Systems Support for Scalable Data Mining

William A. Maniatty
Department of Computer Science
University at Albany
Albany, NY 12222
maniatty@cs.albany.edu

Mohammed J. Zaki
Department of Computer Science
Rensselaer Polytechnic Institute
Troy, NY 12180
zaki@cs.rpi.edu

ABSTRACT

The current generation of data mining tools have limited capacity and performance, since these tools tend to be sequential. This paper explores a migration path out of this bottleneck by considering an integrated hardware and software approach to parallelize data mining. Our analysis shows that parallel data mining solutions require the following components: parallel data mining algorithms, parallel and distributed data bases, parallel file systems, parallel I/O, tertiary storage, management of online data, support for heterogeneous data representations, security, quality of service and pricing metrics. State of the art technology in these areas is surveyed with an eye towards an integration strategy leading to a complete solution.

General Terms

Scalable Knowledge Discovery and Data Mining

Keywords

Data Mining, KDD, Parallelism, Large Data Sets

1. INTRODUCTION

Knowledge discovery in databases (KDD) employs a variety of techniques, collectively called *data mining*, to uncover trends in large volumes of data. Many applications generate (or acquire) data faster than it can be analyzed using existing KDD tools, leading to perpetual data archival without retrieval or analysis. Furthermore, analyzing sufficiently large data sets can exceed the available computational resources of existing computers. In order to reverse the vicious cycle induced by these two problematic trends, the issues of performing KDD faster than the rate of arrival and increasing capacity must simultaneously be dealt with. Fortunately, novel applications of parallel computing techniques should assist in solving these large problems in a timely fashion.

Parallel KDD (PKDD) techniques are not currently that common, though recent algorithmic advances seek to address these problems [37; 113; 115; 54]. However, there has been no work in designing and implementing large-scale parallel KDD systems, which must not only support the mining algorithms, but also the entire KDD process, including the pre-processing and post-processing steps (in fact, it has been posited that around 80% of the KDD effort is spent in these steps, rather than mining). The picture gets even more complicated when one considers persistent data management of mined patterns and models.

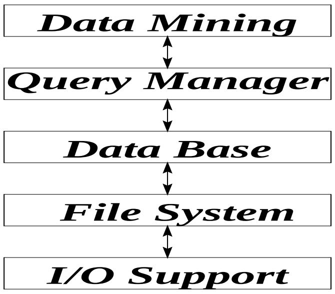

Figure 1: KDD Abstraction Layers

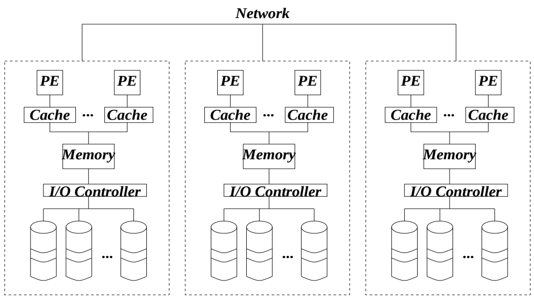

Figure 2: Cluster of SMP Nodes

Given the infancy of KDD in general, and PKDD in particular, it is not clear how or where to start, to realize the goal of building a PKDD system that can handle terabyte-sized (or larger) central or distributed datasets. Part of the problem stems from the fact that PKDD draws input from diverse areas that have been traditionally studied in isolation. Typically, the KDD process is supported by a hierarchical architecture consisting of the following layers: (from bottom to top) I/O Support, File System, Data Base, Query Manager, and Data Mining as shown in Figure 1. However, the current incarnations of this architecture tend to be sequential, limiting both problem size and performance. To implement a successful PKDD toolkit, we need to borrow, adapt, and enhance research in fields such as super-, meta- and heterogeneous-computing environments, parallel and distributed databases, parallel and distributed file systems, parallel I/O, mass storage systems, and so on (not to mention the other fields that make up KDD — statistics, machine learning, visualization, etc.).

This paper represents a first step in the process of unifying these diverse technologies and leveraging them within the PKDD system. Since form follows function, we first examine factors motivating PKDD, including trends in data sources and applications (Section 2). We then explore techniques for migrating the system requirements for PKDD and the extant solutions (or lack thereof), i.e., the *what* and the *how* of PKDD. These requirements follow from: the basic requirements imposed by KDD (Section 3), current KDD algorithmic techniques (Section 4), the trends in commodity hardware design (Section 5) and software requirements (Section 6). One difficulty in making such a survey is that each research community has its own jargon, which we will try to make accessible by describing it within a common PKDD framework.

2. DATA SOURCES IMPACTING KDD

Recall that until relatively recently, data acquisition was hard and scientists manually generated hypotheses and extrapolated trends out of a few data points. Modern automated techniques for data acquisition have reversed this trend [58; 71]. This reversal inundates users with a sea of data, motivating the use of automated forms of hypothesis generation. Here, we examine some trends focusing on sources of data and what users are likely to want to do with that data. Grossman et al. [42] describe prevalent forms of user data as being in one of the following categories:

- web based data,
- business and E-commerce data,
- and scientific, engineering and health-care related data.

In this section we consider some recent trends in these areas and examine their potential impact on KDD.

2.1 Web based data

The growth of the World Wide Web (web) is happening at a phenomenal rate, both in terms of the amount of data (Lawrence and Giles recently estimated that there were over 800 million accessible web pages [63; 64]) and the complexity in the data. Hearst [45] describes information retrieval (IR) approaches as having historically focused on extracting relevant information from large bodies of coexisting information based on user's queries, for such applications as summarization [99]. Kosala and Blockeel [59] recently surveyed web mining research, focusing on a contrast between IR, web mining and database approaches. Brewington et al. [17] have explored the use of distributed agents for IR, to distribute the load and reduce data communication costs. Text mining is an important tool for processing web data, and differs from traditional IR because, the content is different, and information about the data distribution and network is also of interest.

