A Case for Intelligent Disks (IDISKs)

Kimberly Keeton, David A. Patterson and Joseph M. Hellerstein

Computer Science Division
University of California at Berkeley
387 Soda Hall #1776
Berkeley, CA 94720-1776
{kkeeton,patterson,jmh}@cs.berkeley.edu

Abstract: Decision support systems (DSS) and data warehousing workloads comprise an increasing fraction of the database market today. I/O capacity and associated processing requirements for DSS workloads are increasing at a rapid rate, doubling roughly every nine to twelve months [38]. In response to this increasing storage and computational demand, we present a computer architecture for decision support database servers that utilizes "intelligent" disks (IDISKs). IDISKs utilize low-cost embedded general-purpose processing, main memory, and high-speed serial communication links on each disk. IDISKs are connected to each other via these serial links and high-speed crossbar switches, overcoming the I/O bus bottleneck of conventional systems. By off-loading computation from expensive desktop processors, IDISK systems may improve cost-performance. More importantly, the IDISK architecture allows the processing of the system to scale with increasing storage demand.

1 Introduction

Microsoft Research’s Dr. Philip Bernstein estimates that decision support systems (DSS) account for about 35% of database servers [10]. The size and computational requirements of these DSS systems are growing at a rapid rate [38] [56] due to numerous factors:

1. More detailed information being saved in each record, such as all of the items in a shopping cart at a retail store;
2. Companies expanding the length of the history they examine (e.g., three years versus six months) to make better decisions;
3. More people wanting access to the historical systems, increasing the number of queries to the DSS.

In addition, when mergers occur, the decision support system of one company must quickly grow to accommodate the historical record of the other; i.e., mergers lead to fewer, larger decision support systems with a larger number of users.

According to Dr. Greg Papadopoulos, chief technical officer at Sun Microsystems Computer Company, the demand for decision support doubles every 9 to 12 months, making its growth faster than both the growth rate of disk capacity (2X in 18 months) and the growth rate of processor performance (2X in 18 months) [38]. This suggests a demand for server designs that can scale processor performance and the number of disks far beyond the evolutionary path of today’s server systems.

Results from the 1997 and 1998 Winter Very Large Database (VLDB) surveys illustrate these phenomenal growth trends [57] [56]. The Sears, Roebuck, and Co. DSS database, the largest Unix-based DSS database reported in 1998, grew more than 350% in the last year from 1.3 TB to 4.6 TB. As part of this growth, Sears added 550% more rows, for a total of 33 billion in 1998. Wal-Mart’s DSS database, the largest Unix-based DSS database reported in 1997, nearly doubled in size from 2.4 TB to 4.4 TB in 1998. The number of rows increased 150% from 20 billion to 50 billion. Unfortunately, it is not clear that server designs will scale accordingly [56]: “Wal-Mart says that a major obstacle to its VLDB plans is that hardware vendors can barely keep up with its growth!”

Shared-nothing clusters comprised of workstations, PCs or symmetric multiprocessors (SMPs) are commonly used as decision support server architectures. The chief strengths of clusters are their incremental scalability and the high performance afforded by developments in parallel shared-nothing database algorithms. However, shared-nothing clusters have several weaknesses, including the I/O bus bottleneck of their nodes, the challenge of distributed system administration, packaging inefficiencies, and the unsuitability of desktop microprocessors for database applications.

This paper presents a vision of an alternative approach to scalable decision support: replacing the standard nodes in a cluster server with “intelligent” disks (“IDISKs”), where each IDISK is a hard disk containing an embedded processor, tens to hundreds of megabytes of memory, and gigabit per second network links. Links from each disk are connected via switches so that IDISKs can simultaneously communicate with each other at full link bandwidth.

An IDISK-based architecture offers several potential price and performance advantages over traditional server architectures. By off-loading computation from expensive central desktop processors to inexpensive embedded disk processors and by leveraging the cabinetry, cooling and power supplies already needed for the disks, IDISK systems may cost much less than conventional systems. IDISK pushes computation closer to the data, thus providing a potential reduction in data movement through the I/O subsystem. The IDISK architecture allows the processing of the system to scale with increasing storage demand. Finally, by using a switch-based interconnect between IDISKs, this architecture provides a truly scalable I/O subsystem, overcoming the I/O bus bottleneck of conventional bus-based systems.

This architecture is motivated by trends in several computer architecture fields. First, hard disks are being designed with an increasing amount of general purpose processing and memory on-board. Second, embedded processors possess increasing computational capability, and are available at a fraction of the cost and power dissipation of desktop processors. Third, switch-based interconnects and advances in high-speed serial communication mean that high bandwidth does not require physical proximity: processors 10 meters apart can communicate as quickly as those communicating over a bus.

Intelligent disk processing for database workloads is not a new idea. Database machines, an active area of research in the 1970s and early 1980s, typically included a central host processor plus intelligent disk processing in the form of a processor per head, track or disk or a multiprocessor disk cache arrangement. These systems fell out of favor in the late 1980s and 1990s because their modest performance did not justify the high cost of special-purpose hardware. However, the increasing demands of DSS workloads, the commoditization of more intelligent disk subsystems, processor and communication technology trends, and advances in database algorithms suggest that it is now time to re-examine intelligent disk processing.

This paper explores the IDISK vision, rather than the performance of a real system. To set the stage for IDISK we first review the strengths and weaknesses of today's servers in Section 2. Section 3 presents the technological trends that lead to the IDISK hardware and software architecture described in Section 4. Since IDISK is reminiscent of past database machines, Section 5 then reviews the successes and failures of these machines and suggests which might apply to IDISK. Section 6 presents the open issues in IDISK, followed by a brief conclusion.

2 Strengths and Weakness of DSS Clusters

Large-scale decision support systems typically employ shared-nothing clusters of workstations, PCs or SMPs. While this hardware architecture possesses strengths that facilitate the development and execution of database software, the architecture also presents several weaknesses. This section discusses these architectural strengths and weaknesses.

