So there are $s_t - (N/D)^{1/2\rho_{t-1}}$ parallel I/Os remaining, which are available for writes. Of these, we need ρ_{t-1} parallel I/Os to simulate the writes for this step; these are necessary just to maintain the average degree of replication from the previous step. Once these reads and writes are accounted for, any additional steps might add to the replication. Thus by averaging over N records we have

$$\rho_t \leq \rho_{t-1} + \left(s_t - \left(\frac{N}{D}\right)^{1/2\rho_{t-1}} - \rho_{t-1} \right) \frac{D}{N}.$$

Summing over all $t > \sigma$ gives us:

$$\begin{aligned} \sum_{\sigma+1 \leq t \leq T} \rho_t &\leq \sum_{\sigma+1 \leq t \leq T} \rho_{t-1} + \\ &\quad \frac{D}{N} \sum_{\sigma+1 \leq t \leq T} \left(s_t - \left(\frac{N}{D}\right)^{1/2\rho_{t-1}} - \rho_{t-1} \right). \end{aligned}$$

Splitting sums and subtracting, we have

$$\begin{aligned} \rho_T &\leq \rho_\sigma + \frac{D}{N} \sum_{\sigma+1 \leq t \leq T} s_t - \\ &\quad \frac{D}{N} \sum_{\sigma+1 \leq t \leq T} \left(\left(\frac{N}{D}\right)^{1/2\rho_{t-1}} + \rho_{t-1} \right), \end{aligned}$$

and dividing by D/N and rearranging terms gives us

$$\begin{aligned} &\frac{N}{D}\rho_T + \sum_{\sigma+1 \leq t \leq T} \left(\left(\frac{N}{D}\right)^{1/2\rho_{t-1}} + \rho_{t-1} \right) \leq \\ &\qquad \frac{N}{D}\rho_\sigma + \sum_{\sigma+1 \leq t \leq T} s_t. \end{aligned} \tag{2}$$

We want to know the total simulation time, $\sum_{1 \leq t \leq T} s_t$. Using (1) and then (2),

$$\begin{aligned} \sum_{1 \leq t \leq T} s_t &= \sum_{1 \leq t \leq \sigma} s_t + \sum_{\sigma+1 \leq t \leq T} s_t \\ &\geq \frac{N}{D}\rho_\sigma + \sum_{\sigma+1 \leq t \leq T} s_t \\ &\geq \frac{N}{D}\rho_T + \sum_{\sigma+1 \leq t \leq T} \left(\left(\frac{N}{D}\right)^{1/2\rho_{t-1}} + \rho_{t-1} \right) \\ &\geq 2\rho_T + \sum_{\sigma+1 \leq t \leq T} \left(\left(\frac{N}{D}\right)^{1/2\rho_{t-1}} + \rho_{t-1} \right). \end{aligned}$$

The last inequality is true because $N > 2D$. Let $\tilde{\rho}$ be the average degree of replication in the last $T - \sigma$ steps. Then we know

$$\begin{aligned} \sum_{1 \leq t \leq T} s_t &\geq 2\rho_T + \sum_{\sigma+1 \leq t \leq T} \left(\left(\frac{N}{D}\right)^{1/2\rho_{t-1}} + \rho_{t-1} \right) \\ &\geq \tilde{\rho}(T-\sigma) + \sum_{\sigma+1 \leq t \leq T} \left(\frac{N}{D}\right)^{1/2\rho_{t-1}}. \end{aligned} \tag{3}$$

Note that the summand in the summation that remains is a convex function in ρ. By Jensen's inequality (see, e.g., [7]),

$$\sum_{\sigma+1 \leq t \leq T} \left(\frac{N}{D}\right)^{1/2\rho_{t-1}} \geq (T-\sigma) \left(\frac{N}{D}\right)^{1/2\tilde{\rho}}. \tag{4}$$

Finally, putting together (3) and (4), we have

$$\sum_{1 \leq t \leq T} s_t \geq (T-\sigma) \left(\left(\frac{N}{D}\right)^{1/2\tilde{\rho}} + \tilde{\rho} \right).$$

We note that this expression is minimized when $\tilde{\rho} = D$; recalling also that $T \geq (1+\epsilon)\frac{N}{D}$, we have

$$\sum_{1 \leq t \leq T} s_t = \Omega\left(T \frac{\lg(N/D)}{\lg\lg(N/D)} \right).$$

■

3 Upper Bounds on the Simulation

We present randomized and deterministic upper bounds for the simulation of an $\mathcal{SD}$ algorithm on the $\mathcal{MD}$ model. We show that simulation schemes for an analogous pair of models of parallel computation can be used to derive simulation schemes for the $\mathcal{SD}$ and $\mathcal{MD}$ models. These models of parallel computation are well-studied and the reader is referred to [4, 5] for additional details.