Web content is expressed in the *hypertext markup language* (HTML) and the *extended markup language* (XML) which impose additional structure on data than pure text. Florescu, et al. [36] recently proposed query language extensions to XML to access the semi-structured data. Heterogeneity of data types in the web's content is challenging, with multimedia data playing a large role [1; 2]. One of the fields in XML is a *document type descriptor* (DTD) which indicates what kind of information is contained in the document (for use in say search engines, document storage or query processing). Garofalakis et al. [40] recently described their XTRACT system for automating selection of DTD fields from XML pages. The dynamic nature of the data makes web mining challenging. Brewington and Cybenko have recently estimated that 95% of frequently accessed pages change within 7 days prior to access [15] and that a significant number of web pages are generated dynamically. This poses the problem of efficiently tracking recent updates on popular web pages with minimal overhead for search engines [16].

Recent advances in mobile technology have impacted the way people interact with the web. The *wireless application protocol* (WAP) specifies extensions to the XML, called the *wireless markup language* [109; 110] (WML) standards to make content viewable on small hand portable devices [111; 108]. WML is an application of XML, which specifies features for content providers to describe layout of "stacks of cards" on a small display interface in many portable devices. Such specifications can profoundly impact which data will be visible to users, and will influence what solutions are acceptable to queries. *Ubiquitous computing* has been used to refer to immersing large numbers of autonomous networked computers equipped with sensors and controllers into real world environments [105; 106]. *Wearable computing* [72; 73] refers to sensor based computing worn by the user (much like clothing) for sensory enhancement, automated recording of observations, and continuous networked access. Rhodes et al. [81], recently described wearable computing as providing privacy and personalization while ubiquitous computing provides resource management and localizes both information and control. The integration of these two technologies using an agent based system, *Hive*, seeks to provide privacy while permitting users to perform discovery queries on the data [81].

2.2 Business and E-Commerce Data

Both traditional and new firms have greatly expanded e-commerce systems, which introduces both challenging KDD problems and many hard distributed computing problems. Kohavi and Provost [58] recently suggested that e-commerce is the killer domain for data mining technology, due to the accuracy of the data obtained, the ability to check the results of mining, and the volume of data generated.

E-commerce systems require user's trust to ensure widespread adoption, with approaches such as Netbill [95] relying on a combination of security and trust to support electronic transactions [18]. Vendors who gather data about both individuals and customers in general seek to provide targeted advertising via *personalization*. Personalization is challenging to reconcile with the users need for security, and frequently relies on using cookies to monitor a user's web access patterns [9]. Lawrence, et al. [62] partition personalization approaches into

- *content-based filtering*, where recommendations to a customer are made based on past purchasing behavior attributed to that customer and
- *collaborative filtering*, which groups customers accord-

ing to purchase habits, making recommendations based on those customers with similar buying habits.

The collaborative filtering schemes were found to avoid over specialization problems which are induced by the content-based filtering approach [62]. Lawrence, et al. [62] applied collaborative filtering schemes in a supermarket chain, via a client-server system. Customers were issued portable digital assistants (PDAs) that generated recommendations. In the deployed system, the servers did a larger amount of the KDD computing relative to the clients [62], however, the PDAs are expected to support more compute intensive tasks as processor and memory capacity increases. Schafer, et al. [88] describe several e-commerce recommender applications, and describes several requirements for successful recommender systems, including:

- timely delivery of the recommendation as a requirement for a successful recommender system and
- the problem of giving appropriate advice to new customers (who do not have much data).

2.3 Scientific, Engineering and Health-care Data

Scientific and engineering data has always been a popular application domain for researchers. Traditionally, scientists have attempted to combine theory, simulation and experimentation. Modern experimentation generates large volumes of data using sensor technology. This rate of data acquisition is accelerating with the advent of sensor based computers (this is effectively an application of the ubiquitous computing described earlier), motivating distributed KDD approaches for in situ mining of data when possible. Bioinformatics and functional genomics face challenging problems in analyzing genetic data. However non-technical restrictions can be imposed on data, increasing the level of difficulty, i.e., health-care data presents special challenges, due to a strong need for confidentiality.

Many simulations in science use simplifying assumptions to make the model more tractable. Many phenomena which are discrete are approximated using continuous modeling techniques, which tend to be accurate for modeling large numbers of entities. Simulations can be categorized by:

- whether they treat time as continuous or discrete and their degree of synchrony,
- how they treat space, using the following approaches:
  - aspatial,
  - continuous space,
  - discrete space
- how entities are modeled:
  - Discrete aggregated
  - individual based models
  - continuous

Engineers and physical scientists frequently use numerical methods for partial differential equations for modeling materials [35; 34; 33]. Recently, there has been significant interest in biological modeling, which tends to use aspatial models (reminiscent of homogeneous mixing models in differential equations [13]) or diffusion based models [11]. The average case behaviors are often assumed to dominate in many models. However, there are systems where the average case behavior is not representative [30]. For small, or sparse populations, the variation in the distribution of modeled entities tends to dominate [31], motivating a spatially explicit individual based approach for improved simulation fidelity [101; 100; 71; 70]. Discrete event simulation is useful for modeling asynchronous systems of entities undergoing discrete state changes, which makes it popular for modeling computer and network based systems [51; 65; 32]. Parallel discrete event simulation (PDES) [38; 65; 27; 19; 83] is a widely used approach to provide capacity for modeling large numbers of discrete individual simulation entities. In the construction of individual based models of large populations, the efficiency of the simulation engine strongly constrains data placement for efficiency. Measuring the simulation trajectory should use in situ on-line approaches when possible to avoid the prohibitive overhead of remapping the data [68; 69; 71; 70], and to summarize the trajectory to reduce data storage requirements. KDD tools have not yet been widely used for analyzing simulation data, significant opportunities appear to be present for both time series analysis of the trajectory of individual simulations (typically on-line), and for contrasting trajectories of multiple simulations (off-line). For example, clustering of high dimensional data [4; 3] may be useful for phylogenetic analysis of large populations, which can occur via experiment [112; 90], or during simulation [71]. Hidden Markov models and association rule mining can help in modeling and understanding protein folding [117]. Additional opportunities exist for tools to compare, contrast and integrate simulation trajectory between continuous models and discrete models for adaptive simulation approaches.

3. PKDD REQUIREMENTS

We use the range of applications to direct the derivation of a wish-list of desirable features in a functional PKDD system. This wish-list will guide the rest of the survey. We mainly concentrate on aspects of PKDD that have not received wide attention as yet.