The main advantage of cluster organizations is their incremental scalability: machines can easily be added or subtracted as needed by the application. Another advantage of clusters is the high performance afforded by developments in parallel shared-nothing database algorithms. One such example is Berkeley's NOWSort, a shared-nothing sort implemented on a cluster of UltraSparc workstations, which currently holds the MinuteSort record for sorting 8.4 GB in a minute [7]. A final strength is that clusters today leverage the hardware development investment in the desktop, thus using processors, memory, and disks without paying for separate development costs.

We see four weaknesses in cluster architectures:

1. The I/O bus bottleneck
2. System administration challenges
3. Packaging and cost difficulties
4. Inefficiency of desktop microprocessors for database applications

We explore these weaknesses further below.

The first weakness of shared-nothing clusters is I/O busses. Clusters often utilize commodity desktop machines such as workstations or PCs as building blocks, which employ standard processor I/O busses, such as S-bus or PCI. Unfortunately, the performance of these I/O bus standards is limited by the rate of committee decision-making, rather than the rate of technological improvements. As a result, the increase in bus bandwidth is typically much slower than the increase in bandwidth provided by high-performance disk and network peripherals. For example, transfer rates of disks improve at 40% per year, suggesting about a 40 MB/sec transfer rate in 2000. It is not clear that I/O bus performance

Processor	**Digital SA-110**	**MIPS R5000**	**MIPS R5400 (NEC VR 5464)**	**MIPS R10000**	**Intel Pentium II**
Class	Embedded	Embedded	Embedded	Desktop	Desktop
Clock rate	233 MHz	200 MHz	250 MHz	200 MHz	300 MHz
Cache size (I / D)	16 K / 16 K	32 K / 32 K / 512 K	32 K / 32 K	32 K / 32 K / 4 M	16 K / 16 K / 512 K
IC process	0.35 µm 3 M	0.35 µm 3 M	0.25 µm 3 M	0.35 µm 4 M	0.28 µm 4 M
Die size	50 mm^2	84 mm^2	47 mm^2	298 mm^2	203 mm^2
SPEC95 base (int/fp)	n/a	4.7/4.7	10/4.5 (est)	10.7/17.4	11.6/6.8
Dhrystone	268 MIPS	260 MIPS	519 MIPS	203 MIPS	500 MIPS (est)
Power	0.36 W	10 W	4.4 W	30 W	30 W
Est. mfr's cost	$18	$25	$25	$160	$90
Price	$49	$325	$95	$3000	$1981

TABLE 1. 1997-98 Comparison of Embedded and Desktop Processors. [34] [35] [52]

will keep up.

The limitations of current I/O busses become even more evident when we consider that both disk and network peripherals must simultaneously use an I/O bus to transfer data. The I/O subsystem of some workstations is quickly saturated by just a few high-performance disks and network communication for cluster-based database applications [6].

The second weakness of clusters is the high cost of cluster system administration. Managing the distributed resources of a cluster typically involves performing diagnostic and maintenance operations for each individual node in the cluster. Unfortunately, few tools exist to automate these diagnostic and maintenance tasks at the database level [20] and at the operating system level [42]. As a result, cluster system administration typically requires human supervision, which drives up the associated costs [5]. The annual personnel costs can be three times the cost of the disks.

The third cluster weakness is that cluster packaging is often inefficient and expensive. Because clusters rely on more cost-effective commodity building blocks, they must deal with the tower or "pizza box" packaging that houses the processor, memory, and I/O peripherals of these commodity machines. When assembling a large number of such nodes, these packaging configurations occupy significant space and are not nearly as efficient as rack-mountable systems. Commodity desktop processors also have high power requirements, which increases the cost of cooling and power supplies.

A final weakness of clusters is the ineffectiveness of desktop processors for database workloads. The design of a desktop microprocessor requires significant design effort and capital outlay, so a company needs the volume sales of the desktop to justify the expense. Unfortunately, architectural innovations that aid floating point and desktop integer programs, such as multilevel caches, multiple instruction issues per cycle, branch prediction and out-of-order execution [23], are generally less helpful for databases [26] [15] [41] [31]. For example, despite a theoretical peak of 3.0 instructions executed per clock cycle, the Pentium Pro executes on average only 0.74 instructions per clock cycle for the scan and aggregate operations defined in TPC-D query 1 [25]. Transaction processing performance is even worse: the Pentium Pro executes on average only 0.29 instructions per clock cycle for a TPC-C-like workload [26].

The complexity of desktop processor innovations results in increased die size, power requirements, cost, and price, yet yields little improvement in integer performance over much simpler embedded processor designs. Table 1 shows the characteristics of desktop microprocessors versus several embedded microprocessors. Larger die size, power, cost and price of desktop chips may be justified for SPECfp95, but using SPECint95 or Dhrystone we see a much smaller difference. Furthermore, although desktop performance generally increases at roughly 60% per year, experts estimate that desktop performance for transaction processing workloads stays constant at roughly 70 to 100 MIPS [19]. Thus we see cost/power/price differences of factors of 5 to 20 yielding integer performance differences of 1.5X to 2X. While it makes economic sense to use desktop processors in clusters, we see that they are not well-suited to database workloads.

Figure 1 shows the configuration of a high-end cluster-based decision support server used for the 1 TB TPC-D benchmark [51]. The NCR WorldMark cluster is fully configured with 32 nodes, where each node contains a four-processor PentiumPro SMP, 1 GB of main memory and a disk array containing 41 disks [36]. NCR adds value by repackaging these commodity parts more efficiently, and including their proprietary interconnect, the BYNET switched network. This server contains a hierarchy of busses and networks to connect disks to processors, which not only add cost to system, but also serve as potential bottlenecks.