- *EREW PRAM* (Exclusive-Read, Exclusive-Write Parallel RAM). This machine consists of a global shared memory and p processors. The processors have a bounded set of registers and are capable of arithmetic and other operations typical of a RAM. In a given step, a processor may execute a global

read, a global write, or compute a value from arguments in its registers. No more than one procesor may access a given memory location in a step.

- *Direct Connection Machine (DCM).* This machine has p processors, and its memory is divided into p banks. In a given step, no more than one processor may access a given memory bank. (Variations on this model can also be found in [6, 8, 9].)

Theorem 3.1 *Any simulation of an* EREW PRAM *by a* DCM *can be used to construct a simulation of an $\mathcal{SD}$ by an $\mathcal{MD}$.*

Proof. Suppose there is a scheme S for simulating an EREW PRAM on a DCM, and we're given an $\mathcal{SD}$ program P to simulate on the $\mathcal{MD}$ model. Setting $p = D$, we use the mapping given by S as the mapping from $\mathcal{SD}$ block addresses to $\mathcal{MD}$ block addresses. We then simulate each read or write of P by reading or writing to the address(es) called for by the protocols of S. Because the mapping for S must guarantee that no two processors access the same memory bank, it will also guarantee that each block read or written in a parallel I/O will be from a different disk. Both $\mathcal{SD}$ and $\mathcal{MD}$ processors are assumed to be arbitrarily powerful, so computation steps are not critical. ■

The following results now apply:

- Melhorn and Vishkin [6] use universal hash functions to construct a simulation with a slowndown of only $O(\frac{\log D}{\log \log D})$. No replication is required. $\mathcal{SD}$ "logical" addresses are mapped to $\mathcal{MD}$ "physical addresses".

- Upfal and Widerson [8] give an $O(\log D(\log \log D)^2)$ deterministic simulation scheme. This scheme involves replication by a factor of $O(\log D)$ and distribution of the copies according to a randomly selected bipartite graph. (As in [2], this graph is chosen only once, when the system is constructed.) I/Os are accomplished by majority protocols.

4 Discussion

We have produced evidence of the separation of the $\mathcal{SD}$ and $\mathcal{MD}$ models, thus providing a first answer to the question of whether the two models are equally powerful. As noted in Section 1, however, all problems for which there are known optimal algorithms have the same asymptotic performance on either model. It would be more satisfying to discover a natural problem or class of problems for which we can show a lower bound on the $\mathcal{MD}$ model that is asymptotically greater than an upper bound for the same problem on the $\mathcal{SD}$ model. One candidate for such a problem is multiway merge, in particular, perhaps, D-way merge. This problem is attractive for this purpose because we may consider all permutations of the records in attempting to produce a lower bound for the $\mathcal{MD}$ model. This open question, the lack of a separating problem, applies to the analogous PRAM and DCM models as well.

Acknowledgments

I thank Tom Cormen [3] for many helpful discussions and for raising this question in his 1993 graduate seminar on parallel I/O at Dartmouth. He in turn credits Tom Leighton for raising the question at MIT. I also thank my advisor, Cliff Stein, for his guidance and insights, and the anonymous reviewers for their helpful comments.

References

[1] Alok Aggarwal and Jeffrey Scott Vitter. The input/output complexity of sorting and related problems. *Communications of the ACM*, 31(9):1116–1126, 1988.

[2] M. Ajtai, J. Komlos, and E. Szemeredi. An o(log n) sorting network. In *Proceedings of the 15th Annual ACM Symposium on Theory of Computing*, pages 1–9, April 1983.

[3] Thomas H. Cormen. *Virtual memory for data-parallel computing.* PhD thesis, M.I.T., January 1993.

[4] Richard M. Karp and Vijaya Ramachandran. A survey of parallel algorithms for shared-memory machines. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, pages 945–971. Elsevier Science Publishers B.V, 1990.

[5] Clyde P. Kruskal, Larry Rudolph, and Marc Snir. A complexity theory of efficient parallel algorithms. In *Theoretical Computer Science 71*, pages 95–132. Elsevier Science Publishers B.V, 1990.

[6] Kurt Mehlhorn and Uzi Vishkin. Randomized and deterministic simulations of prams by parallel machines with restricted granularity of parallel memories. *Acta Informatica*, 21(4):339–374, 1984.

[7] H.L. Royden. *Real Analysis*. MacMillian, second edition, 1968.

[8] Eli Upfal and Avi Widgerson. How to share memory in a distributed system. *Journal of the ACM*, 34(1):116–127, 1987.

[9] L.G. Valiant. General purpose parallel architectures. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, pages 945–971. Elsevier Science Publishers B.V, 1990.

[10] Jeffrey Scott Vitter and Elizabeth A.M. Shriver. Algorithms for parallel memory: Two-level memories. *Algorithmica*, 12(2/3):110–147, 1994.