- *Algorithm Evaluation:* Algorithmic aspects that need attention are the ability to handle high dimensional datasets, to support terabyte data-stores, to minimize number of data scans, etc. An even more important research area is to provide a rapid development framework to implement and conduct the performance evaluation of a number of competing parallel methods for a given mining task. Currently this is a very time-consuming process, and there are no guidelines when to use a particular algorithm over another.
- *Process Support:* The toolkit should support all KDD steps, from pre-processing operations for like sampling, discretization, and feature subset selection, to post-processing operations like rule grouping, pruning, summarization and model scoring. Other aspects include (persistent) pattern management operations like caching, efficient retrieval, and meta-level mining.
- *Location Transparency:* The PKDD system should be able to seamlessly access and mine datasets regardless of their location, be they centralized or distributed.

- *Data Type Transparency:* The system should be able to cope with heterogeneity (e.g., different database schemas), without having to materialize a join of multiple tables. Other difficult aspects deal with handling unstructured and semistructured data, including: (hyper-)text, spreadsheets, and a variety of other data types.
- *System Transparency:* This refers to the fact that the PKDD system should be able to seamlessly access file systems, databases, or data archives. Databases and data warehouses represent one kind of data repositories, and thus it is crucial to integrate mining with DBMS to avoid extracting data to flat files. On the other hand, a huge amount of data remains outside databases in flat-files, web-logs, etc. The PKDD system must therefore bridge the gap that exists today between databases and file-systems [24]. This is required since database systems today offer little functionality to support mining applications [5], and most research on parallel file systems and parallel I/O has looked at scientific applications, while data mining operations have very different workload characteristics.
- *Security, QoS and Pricing:* In an increasingly networked world, one constantly needs access to proprietary third-party and other remote datasets. The two main issues that need attention here are security and Quality-of-Service (QoS). We need to prevent unauthorized mining, and we need to provide cost-sensitive mining to guarantee a level of performance. These issues are paramount in web-mining for e-commerce.
- *Availability, Fault Tolerance and Mobility:* Distributed and parallel systems have more points of failure than centralized systems. Furthermore temporary disconnections (which are frequent in mobile computing environments) and reconnections by users should be tolerated with a minimal penalty to the user. Many real world applications cannot tolerate outages, and in the presence of QoS guarantees and contracts, outages can breach the agreements between providers and users. Little work has been done to address this area as well.

In the discussion below, due to space constraints, we choose to concentrate only on the algorithmic and hardware trends, and system transparency issues (i.e., parallel I/O and parallel and distributed databases), while briefly touching on other aspects.

4. MINING METHODS

Faster and scalable algorithms for mining will always be required. Parallel and distributed computing seems ideally placed to address these big data performance issues. However, achieving good performance on today's multiprocessor systems is a non-trivial task. The main challenges include synchronization and communication minimization, work-load balancing, finding good data layout and data decomposition, and disk I/O minimization.

The parallel design space spans a number of systems and algorithmic components such as the hardware platform (shared vs. distributed), kind of parallelism (task vs. data), load balancing strategy (static vs. dynamic), data layout (horizontal vs. vertical) and search procedure used (complete vs. greedy).

Recent algorithmic work has been very successful in showing the benefits of parallelism for many of the common data mining tasks including association rules [6; 23; 43; 118; 46], sequential patterns [93; 114], classification [92; 52; 116; 97], regression [107] and clustering [53; 29; 84].

The typical trend in parallel mining is to start with a sequential method and pose various parallel formulations, implement them, and conduct a performance evaluation. While this is very important, it is a very costly process. After all, the parallel design space is vast and results on the parallelization of one serial method may not be applicable to other methods. The result is that there is a proliferation of parallel algorithms without any standardized benchmarking to compare and provide guidelines on which methods work better under what circumstances. The problem becomes even worse when a new and improved serial algorithm is found, and one is forced to come up with new parallel formulations. Thus, it is crucial that the PKDD system support rapid development and testing of algorithms to facilitate algorithmic performance evaluation.

One recent effort in this direction is discussed by [96], which emphasizes the importance of and presents a set of cost measures that can be applied to parallel algorithms to predict their computation, data access, and communication performance. These measures make it possible to compare different parallel implementation strategies for data-mining techniques without benchmarking each one.

A different approach is to build a data mining kernel that supports common data mining operations, and is modular in design so that new algorithms or their "primitive" components can be easily added to increase functionality. An example is the MKS [8] kernel. Also, generic set-oriented primitive operations were proposed in [37] for classification and clustering, which were integrated with a parallel DBMS.

5. HARDWARE MODELS AND TRENDS

The current hardware trends reflect that memory and disk capacity are increasing at a much higher rate than their speed. Furthermore, CPU capacity is roughly obeying Moore's law, which predicts doubling performance approximately every 18 months. To combat bus and memory bandwidth limitations, caching is used to improve the mean access time, giving rise to Non-Uniform Memory Access architectures. To accelerate the rate of computation, modern machines frequently increase the number of processing elements in an architecture. Logically, the memory of such machines is kept consistent, giving rise to a shared memory model, called Symmetric Multiprocessing (SMP) in the architecture community and *shared everything* in the database community [28; 104]. However, the scalability of such architectures is limited. So, for higher degrees of parallelism, a cluster of SMP nodes is used a network is used to permit cooperation between SMP nodes, as shown in Figure 2. This model, called *shared-nothing* in database literature, is also the preferred architecture for parallel databases [28].

Redundant arrays of independent (or inexpensive) disks, called RAID [22], has gained popularity to increase I/O bandwidth and storage capacity, reduce latency, and (optionally) support fault tolerance. In many systems, since the amount of data exceeds that which can be stored on disk, *tertiary storage* is used, typically consisting of one or more removable media devices with a juke box to swap the loaded media.

In addition to the current trends, there have been other ideas to improve the memory and storage bottlenecks. *Active Disks* [82] and *Intelligent Disks* [55] have been proposed as a means to exploit the improved processor performance of embedded processors in disk controllers to allow more complex I/O operations and optimizations, while reducing the amount of traffic over a congested I/O bus. Intelligent RAM (IRAM) [61] seeks to integrate processing elements in the memory. Active disks and IRAM are not currently prevalent, as the required hardware and systems software are not commonly available.