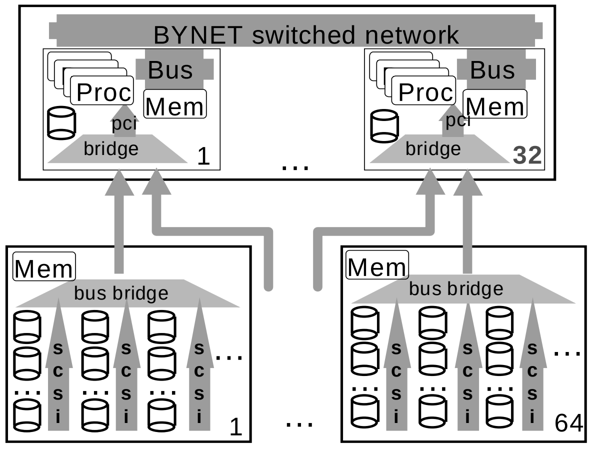

FIGURE 1. Typical High-End Decision Support Server Computer Architecture: NCR WorldMark. This cluster has 32 nodes, each with four processors and 1 GB memory and 41 disks. This system uses a total of 128 processors, 32 GB of memory, and 1312 disks for a 1 TB TPC-D configuration [36].

Table 2 shows the hardware prices for this WorldMark configuration. The repackaging and proprietary interconnect provided by NCR prove to be quite costly. Note that the processors, main memory, boards and enclosures cost over $5M. The disk I/O subsystem, including the disks, controllers, and disk enclosures, costs nearly $3M.

CPUs, DRAM, enclosures, boards, power.	$5,360k
Disks, controllers	$2,164k
Disk enclosures	$674k
Cables	$126k
Console	$16k
Hardware total	$8,340k

TABLE 2. NCR WorldMark Price Breakdown.

Given these characteristics of clusters, performance analysis experts suggest that configuring a system for decision support databases typically involves:

1. Putting the maximum number of processors in a box to amortize the cost of the enclosures;
2. Putting the maximum amount of memory into the box to get maximum memory bandwidth and to avoid going to disk, thereby avoiding I/O bus bottlenecks;
3. Attaching as many disks as needed for capacity and trying to spread data over multiple disks to quickly load information into memory.

Decision support databases on architectures configured to follow this advice are normally CPU bound, and the disks are typically not very busy [21].

We found this observation surprising: we thought that the data-intensive nature of decision support would translate into an I/O-bound system, rather than a CPU-bound system. Given that a significant fraction of the cost of such a configuration is the disk subsystem, we should be able to construct a system with sufficient processing and communication bandwidth so that the performance limit is also the disks.

3 Technological Trends

We see five trends that make IDISK viable:

1. Commodity disks include embedded processors and memory.
2. Embedded CPUs have much lower costs, with integer performance within 2X that of desktop CPUs.
3. Disks are beginning to include high speed serial links.
4. High speed links and switches are economical.
5. Integrated devices combining embedded processors, memories, and serial lines are being developed that match the requirements of disk manufacturers.

In this section, we explore these trends further. For a more detailed discussion of processor, memory, network, and disk trends, see [39].

The first trend is increased disk-resident processing and memory. Moore’s Law says the number of transistors per chip will double every 1.5 years and transistor speed is improving at about 1.3X per year. In light of these advances, disk manufacturers are increasingly using embedded processors to perform tasks previously done in special-purpose hardware. Using software for features such as servo control, error correction calculation, and SCSI command processing gives them flexibility and reuse in disk development. Power budgets for such processors inside the disk assembly are on the order of 2 watts [32]. The low power consumption of embedded processors allows the combined disk and processor to remain well within the allotted power budget.

In addition to on-disk processing, disk manufacturers are including more and more memory on disk. For instance, the Seagate Cheetah hard drive includes 1 MB of track buffer memory. Seagate also offers the option of increasing the track buffer memory to 4 MB, for an additional fee [44]. By the year 2001, experts at Seagate estimate that a high-end disk drive will have 100 MIPS to 200 MIPS in processing, plus up to 64 MB of memory provided by one or two high-density memory chips [4].

A second trend that supports IDISK is the cost-performance of embedded processors versus desktop processors. As mentioned above, given the highly competitive desktop environment, designers are willing to make trade-offs that raise the costs by factors of 5 to 10, while improving integer performance by less than factors of 2. Embedded designs provide integer performance within 2X that of desktop designs, at a fraction of the cost and power.

A third technology trend is that disk manufacturers are moving away from busses to fast serial lines, such as Fibre Channel Arbitrated Loop (FC-AL) and Serial Storage Architectures (SSA). In addition, Intel recently announced their plans to use a Gbit/sec serial bus as a follow-on to 64-bit, 66 MHz PCI busses [13]. The inclusion of fast serial line technology and serial cables in mainstream disks makes IDISK more plausible. Gigabit per second serial lines imply that a few pins can supply substantial bandwidth. A state-of-the-art serial line today operates at 4 to 5 Gbit/sec [16] [58]. To put these bandwidths into perspective, today's peak PCI bus bandwidth (32-bit wide, 33 MHz) is 1.1 Gbit per second. Hence, moving from busses to point-to-point connections can offer tremendous bandwidth increases as well as pin efficiency.

Serial lines provide a unique opportunity for IDISK that is not mirrored for conventional cluster building blocks. The problem is that workstations, PCs, and SMPs are not designed to have anything on their proprietary memory interconnect but a fixed number of processors, memory modules, and bridges to standard I/O busses. These devices must be few in number and carefully designed to match the speed demands, to maintain the symmetry of uniform latency to memory or I/O devices. As long as these commodity building blocks are constructed from many chips connected over short distances via proprietary interfaces, optional peripherals, such as the network, will likely be connected via the I/O bus.