6. SOFTWARE INFRASTRUCTURE

Since our goal is to use commodity hardware, much of the support for our desired functionality is pushed back into the software. Much of the support for the exploitation of parallelism in PKDD has not been developed. In this section we discuss some of the system transparency issues in PKDD systems, i.e., support for seamless access to databases and file systems and parallel I/O. We review selected aspects of these areas.

The most common database constructions currently in use are relational databases, object oriented databases, and object-relational databases. The data base layer ensures referential integrity and provides support for queries and transactions on the data [77]. The data base layer is frequently accessed via a query language, such as SQL. We are primarily interested in parallel and distributed database systems [28; 104], which have data sets spanning disks. The primary advantages of such systems are that capacity of storage is improved and that parallelizing of disk access improves bandwidth and (for large I/O's) can reduce latency. Early parallel database research explored special-purpose database machines for performance [48]. However, the current preference is to use available parallel platforms, with shared-nothing paradigm as the architecture of choice. Shared-nothing database systems include Teradata, Gamma [26], Tandem [102], Bubba [12], Arbre [66], etc. We refer the reader to [28; 104; 56] for excellent survey articles on parallel and distributed databases. Parallel database research into data partitioning (over disks) methods used is particularly relevant to PKDD. Parallel database partitioning methods include:

- simple *round-robin* partitioning, where records are distributed evenly among the disks,
- *hash partitioning*, which is most effective for applications requiring associative access and
- *range partitioning* clusters records with similar attributes together.

Most parallel data mining work to-date has used a round-robin approach to data partitioning. Other methods might be more suitable. Exploration of efficient multidimensional indexing structures for PKDD is required [39]. The vast amount of work on parallel relational query operators, particularly parallel join algorithms, is also of relevance [79]. The use of DBMS *views* [77] to restrict the access of a DBMS user to a subset of the data, can be used to provide security in KDD systems.

Distributed File Systems (DFS) support collections of many small files [86; 14; 10] while providing location transparency, and in some cases operation during periods of disconnection (as happens when mobile clients are used). Parallel I/O and file systems techniques are geared to handling large data sets in a distributed memory environment (a few large files), and appear to be a better fit than DFS for managing the large data sets found in KDD applications. Parallel File Systems and Parallel I/O techniques have been widely studied; Kotz maintains an archive and bibliography [60], which has a nice reference guide [98]. Use of parallel I/O and file systems becomes necessary if RAID devices have insufficient capacity (due to scaling limitations) or contention for shared resources (e.g. buses or processors) exceeds the capacity of SMP architectures. The Scalable I/O initiative (SIO) includes many groups, including the *Message Passing Interface* (MPI) forum, which has adopted a MPI-IO API [103] for parallel file management. MPI-IO is layered on top of local file systems. MPI uses a run time type definition scheme to define communication and I/O entity types. The ROMIO library [103] implements MPI-IO in Argonne's MPICH implementation of MPI. ROMIO automates scheduling of aggregated I/O requests and uses a middleware layer, called ADIO, to provide portability and isolate implementation dependent parts of MPIO. PABLO, another SIO member group, has created the *portable parallel file systems* (PPFS II), designed to support efficient access of large data sets in scientific applications with irregular access patterns. More information on parallel and distributed I/O and file systems appears in [60; 20; 41; 87; 75; 76; 89; 91].

Users of PKDD systems are interested in maximizing performance. Prefetching is an important performance enhancing technique that can reduce the impact of latency by overlapping computation and I/O [25; 57; 78]. In order for prefetching to be effective, the distributed system uses *hints* which indicate what data is likely to be used in the near future. Generation of accurate hints (not surprisingly) tends to be difficult since it relies on predicting a program's flow of control. Many hint generation techniques rely on traces of a program's I/O access patterns. [57] surveyed a range of trace driven techniques and prefetching strategies, and provided performance comparisons. [67] recently used machine learning tools to analyze I/O traces from the PPFS, relying on artificial neural networks for on-line analysis of the current trace, and hidden markov models to analyze data obtained by profiling. [21] developed *SpecHint* which generates hints via speculative execution. We conjecture that PKDD techniques can be used to identify reference patterns, to provide hint generation and to address open performance analysis issues [80].

As we noted earlier, integration of various systems components for effective KDD is lagging. The current state of KDD tools can accurately be captured by the term *flat-file mining*, i.e., prior to mining, all the data is extracted into a flat file, which is then used for mining, effectively bypassing all database functionality. This is mainly because traditional databases are ill-equipped to handle/optimize the complex query structure of mining methods. However, recent work has recognized the need for integrating of the database, query management and data mining layers [7; 85]. [7] postulated that better integration of the query manager, database and data mining layers would provide a speedup. [85] confirmed that performance improvements could be attained, with the best performance obtained in *cache-mine* which caches and mines the query results on a local disk. SQL-like operators for mining association rules have also been developed [74]. Further, proposals for data mining query language [44; 49; 50; 94] have emerged. We note that most of this work is targeted for serial environments. PKDD efforts will benefit from this research, but the optimization problems will of course be different in a parallel setting. Some exceptions include the parallel generic primitives proposed in [37], and Data Surveyor [47], a mining tool that uses the Monet database server for parallel classification rule induction. We further argue that we need a wider integration of parallel and distributed databases and file systems, to fully mine all available data (only a modest fraction of which actually resides in databases). Integration of PKDD and parallel file systems should enhance performance by improving hint generation in prefetching. Integrated PKDD can use parallel file systems for storing and managing large data sets and use distributed file systems as an access point suited to mobile clients for management of query results.