Switch designers are also taking advantage of the transistor density and speed growth trends of Moore's Law to build switches on a single chip. One example of a state-of-the-art nonblocking, fully connected switch comes from AMCC [3]. This single-chip, 32 x 32 port switch has links that operate at 1.5 Gbit/sec. Since it is a single chip, the price is about $400 [3]. Such switches can be cascaded together in various topologies to provide the desired bandwidth and connectivity [14].

Fitting sufficient logic into the constrained space and power budgets of commodity disks has proven to be a challenge for disk manufacturers. Fortunately, there are products and research projects looking at integrating several components into a single chip--processor, DRAM, and even networks--thereby reducing some of these latencies and vastly increasing the bandwidth [40] [46]. One example project is Berkeley's IRAM, or "intelligent RAM," project [40]. Such integration reduces memory latency by factors of 5 to 10 and increases memory bandwidth by factors of 25 to 50 [40]. This integration is also important to meet the power and size restrictions of the disk assembly, since integration of several chips reduces size and also reduces power by avoiding the driving of the pins to send signals off chip.

4 The Intelligent Disk Architecture

Given the background of the prior sections, we can now describe the hardware parameters and software model for the IDISK architecture.

4.1 Hardware Architecture

Figure 2 shows the IDISK architecture. The standard nodes in a conventional shared-nothing cluster decision support database server are replaced with IDISKs. Each IDISK has an embedded processor, memory, and gigabit per second serial links. IDISKs communicate with each other using these serial lines, which are connected via high-speed non-blocking crossbar switches.

The goal of IDISK is to leverage the increasing integration of processing, memory, and high-speed communication on the actual disk. The associated physical restrictions impact the speed of the processor and the size of memory. An IDISK processor will have about half the performance of a central desktop processor, and its memory might be limited to about 32 MB to 64 MB in the year 2000 time frame. Thus with hundreds of disks there are gigabytes of memory, but the memory is distributed. Each node could have up to eight 2 gigabit/

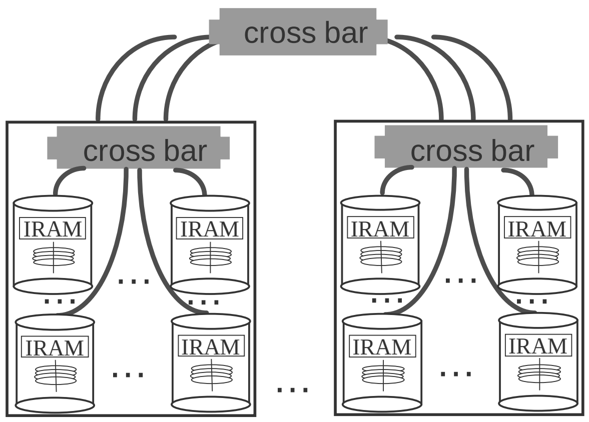

FIGURE 2. IDISK Architecture.

sec links. Depending on the bandwidth desired, such links can be connected in topologies that range from point-to-point rings to fully interconnected switches, with many possible compromises between these two extremes. Applications requiring low bandwidth could use some of these serial lines for redundancy.

Since the disk must already have a power supply, enclosure, and cabling to the outside world, an IDISK system could be made available at little extra cost over a conventional disk subsystem. Even including the extra cost of the crossbar switches, it is possible to imagine the extra cost of an IDISK system to be on the order of 10% of the cost of a traditional disk subsystem.

While the IDISK architecture is similar in some ways to a traditional cluster, IDISK also exhibits several key differences. First, the memory capacity on a single IDISK is limited, due to the strict power and area budget of the disk. IDISK memory capacity may be as much as 5X to 10X smaller than the capacity of a cluster workstation, PC, or SMP. This constraint presents research challenges in designing memory-conscious algorithms, as described in Section 6.

Second, IDISK's ratios between computation power, memory capacity, disk bandwidth, and network bandwidth are more fixed than those of the traditional cluster building block. IDISK's one-to-one mapping between disks and processors and memory capacity limits present fewer node configuration options than the options provided by traditional cluster building blocks, which have a lower degree of system integration.

While the IDISK architecture presents several research challenges, it also solves several of the shortcomings of traditional cluster architectures. First, by integrating the processor with the disk, the issue of limited I/O bus bandwidth is sidestepped, provided that the processor's I/O interface can keep up with the bandwidth requirements of a single disk and the high-speed network. By using a switch-based interconnect between IDISKs, this architecture provides a truly scalable I/O subsystem.

Second, IDISK systems may cost less than conventional cluster systems, for several reasons. One cost reduction comes from off-loading computation from expensive desktop processors to less expensive embedded disk processors. Since computation and communication are directly integrated into the disk, the bulky packaging of conventional cluster building blocks can be eliminated. An IDISK system can then be built in roughly the same floor space and with the same amount of cabinetry as a conventional disk subsystem. Additionally, an IDISK architecture will require only marginally greater power and cooling budgets than conventional disk systems, which are far less than the power-hungry behavior of central processing in conventional cluster systems.

Finally, by using one or more of the IDISK processors as an interface processor, the IDISK cluster can be configured to look like a traditional disk array externally. This configuration allows it to be administered with the same ease as a traditional centralized SMP-based system, rather than the difficulty of a distributed cluster.

In addition to solving cluster challenges, IDISK also provides several unique opportunities. First, through the one-to-one coupling of processors and disks, the IDISK architecture allows the processing of the system to scale with increasing storage demand. Since each IDISK also contains a high-speed network interface, the communication bandwidth of the system grows, as well. Adding another IDISK to the system automatically adds processing and communication bandwidth to help address the explosive growth in decision support data capacity and associated processing and communication requirements.

Second, by locating processing closer to the disk, IDISK gives the opportunity for closer coupling between application software and on-disk scheduling and physical resource management. For instance, by understanding the physical layout of data and the location of the disk head, an IDISK processor could reorder a series of IDISK requests to minimize the number of costly random seek operations, thus improving disk performance.