7. CONCLUSIONS

The requirements of KDD systems follow from what kinds of data users have, and what the users are likely to want to do with the data. We explored these issues, and how trends in networking, commerce, and science impacted these requirements. These requirements were presented as a list of desirable design features of parallel KDD systems and motivated a brief survey of existing algorithmic and systems support for building such large-scale mining tools. We focused on the state-of-the-art in databases, file systems and parallel I/O techniques. We observe that implementing an effective PKDD system requires integration of these diverse sub-fields into a coherent and seamless system. Emerging issues in PKDD include benchmarking, security, availability, mobility and QoS, motivating fresh research in these disciplines. Finally, PKDD approaches may be used as a tool in these areas (e.g. hint generation for prefetching in parallel I/O), resulting in a bootstrapping approach to software development.

Acknowledgements

We are grateful to the comments of Ganesh Ramesh on this manuscript and the anonymous reviewers remarks.

8. REFERENCES

[1] S. Adali. Making peace with your multimedia. *IEEE Intelligent Systems*, 1998.

[2] S. Adali, C. Bufi, and Y. Temtanapat. Integrated search engine. In *Proceeding of the IEEE Knowledge abd Data Engineering Exchange Workshop, KDEX97*, pages 140–147, November 4 1997.

[3] C. C. Aggarwal and P. S. Yu. Finding generalized projected clusters in high dimensional spaces. In W. Chen, J. F. Naughton, and P. A. Bernstein, editors, *Proceedings of SIGMOD Conference 2000*, volume 29, pages 70–81. SIGMOD, ACM, 2000.

[4] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 560–573, Seattle, WA, June 1998.

[5] R. Agrawal, T. Imielinski, and A. Swami. Database mining: A performance perspective. *IEEE Transactions on Knowledge and Data Engineering*, 5(6):914–925, Dec. 1993.

[6] R. Agrawal and J. Shafer. Parallel mining of association rules. *IEEE Trans. on Knowledge and Data Engg.*, 8(6):962–969, Dec. 1996.

[7] R. Agrawal and K. Shim. Developing tightly-coupled data mining applications on a relational database system. In *Proc. of the 2nd Int'l Conference on Knowledge Discovery in Databases and Data Mining*, Portland, Oregon, 1996.

[8] S. Anand, B. Scotney, M. Tan, S. McLean, D. Bell, J. Hughes, and I. Magill. Designing a kernel for data mining. *IEEE Expert: Intelligent Systems and Their Applications*, pages 65–74, Mar. 1997.

[9] S. Ansari, R. Kohavi, L. Mason, and Z. Zheng. Integrating e-commerce and data mining: Architecture and challenges. In *Proceedings of WEBKDD'2000 Workshop, Web Mining for E-Commerce – Challenges and Opportunities*, Boston, MA, August 20 2000. ACM. Held in conjunction with the ACM-SIGKDD Conference on Knowledge Discovery in Databases (KDD'2000).

[10] R. H. Arpaci-Dusseau, E. Anderson, N. Treuhaft, D. E. Culler, J. M. Hellerstein, D. Patterson, and K. Yelick. Cluster i/o with river: Making the fast case common. In *Proceedings of the Sixth Workshop on I/O in Parallel and Distributed Systemm IOPADS '99*, Atlanta, GA, May 1999.

[11] R. B. Banks. *Growth and Diffusion Phenomena : Mathematical Frameworks and Applications*. Springer-Verlag, Berlin, Germany, 1994.

[12] H. Boral, W. Alexander, L. Clay, G. Copeland, S. Danforth, M. Franklin, B. Hart, M. Smith, and P. Valduriez. Prototyping Bubba, a highly parallel database system. *IEEE Transactions on Knowledge and Data Engineering*, 2(1), Mar. 1990.

[13] W. E. Boyce and R. C. DiPrima. *Elementary Differential Equations and Boundary Value Problems*. Wiley, 1992.

[14] P. J. Braam. File systems for clusters from a protocol perspective. In *Proceedings of the Second Extreme Linux Topics Workshop*, Monterey, CA, June 1999.

[15] B. Brewington and G. Cybenko. How dynamic is the web? In *Proceedings of the Ninth International World Wide Web Conference*, May 2000.

[16] B. Brewington and G. Cybenko. Keeping up with the changing web. *IEEE Computer*, 33(5):52–58, May 2000.

[17] B. Brewington, R. Gray, K. Moizumi, D. Kotz, G. Cybenko, and D. Rus. Mobile agents for distributed information retrieval. In M. Klusch, editor, *Intelligent Information Agents*, chapter 15, pages 355–395. Springer-Verlag, 1999.

[18] L. Camp, M. Harkavy, J. D. Tygar, and B. Yee. Anonymous atomic transactions. In *Proceedings of the 2nd Usenix Workshop on Electronic Commerce*, pages 123 – 133, November 1996.

[19] C. Carothers, K. Perumalla, and R. M. Fujimoto. Efficient optimistic parallel simulations using reverse computation. In *Proceedings of the 1999 Workshop on Parallel and Distributed Simulations*, Atlanta, GA, May 1999.

[20] J. Carretero, F. Pérez, P. de Miguel, F. García, and L. Alonso. ParFiSys: A parallel file system for MPP. *ACM Operating Systems Review*, 30(2):74–80, Apr. 1996.

[21] F. Chang and G. Gibson. Automatic i/o hint generation through speculative execution. In *Proceedings of the 3rd Symposium on Operating Systems Design and Implementation*, February 1999.

[22] P. M. Chen, E. K. Lee, G. A. Gibson, R. H. Katz, and D. A. Patterson. RAID: High-performance, reliable secondary storage. *ACM Computing Surveys*, 26(2):145–185, June 1994.

[23] D. Cheung, J. Han, V. Ng, A. Fu, and Y. Fu. A fast distributed algorithm for mining association rules. In *4th Intl. Conf. Parallel and Distributed Info. Systems*, Dec. 1996.

[24] A. Choudhary and D. Kotz. Large-scale file systems with the flexibility of databases. *ACM Computing Surveys*, 28A(4), December 1996.

[25] T. Cortes. *High Performance Cluster Computing: Architectures and Systems*, volume 1, chapter Software Raid and Parallel File Systems, pages 463–495. Prentice Hall, 1999.

[26] D. DeWitt et al. The GAMMA database machine project. *IEEE Trans. on Knowledge and Data Engineering*, 2(1):44–62, Mar. 1990.