Perhaps the easiest way to think about the IDISK architecture shown in Figure 2 is as the next step in the life cycle of clusters for data-intensive applications. Economic and packaging arguments suggest that IDISK will eventually be the cluster building block for these applications. However, until a single IDISK possesses enough computation and memory capacity to run arbitrary database applications, we envision a more evolutionary design. Figure 3 illustrates this evolutionary IDISK architecture. Although IDISK trades inexpensive IDISK processing for expensive central processing, the evolutionary design retains centralized processing to simplify the programming model, accept and optimize user queries, and assist in executing queries too complex for IDISK alone. To reduce price, evolutionary IDISK servers may incorporate inexpensive lower-end PCs or SMPs for the centralized resources, with few CPUs and less memory.

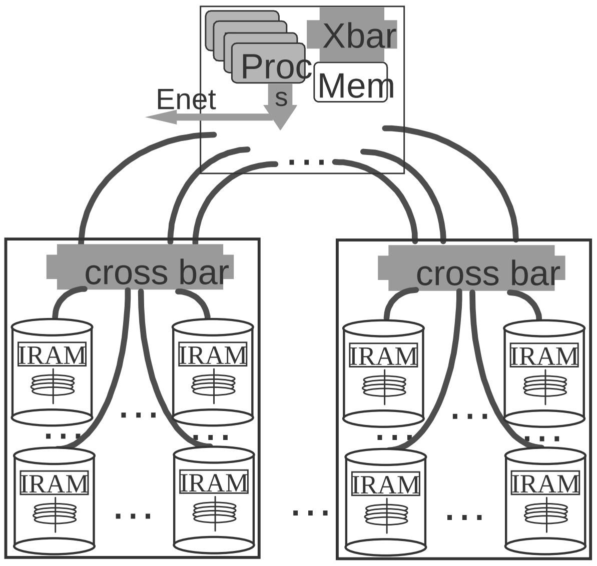

FIGURE 3. Evolutionary IDISK Architecture.

4.2 Software Architecture

The goal of IDISK is to move data-intensive processing closer to the data. We see four main software architecture options, where each option seems well-suited for a particular IDISK architecture:

1. Run a complete shared-nothing database server and operating system on each disk processor.

   Given the commercial and research experience with shared-nothing databases, this well-understood model seems suited to the IDISK architecture shown in Figure 2. The disadvantage is that it takes up a non-trivial amount of the precious DRAM inside each disk to contain copies of whole operating systems and database code. At 32 MB to 64 MB per IDISK in 2000, this could sacrifice performance. However, as computational and memory capacity per IDISK grow, this approach seems the most desirable for the cluster-based IDISK architecture.

2. Run only a library of functions specified by the disk manufacturer on the disk processor.

   In this alternative, the database would be executed entirely on the centralized front end node of an evolutionary IDISK design, as shown in Figure 3. A small library would contain the functions needed to turn record accesses into physical disk addresses. This library might also perform low-level optimizations, such as scheduling of accesses to minimize costly seek operations. The chief advantage of this approach is that it would entail minimal changes to database software. However, this alternative provides only a small evolutionary step beyond the interface that disks offer today; it is not clear that this approach would take advantage of the available on-disk processing capabilities.

3. Run all of the database storage/data manager and a reduced operating system on each disk node.

   Similarly to the previous alternative, the remaining database functionality would be executed on a centralized front end node, as shown in Figure 3. This approach would reduce the memory costs, simplifying both the database code and hopefully the operating system to be run on the IDISK. An advantage of this approach is that the interface between the storage/data manager and the rest of the database is reasonably clearly defined. However, this tuple-based interface may be too low-level to allow considerable functionality to be implemented on-disk.

4. Run a reduced operating system and relational operators, such as scan, sort, and join, on the disk processor.

   As in the previous two alternatives, the remaining database functionality would be executed on the front end. This approach would keep memory costs low, yet allow considerable data-intensive functionality to be implemented on disk. The research challenge to this approach is to define a general-purpose interface to allow primitive operations to be downloaded to and executed on the IDISK processor. From our perspective, this option is the most desirable for the evolutionary IDISK architecture.

Initial reactions to our proposal have been mixed, with some industrial database experts suggesting that IDISK can succeed only if it requires minimal changes to existing database software. This outlook inverts the historic roles of hardware and software, implying that hardware is now flexible and easy to change, while software is now brittle and hard to change.

We think it is unwise to accept such restrictions, for several reasons. First, researchers *should* explore a wider space than what industry considers practical immediately if we are to have a foundation of ideas for future systems. Second, the cost of software maintenance and innovations such as Java may lead companies to re-engineer their software just to stay in business, thereby simplifying such changes. Finally, if it is true that legacy software prevents database companies from innovating, then such inertia will create a market opportunity for new companies with an innovative idea and no software legacy.

5 Historical Perspective and Related Work

The concept of putting processing closer to the disk is not a new one. In the late 1970s and early 1980s the field of hardware database machines was an active area of research [24] [18]. The disk processing in the database machines fell into roughly four categories: processor per head (e.g., OSU's DBC [9], SURE [27]), processor per track (e.g., CASSM [50], RAP [37], RARES [29], and CAFS [8]), processor per disk (e.g., SURE [27]) and multi-processor cache (e.g., RAP.2 [45], DIRECT [17], INFOPLEX [30], RDBM [22], and DBMAC [33]). In each architecture, a central processor(s) acted as the front end of the system. For the most part, these machines were exceptionally good at pushing simple processing, such as scan operations, closer to the disk, achieving great performance improvements.