[27] E. Deelman and B. K. Szymanski. Breadth-first rollback in spatially explicit simulations. In *Proc. PADS97, 11th Workshop on Parallel and Distributed-Simulation*, Los Alamitos, CA, 1997. IEEE Computer Society.

[28] D. DeWitt and J. Gray. Parallel database systems: The future of high-performance database systems. *Communications of the ACM*, 35(6):85–98, June 1992.

[29] I. S. Dhillon and D. S. Modha. A data clustering algorithm on distributed memory machines. In [115].

[30] R. Durrett and S. A. Levin. The importance of being discrete (and spatial). *Theoretical Population Biology*, 46:363–394, 1994.

[31] M. Duryea, J. Gardner, T. Caraco, W. Maniatty, and B. K. Szymanski. Population dispersion and equilibrium infection frequency in a spatial epidemic. *Physica D*, 132:511–519, 1999.

[32] D. Estrin, M. Handley, J. Heidemann, S. McCanne, Y. Xu, and H. Yu. Network visualization with nam, the vint network animator. *IEEE Computer*, 33(11):63–68, November 2000.

[33] J. Flaherty, M. Dindar, R. Loy, M. Shephard, B. Szymanski, J. Teresco, and L. Ziantz. *Numerical Analysis*, chapter An adaptive and parallel framework for partial differential equations. Addison Wesley Longman, Edinburgh, UK, 1998.

[34] J. Flaherty, R. Loy, C. Ozturan, M. Shephard, B. Szymanski, and L. Z. J.D. Teresco. Parallel structures and dynamic load balancing for adaptive finite element computation. *Applied Numerical Mathematics*, 26(1-2):241–263, 1998.

[35] J. Flaherty, R. Loy, M. Shephard, B. K. Szymanski, J. Teresco, and L. Ziantz. Adaptive local refinement with octree load-balancing for the parallel solution of three-dimensional conservation laws. *Journal of Parallel and Distributed Computing*, 47:139–152, 1997.

[36] D. Florescu, A. Deutsch, A. Levy, D. Suciu, and M. Fernández. A query language for XML. In *Proceedings of Eighth International World Wide Web Conference*, Toronto, Canada, May 11-14 1999.

[37] A. Freitas and S. Lavington. *Mining very large databases with parallel processing*. Kluwer Academic Pub., Boston, MA, 1998.

[38] R. M. Fujimoto. Parallel discrete event simulation. In P. Heidelberger, K. J. Musselman, and E. A. MacNair, editors, *Proceedings of the 1989 Winter Simlation Conference*, pages 19–28, 1989.

[39] V. Gaede and O. Gunther. Multidimensional access methods. *ACM Computing Surveys*, 30(2):170–231, June 1998.

[40] M. Garofalakis, A. Gionis, R. Rastogi, S. Seshadri, and K. Shim. XTRACT: A system for extracting document type descriptors from XML document. In *SIGMOD International Conference on Management of Data*, Dallas, Texas, May 16-18 2000. ACM.

[41] G. Gibson, D. Nagle, W. Courtright, N. Lanza, P. Mazaitis, M. Unangst, and J. Zelenka. NASD scalable storage systems. In *USENIX99, Extreme Linux Workshop*, June 1999.

[42] R. Grossman, S. Kasif, R. Moore, D. Rocke, and J. Ullman. Data mining research: Opportunities and challenges - a report of three nsf workshops on mining large, massive, and distributed data. Available at `http://www.ncdm.uic.edu/M3D-finalreport.htm`.

[43] E.-H. Han, G. Karypis, and V. Kumar. Scalable parallel data mining for association rules. In *ACM SIGMOD Conf. Management of Data*, May 1997.

[44] J. Han, Y. Fu, W. Wang, K. Koperski, and O. Zaiane. DMQL: A data mining query language for relational databases. In *1st ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, June 1996.

[45] M. Hearst. Untangling text data mining. In *Proceedings of ACL'99: the 37th Annual Meeting of the Association for Computational Linguistics*, University of Maryland, June 20-26 1999. An Invited Paper.

[46] J. Hipp, U. Güntzer, and G. Nakhaeizade. Algorithms for association rule mining – a general survey and comparison. *SIGKDD Explorations*, 2(1):58–64, 2000.

[47] M. Holsheimer, M. L. Kersten, and A. Siebes. Data surveyor: Searching the nuggets in parallel. In U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*. AAAI Press, Menlo Park, CA, 1996.

[48] D. Hsiao. *Advanced Database Machine Architectures*. Prentice Hall, 1983.

[49] T. Imielinski and H. Mannila. A database perspective on knowledge discovery. *Communications of the ACM*, 39(11), Nov. 1996.

[50] T. Imielinski, A. Virmani, and A. Abdulghani. DataMine: Application programming interface and query language for database mining. In *2nd Intl. Conf. Knowledge Discovery and Data Mining*, Aug. 1996.

[51] R. Jain. *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, And Modeling*. Wiley, New York City, 1991.

[52] M. Joshi, G. Karypis, and V. Kumar. ScalParC: A scalable and parallel classification algorithm for mining large datasets. In *Intl. Parallel Processing Symposium*, 1998.

[53] D. Judd, P. McKinley, and A. Jain. Large-scale parallel data clustering. In *Int'l Conf. Pattern Recognition*, Aug. 1996.

[54] H. Kargupta and P. Chan, editors. *Advances in Distributed Data Mining*. AAAI Press, Menlo Park, CA, 2000.

[55] K. Keeton, D. Patterson, and J. Hellerstein. The case for intelligent disks. *SIGMOD Record*, 27(3):42–52, September 1998.

[56] M. Khan, R. Paul, I. Ahmed, and A. Gafoor. Intensive data management in parallel systems: A survey. *Distributed and Parallel Databases*, 7:383–414, 1999.

[57] T. Kimbrel, A. Tomkins, R. H. Patterson III, B. Bershad, P. Cao, E. W. Felton, G. A. Gibson, A. R. Karlin, and K. Li. A trace-driven comparison of algorithms for parallel prefetching and caching. In *Proceedings of the 2nd USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, pages 19–34, Seattle, WA, October 1996.

[58] R. Kohavi and F. Provost. Applications of data mining to e-commerce (editorial). *Data Mining and Knowledge Discovery*, January 2001.