Database machines experienced many pitfalls, however, which eventually led to their demise [12]. First and foremost, most database machines used non-commodity hardware, such as associative disks, associative CCD devices, and magnetic bubble memory. Unfortunately, the performance gains weren't great enough to justify the additional cost of the special-purpose hardware. Second, while scan performance was impressive, performance gains were not as forthcoming for more complex operations, such as joins and sorts. Again, such narrowly defined speedup often did not justify the extra cost. Third, the "brute" force solution provided by database machines was surpassed by the performance given by smarter algorithms, such as new indexing techniques. Fourth, communication performance between the processing elements was insufficient: bandwidth was not high, and message overheads were quite large. Fifth, improvements in disk transfer rates didn't keep pace with processing element speed improvements, resulting in low utilization for disk processors. Finally, authors of legacy commercial database code didn't rewrite their applications to take advantage of the new hardware advances.

While initially the IDISK concept seems similar to the database machines of old, there are, in fact, a number of reasons why the same criticisms do not apply today. Most importantly, disk manufacturers are including general-purpose embedded processing and increased memory on disk. If such processing can provide great performance gains for important applications like databases, it could lead to even greater processing and memory in commodity disks. In addition, numerous algorithmic advances have been made over the last 15 to 20 years, including shared-nothing sort and join algorithms and multi-pass algorithms that trade off I/O bandwidth for memory capacity. IDISK can leverage these developments to provide high performance, even for more complex operations.

From a technological standpoint, advances in serial line communication will provide high-bandwidth communication between disk processors. With serial standardization efforts, such as Fibre Channel Arbitrated Loop (FC-AL), these interfaces will be available on commodity disk drives.

Research in disk-resident processing has experienced a resurgence in the 1990s [11]. Two other academic research projects, the "active" disk projects at Carnegie Mellon and UC Santa Barbara/Maryland, are examining the advantages of downloading application-specific code to more intelligent disks for database, data mining and multimedia applications [1] [2] [43]. Figure 4 shows the main differences between these two projects and IDISK, with respect to application complexity and hardware complexity.

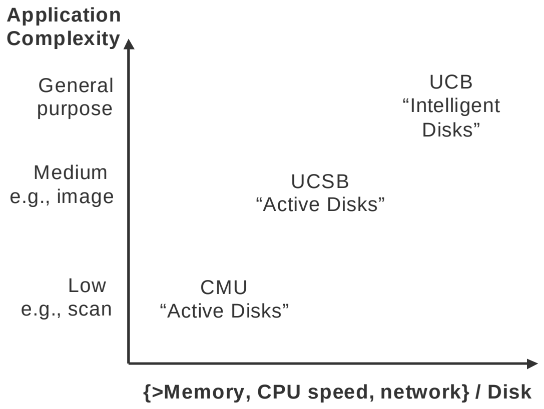

FIGURE 4. Comparison of Ongoing Intelligent Disk Research Projects.

CMU's active disk project has focused on scan-based algorithms for nearest neighbor searches, frequent sets, and image edge detection [43]. These algorithms require relatively weak on-disk processing, low on-disk memory capacity and no communication between disks. The Santa Barbara/Maryland active disk group has focused on similar applications, including database select, external sort, datacube operations, and image processing [1] [2]. Their architecture assumes more powerful on-disk processing, higher on-disk memory capacity, and restricted disk-to-disk communication through a central front end. In contrast, the IDISK proposal described in this paper permits much higher bandwidth disk-to-disk communication, permitting more general-purpose parallel computation.

Ongoing IDISK research is not limited to database applications. One example IDISK file system research issue is reducing write latency by having writes occur anywhere within a cylinder, leaving the decision to the IDISK processor and informing the file system later [53]. Earlier work in the DataMesh project by Wilkes, et al., foreshadowed the IDISK architecture shown in Figure 2, and presented a file system to exploit the DataMesh architecture [54] [55].

In addition to file service, we can also imagine IDISKs offering an advantage for many other application areas:

- software RAID.
- automatic reconfiguration: IDISKs could automatically balance the load between disks, and provide support for “plug and play,” where a new IDISK is automatically recognized and incorporated into the system.
- backup acceleration: IDISKs could compress data as it is written to tertiary storage across the high-speed interconnect.
- multimedia service: IDISKs could perform image manipulations or on-the-fly video transcoding.
- web service.

6 IDISK Challenges and Research Areas

Given that this paper presents a new vision, not surprisingly it leaves many open questions. Below are several questions that we are currently exploring:

- What is the software model for IDISK; i.e., in an evolutionary IDISK architecture, how should application software be partitioned between the central processor(s) and the IDISK processors?
- What operating system services should be provided by the IDISK runtime system?
- How intelligent must IDISK nodes be to run important database software? Can algorithms for these kernels trade-off more disk accesses for less memory capacity, to accommodate IDISK’s reduced memory capacity, and still have reasonable performance?
- Will algorithmic and index innovation reduce processing demands and disk accesses for decision support? Or will decision support, by its ad hoc nature, always ask new questions that are best answered using substantial processing and disk accesses?
- How can IDISK processing and multiple serial lines be used to provide data redundancy and high availability in the presence of disk failures?
- How can the information made available by the tighter integration of disk and CPU be used to improve performance?
- An IDISK processor can be used for many operations (e.g., database functionality, disk arm scheduling). How should these and other competing demands be scheduled effectively?
- How well do IDISK architectures scale as decision support systems grow in size and requests in the future?
- How well will the IDISK architecture work for more update-intensive workloads, such as online transaction processing (OLTP)?
- What are the solutions to “hot spots,” whereby most of the queries naturally go to a single IDISK?
- Will commercial database authors restructure their code to take advantage of potential IDISK benefits?

7 Conclusions

"The history of DBMS research is littered with innumerable proposals to construct hardware database machines to provide high performance operations. In general these have been proposed by hardware types with a clever solution in search of a problem on which it might work. [49]”

As this quote indicates, work on database machines has a checkered past, causing some to doubt this new line of research. We view this skepticism as healthy, and in overcoming our own doubts we have become increasingly convinced that intelligent disks can be efficient and cost-effective in the short term. Advances in algorithms, communication and the possibility that on-disk processors will become commodities make IDISK more viable than these earlier efforts. If we are correct, then the emergence of IDISK represents a major opportunity to retool data-intensive software systems for significantly increased performance.