[59] R. Kosala and H. Blockeel. Web mining research: A survey. *SIGKDD Explorations*, 2(1):1–15, July 2000.

[60] D. Kotz. The parallel i/o archive. Includes pointers to his Parallel I/O Bibliography, can be found at *http://www.cs.dartmouth.edu/pario/*.

[61] C. E. Kozyrakis and D. A. Patterson. New direction in computer architecture research. *IEEE Computer*, pages 24–32, November 1998.

[62] R. D. Lawrence, G. S. Almasi, V. Kotlyar, M. S. Viveros, and S. S. Duri. Personalization of supermarket product recommendations. *Data Mining and Knowledge Discovery*, January 2001.

[63] S. Lawrence and C. L. Giles. Searching the world wide web. *SCIENCE*, 280:98, 1998.

[64] S. Lawrence and C. L. Giles. Accessibility of information on the web. *Nature*, 400:107–109, 1999.

[65] Y.-B. Lin and P. Fishwick. Asynchronous parallel and distributed simulation (for PCS). *IEEE Transactions on Systems, Man, and Cybgernetics*, 26(4), 1996. University of Florita Tech Report tr95-005.

[66] R. Lorie, J. Daudenarde, G. Hallmark, J. Stamos, and H. Young. Adding inter-transaction parallelism to existing DBMS: Early experience. *IEEE Data Engineering Newsletter*, 12(1), Mar. 1989.

[67] T. M. Madhyastha and D. A. Reed. Exploiting global input/output access pattern classification. In *Proceedings of SC'97*, 1997. On CDROM.

[68] W. Maniatty, B. Szymanski, and T. Caraco. Implementation and performance of parallel ecological simulations. In *Proceedings IFIP WG10.3 International Conference on Applications in Parallel and Distributed Computing*, pages 93–102, Caracas, Venezuala, 1993. Elseveir Science Publishers B.V., Amsterdam, The Netherlands.

[69] W. Maniatty, B. Szymanski, and T. Caraco. Tempest: a fast spatially explicit model of ecological dynamics on parallel machines. In *Proceedings of the 1994 International Simulation Conference*, San Diego, CA, 1994. The Society For Computer Simulation.

[70] W. A. Maniatty, B. K. Szymanski, and T. Caraco. High-performance simulation of evolutionary aspects of epidemics. In *Proceedings of the PARA98 Workshop on Applied Parallel Computing in Large Scale Industrial and Scientific Problems*, volume 1541 of *Lecture Notes in Computer Science*, pages 322–331, Umeå University, Umeå Sweden, June 1998. Springer-Verlag.

[71] W. A. Maniatty, B. K. Szymanski, and T. Caraco. Parallel computing with generalized cellular automata. *Parallel and Distributed Programming Practices*, 1(1):85–104, January 1998. Also Technical Report 97-3 Department of Computer Science, Rensselaer Polytenchic Institute, Troy, NY 12180.

[72] S. Mann. An historical account of the 'wearcomp' and 'wearcam' inventions developed for applications in 'personal imaging'. In *IEEE Proceedings of the first ISWC*, pages 66–73, Cambridge, MA, October 13-14 1997.

[73] S. Mann. Smart clothing: The wearable computer and wearcam. *Personal Technologies*, 1(1):21–27, 1997.

[74] R. Meo, G. Psaila, and S. Ceri. A new SQL-like operator for mining association rules. In *22nd Intl. Conf. Very Large Databases*, 1996.

[75] S. A. Moyer and V. S. Sunderam. PIOUS: a scalable parallel I/O system for distributed computing environments. In *Scalable High-Performance Computing Conference*, 1994.

[76] N. Nieuwejaar and D. Kotz. The galley parallel file system. *Parallel Computing*, 23(4), June 1997.

[77] M. T. Oszu and P. Valduriez. *Principles of Distributed Database Systems*. Prentice Hall, 2 edition, 1999.

[78] R. H. Patterson III. *Informed Prefetching and Caching*. PhD thesis, School of Computer Science, Carnegie Mellon University, December 1997.

[79] H. Pirahesh, C. Mohan, J. Cheng, T. Liu, and P. Selinger. Parallelism in relational data base systems: Architectural issues and design approaches. In *2nd Int'l Symp. on databases in parallel and distributed systems*, july 1990.

[80] D. A. Reed, R. A. Aydt, L. DeRose, C. L. Mendes, R. L. Ribler, E. Shaffer, H. Simitci, J. S. Vetter, D. R. Wells, S. Whitmore, and Y. Zhang. Performance analysis of parallel systems: Approaches and open problems. In *Proceedings of the Joint Symposium on Parallel Processing (JSPP)*, pages 239–256, Nagoya, Japan, June 1998. Invited paper and keynote presentation.

[81] B. J. Rhodes, N. Minar, and J. Weaver. Wearable computing meets ubiquitous computing reaping the best of both worlds. In *Proceedings of The Third International Symposium on Wearable Computers (ISWC '99)*, pages 141–149, San Francisco, CA, October 18-19 1999. IEEE.

[82] E. Riedel, G. A. Gibson, and C. Faloutsos. Active storage for large-scale data mining and multimedia. In *Proceedings of the 24th international Conference on Very Large Databases (VLDB '98)*, New York, NY, August 1997.

[83] G. F. Riley, R. M. Fujimoto, and M. H. Ammar. A generic framework for parallelization of network simulations. In *Proceedings of Mascots 1999*, October 1999.

[84] H. N. S. Goil and A. Choudhary. MAFIA: Efficient and scalable subspace clustering for very large data sets. Technical Report 9906-010, Center for Parallel and Distributed Computing, Northwestern University, jun 1999.

[85] S. Sarawagi, S. Thomas, and R. Agrawal. Integrating association rule mining with databases: alternatives and implications. In *Proc. of the ACM SIGMOD Int'l Conference on Management of Data*, Seattle, Washington, June 1998. Expanded version available as IBM Research Report RJ 10107 (91923), March, 1998.

[86] M. Satyanarayanan. *Distributed Systems*, chapter Distributed File Systems, pages 353–381. Addison Wesley, 1993.

[87] Scalable I/O Initiative. *http://www.cacr.caltech.edu/SIO*. California Institute of Technology.

[88] J. Schafer, J. Konstan, and J. Riedl. Electronic commerce recommender applications. *Data Mining and Knowledge Discovery*, January 2001.

[89] E. Schikuta, T. Fuerle, and H. Wanek. ViPIOS: The vienna parallel input/output system. In *Euro-Par'98*, Sept. 1998.

[90] T. Schmitt and N. Lehman. Non-unity molecular heritability demonstrated by continuous evolution *in vitro*. *Chemistry and Biology*, 6(12):857–869, December 1999.

[91] K. E. Seamons and M. Winslett. Multidimensional array I/O in Panda 1.0. *Journal of Supercomputing*, 10(2):191–211, 1996.

[92] J. Shafer, R. Agrawal, and M. Mehta. Sprint: A scalable parallel classifier for data mining. In *22nd VLDB Conference*, Mar. 1996.

[93] T. Shintani and M. Kitsuregawa. Mining algorithms for sequential patterns in parallel: Hash based approach. In *2nd Pacific-Asia Conf. on Knowledge Discovery and Data Mining*, Apr. 1998.

[94] A. Siebes. Foundations of an inductive query language. In *1st Intl. Conf. on Knowledge Discovery in Databases and Data Mining*, Aug. 1995.

[95] M. Sirbu and J. D. Tygar. Netbill: An internet commerce system optimized for network-delivered systems. *IEEE Personal Communications*, 2(4):20–25, August 1995.

[96] D. Skillicorn. Strategies for parallel data mining. *IEEE Concurrency*, 7(4):26–35, October-December 1999.

[97] M. Sreenivas, K. Alsabti, and S. Ranka. Parallel out-of-core divide and conquer techniques with application to classification trees. In *13th International Parallel Processing Symposium*, Apr. 1999.

[98] H. Stockinger. Dictionary on parallel input/output. Master's thesis, Dept. of Data Engineering, University of Vienna, February 1998. A detailed reference of David Kotz's Parallel I/O Bibliography.

[99] T. Strzalkowski, J. Wang, and B. Wise. A robust practical text summarization. In *Proceedings of the AAAI Symposium on Intelligent Text Summarization*, pages 26–33, Stanford University, Stanford, California, March 1998. American Association for Artificial Intelligence.

[100] B. K. Szymanski. An upper bound for a time step in parallel spatially explicit biological simulations. In *System Analysis, Modeling, Simulation*, volume 18-19, pages 717–720, Berlin, June 1995. special issue with *Proc. 5th IMACS Symposium on System Analysis and Simulation*.

[101] B. K. Szymanski and T. Caraco. Spatial analysis of vector-borne disease: A four species model. *Evolutionary Ecology*, 8:299–314, 1994.

[102] Tandem Performance Group. A benchmark of non-stop SQL on the debit credit transaction. In *SIGMOD Conference*, June 1988.

[103] R. Thakur, W. Gropp, and E. Lusk. On implementing mpi-io portably and with high performance. In *Proceedings of the Sixth Workshop on I/O in Parallel and Distributed Systemm IOPADS '99*, Atlanta, GA, May 1999.

[104] P. Valduriez. Parallel database systems: Open problems and new issues. *Distributed and Parallel Databases*, 1:137–165, 1993.

[105] M. Weiser. Hot topics: Ubiquitous computing. *IEEE Computer*, pages 71–72, 1993.

[106] M. Weiser. Some computer science problems in ubiquitous computing. *CACM*, pages 137–143, July 1993.

[107] G. Williams, I. Altas, S. Bakin, P. Christen, M. Hegland, A. Marquez, P. Milne, R. Nagappan, and S. Roberts. The integrated delivery of large-scale data mining: The ACSys data mining project. In [115].

[108] Wireless Application Protocol Forum, Ltd., On Line at: `http://www.wapforum.org/what/technical.htm`. *Wireless Application Protocol, Architecture Specification*, version 30-apr-1998 edition, April 1998. Also called WAP-Architecture.

[109] Wireless Application Protocol Forum, Ltd., On Line at: `http://www.wapforum.org/what/technical.htm`. *Wireless Application Protocol, Wireless Markup Language Specification*, version 1.3 edition, February 19 2000. Also called WAP-191-WML.

[110] Wireless Application Protocol Forum, Ltd., On Line at: `http://www.wapforum.org/what/technical.htm`. *Wireless Application Protocol, WMLScript Language Specification*, version 1.2 edition, June 2000. Also called WAP-193-WMLScript Language Specification.

[111] Wireless Application Protocol Forum, Ltd., On Line at: `http://www.wapforum.org/what/whitepapers.htm`. *Wireless Internet Today*, June 2000. Wireless Application Protocol White Paper.

[112] M. Wright and G. Joyce. Continuous in vitro evolution of catalytic function. *Science*, 276:614, 1997.

[113] M. J. Zaki. Parallel and distributed association mining: A survey. *IEEE Concurrency*, 7(4):14–25, October-December 1999. Special issue on Parallel Data Mining.

[114] M. J. Zaki. Parallel sequence mining on SMP machines. In [115].

[115] M. J. Zaki and C.-T. Ho, editors. *Large-Scale Parallel Data Mining*, volume 1759 of *LNCS/LNAI*. Springer-Verlag, Heidelberg, Germany, 2000.

[116] M. J. Zaki, C.-T. Ho, and R. Agrawal. Parallel classification for data mining on shared-memory multiprocessors. In *15th IEEE Intl. Conf. on Data Engineering*, Mar. 1999.

[117] M. J. Zaki, S. Jin, and C. Bystroff. Mining residue contacts in proteins using local structure predictions. In *IEEE International Symposium on Bioinformatics and Biomedical Engineering*, pages 168–175, Washington, DC, November 2000. IEEE.

[118] M. J. Zaki, S. Parthasarathy, M. Ogihara, and W. Li. Parallel algorithms for discovery of association rules. *Data Mining and Knowledge Discovery: An International Journal*, pages 343–373, 1997. special issue on Scalable High-Performance Computing for KDD.