At this early stage of our research, we are mapping out a variety of hardware/software scenarios for designing efficient, cost-effective data systems of the future. In this paper we described two hardware architectures and several software scenarios for IDISK. In addition, we enumerated several potential advantages of this system design, and outlined challenges that must be overcome for this design to be successful. As this research progresses, we should develop a better understanding of these challenges and their solutions. Meanwhile we believe that realizing the high-performance database system of the future may require significant research at both the hardware and software levels.

8 Acknowledgments

The case presented in this paper benefited from discussions with Eric Anderson, Paul Aoki, Remzi Arpaci-Dusseau, James Beck, Aaron Brown, John Kubiatowicz, David Oppenheimer, Randi Thomas, and Kathy Yelick at Berkeley, and with Jim Gray and Catharine van Ingen at Microsoft Bay Area Research Center. This research was supported by DARPA (DABT63-C-0056), the California State MICRO program, NASA grant FDNAGW-5198, and by research grants from Hitachi, Intel, LG Semicon, Microsoft, Silicon Graphics, and Sun Microsystems. Kim Keeton was supported by a Lucent Technologies Graduate Fellowship. Joe Hellerstein was supported by an NSF Career Award and a Sloan Foundation Fellowship.

9 References

[1] A. Acharya, M. Uysal, and J. Saltz. “Active disks: programming model, algorithms and evaluation,” to appear in *Proc. 8th Int. Conf. on Architectural Support for Programming Languages and Operating Systems (ASPLOS-VIII)*, Oct. 1998.

[2] A. Acharya, M. Uysal, and J. Saltz. “Active disks,” Technical Report, TRCS98-06, University of California, Santa Barbara. March 1998.

[3] AMCC S2025 single-chip switch. `http://www.amcc.com/Products/CPSwitch/S2025.htm`.

[4] D. Anderson. “Consideration for smarter storage devices,” presentation given at National Storage Industry Consortium’s (NSIC’s) Network-Attached Storage Devices (NASD) working group meeting, June 1998. Available from `http://www.nsic.org/nasd/`.

[5] E. Anderson. “Results of the 1995 SANS Survey,” ;login, the Usenix Association newsletter, Vol. 20, No. 5, Oct. 1995.

[6] R. H. Arpaci-Dusseau, A. C. Arpaci-Dusseau, D. E. Culler, J. M. Hellerstein, and D. A. Patterson. “The architectural costs of streaming I/O: a comparison of workstations, clusters, and SMPs,” *Proc. 4th Sym. on High-Performance Computer Architecture*, February 1998, pp. 90 - 101.

[7] A. C. Arpaci-Dusseau, R. H. Arpaci-Dusseau, D. E. Culler, J. M. Hellerstein, and D. A. Patterson. “High-performance sorting on networks of workstations,” *Proceedings of SIGMOD ‘97*, May 1997, pp. 243 - 254.

[8] E. Babb. “Implementing a relational database by means of specialized hardware,” *ACM Transactions on Database Systems*, Vol. 4, No. 1, March 1979.

[9] J. Banerjee, et al. “DBC - a database computer for very large data bases,” *IEEE Trans. on Computers*, June 1979.

[10] P. Bernstein, “Database Technology: What’s Coming Next?” Keynote presentation at *Fourth Symposium on High-Performance Computer Architecture*, February 1998.

[11] D. Bitton and J. Gray. “The rebirth of database machine research,” invited talk at *the 24th International Conference on Very Large Databases (VLDB ‘98)*, August 1998.

[12] H. Boral and D. J. DeWitt. “Database machines: an idea whose time has passed? A critique of the future of database machines,” *Proceedings of the Third International Workshop on Database Machines*, 1983, pp. 166 - 187.

[13] R. Boyd-Merritt. “Gigabit bus carries Intel into communications territory,” *EE Times*, `http://techweb.cmp.com/eet/news/98/998news/gigabit.html`.

[14] F. T. Chong, E. Brewer, F. T. Leighton, and T. F. Knight, Jr. “Building a better butterfly: the multiplexed metabutterfly,” Proceedings of the International Symposium on Parallel Architectures, Algorithms, and Networks, December 1994.

[15] Z. Cvetanovic and D. D. Donaldson. “AlphaServer 4100 performance characterization.” Digital Technical Journal. 8(4):3-20, 1996.

[16] W. J. Dally and J. Poulton. “Equalized 4 Gb/s signalling,” *Hot Interconnects IV Symposium Record*, September 1996.

[17] D. J. DeWitt. “DIRECT - A multiprocessor organization for supporting relational database management systems,” *IEEE Transactions on Computers*, June 1979, pp. 395-406.

[18] D. J. DeWitt and P. B. Hawthorn. “A performance evaluation of database machine architectures,” Proceedings of VLDB ‘81, pp. 199 - 213.

[19] J. Gray, private communication, Microsoft Bay Area Research Center, June 1998.

[20] J. Gray. “Parallel Database Systems,” tutorial given at *VLDB ‘94*, September 1994. Available from `http://www.research.microsoft.com/barc/Gray/`.

[21]J. He and R. Raphael, Informix Software, personal communication, January 1998.

[22] W. Hell. “RDBM - A relational data base machine: architecture and hardware design,” *Proc. 6th Workshop on Computer Architecture for Non-Numeric Processing*, June 1981.

[23] J. L. Hennessy and D. A. Patterson. *Computer Architecture: A Quantitative Approach*, second edition, Morgan Kaufman, San Mateo, CA., 1996.

[24] A. R. Hurson, L. L. Miller and S. H. Pakzad. *Parallel Architectures for Database Systems*, IEEE Computer Society Press, Washington, D. C., 1989.

[25] K. Keeton. "Statistical analysis of database decision support workloads on the Pentium Pro SMP," Unpublished project report for UC Berkeley graduate statistics course, May 1997.

[26] K. Keeton, D. Patterson, Y. He, R. Raphael, and W. Baker. "Performance characterization of the quad Pentium Pro SMP using OLTP workloads," *Proceedings of ISCA '98*, June 1998, pp. 15 - 26.

[27] H. O. Leilich, G. Stiege, and H. C. Zeidler. "A search processor for data base management systems," *Proceedings of the 4th Conference on Very Large Databases*, 1978.

[28] C. H. C. Leung and K. S. Wong. "File processing efficiency on the content addressable file store," *Proceedings of VLDB '85*, pp. 282 - 291.

[29] S. C. Lin, D. C. P. Smith, and J. M. Smith. "The design of a rotating associative memory for relational database applications," *Transactions on Database Systems*, Vol. 1, No. 1, March 1976, pp. 53-75.

[30] S. E. Madnick. "The Infoplex database computer: concepts and directions," *Proc. IEEE Computer Conf.*, Feb. 1979.

[31] A. Grizzaffi Maynard, C. M. Donnelly, and B. R. Olszewski. "Contrasting characteristics and cache performance of technical and multi-user commercial workloads." In *Proc. 6th Int. Conf. on Architectural Support for Programming Languages and Operating Systems*, pages 145–156, Oct. 1994.

[32] J. Menon, private communication, IBM, Feb. 22, 1998.

[33] M. Missikoff. "An overview of the project DBMAC for a relational machine," *Proceedings of the 6th Workshop on Computer Architecture for Non-Numeric Processing*, June 1981.

[34] "Chart watch: mobile processors," *Microprocessor Report*, June 2, 1997, p. 35.

[35] "Chart watch: workstation processors," *Microprocessor Report*, July 14, 1997, pp. 23.

[36] NCR WorldMark/Teradata 1 TB TPC-D executive summary, available from `http://www.tpc.org/`.

[37] E. A. Ozkarahan, S. A Schuster, and K. C. Smith. "RAP - associative processor for database management," *AFIPS Conference Proceedings*, Vol. 44, 1975, pp. 379 - 388.

[38] G. Papadopoulos. "Future of Computing." Unpublished talk, NOW Workshop, Lake Tahoe, CA USA, 27 July 1997.

[39] D. Patterson and K. Keeton. "Hardware Technology Trends and Database Opportunities," Keynote address at *SIGMOD '98*, June 1998. Available at `http://www.cs.berkeley.edu/~pattrsn/talks.html`.

[40] D. Patterson, T. Anderson, N. Cardwell, R. Fromm, K. Keeton, C. Kozyrakis, R. Thomas, and K. Yelick. "A case for intelligent RAM," *IEEE Micro*, vol.17, no.2, March-April 1997. pp.34-44.

[41] S. E. Perl and R. L. Sites. "Studies of windows NT performance using dynamic execution traces," *Proc. of the Second USENIX Symposium on Operating Systems Design and Implementation*, pages 169-184, 1996.

[42] G. F. Pfister. *In search of clusters: the coming battle in lowly parallel computing*, Prentice Hall PTR, Upper Saddle River, New Jersey, 1995.

[43] E. Riedel, G. Gibson, and C. Faloutsos. "Active Storage For Large-Scale Data Mining and Multimedia," *Proceedings of the 24th International Conference on Very Large Databases (VLDB '98)*, August 1998.

[44] Seagate Cheetah drive homepage, accessible from `http://www.seagate.com/`.

[45] S. Schuster, et al. "RAP.2 - an associative processor for databases and its applications," *IEEE Trans. on Computers*, June 1979.

[46] T. Shimizu, et al. "A multimedia 32 b RISC microprocessor with 16 Mb DRAM," *ISSCC Digest of Technical Papers*, San Francisco, CA, USA, 8-10 Feb. 1996, pp. 216-17, 448.

[47] The Sort Benchmark Homepage. `http://www.research.microsoft.com/research/barc/SortBenchmark/default.html`.

[48] P. Stenstrom, E. Hagersten, D. J. Lilja, M. Martonosi, and M. Venugopal. "Trends in shared memory multiprocessing." *IEEE Computer*, pages 44-50, December, 1997.

[49] M. Stonebraker, editor. *Readings in Database Systems*, second edition, Morgan Kaufmann Publishers, San Francisco, 1994, p. 603.

[50] S. Y. W. Su and G. J. Lipovski. "CASSM: a cellular system for very large data bases," *Proceedings of the VLDB Conference*, 1975, pp. 456-472.

[51] Transaction Processing Performance Council, TPC Benchmark D (Decision Support) Standard Specification Revision 1.2.1, December 15, 1996.

[52] J. Turley. "NEC VR5400 makes media debut," *Microprocessor Report*, March 9, 1998.

[53] R. Wang. "Cluster file systems," Unpublished talk at NOW Workshop, Lake Tahoe, CA USA, January 1998.

[54] J. Wilkes. "DataMesh - parallel storage systems for the 1990s," *Proceedings of the 11th IEEE Mass Storage Symposium*, October 1991.

[55] J. Wilkes. "DataMesh research project, phase 1," *Proc. USENIX File Systems Workshop*, May 1992. pp. 63 - 69.

[56] R. Winter and K. Auerbach. "Giants walk the earth: the 1997 VLDB survey," *Database Programming and Design*, volume 10, number 9, September 1997, pp. S2 - S9+.

[57]R. Winter and K. Auerbach. "The big time: the 1998 VLDB survey," *Database Programming and Design*, volume 11, number 8, August 1998.

[58] C. K. Yang and M. A. Horowitz. "A 0.8 mm CMOS 2.5 Gb/s oversampled receiver for serial links," *1996 IEEE ISSCC Digest of Technical Papers*, February 1996.