Research Statement
Shivnath Babu
Stanford University

Database Management Systems (DBMSs) have played a central role in information technology over the last three decades. While DBMSs continue to remain highly relevant for new and emerging application domains, certain fundamental assumptions underlying DBMS architectures do not hold in some of these domains. My research in general addresses this interesting and challenging problem.

Data Stream Management Systems
Many modern applications deal with data that is updated continuously and must be processed in real-time: network performance monitoring, financial analysis over stock tickers, sensor data processing, and others. Data Stream Management Systems (DSMSs) have recently emerged for this class of applications: In addition to managing traditional stored data such as relations, a DSMS handles multiple continuous, unbounded, possibly rapid and time-varying data streams; citations [8,23] in curriculum vita. The STREAM project at Stanford is building a comprehensive prototype DSMS. I have participated in this project since its inception and made the following contributions to date.

Specifying Continuous Queries: Due to the continuous unbounded nature of data streams, a DSMS typically supports long-running continuous queries, which are expected to produce answers in a continuous and timely fashion [23]. I was one of the two primary Ph.D. students in the design and implementation of CQL, a rich declarative query language for specifying continuous queries in a DSMS [10]. This work was among the first to define a concrete semantics for arbitrarily complex continuous queries.

Adaptive Query Processing Engine: A fundamental challenge faced by DSMSs is that stream data and arrival characteristics as well as system conditions may vary significantly over the lifetime of a continuous query. Conventional DBMS query processing architectures are not designed to deal with changes in input characteristics while a query is running, so their performance may drop drastically over time as characteristics change. I designed and implemented StreaMon, a new adaptive query processing architecture for DSMSs [15]. StreaMon's overall approach is to generate a straightforward initial execution plan for a given CQL query, which then adapts automatically to a better initial plan, and continues to adapt as input characteristics change. StreaMon supports efficient adaptive algorithms with performance guarantees for two important classes of CQL queries: commutative filters over a single stream [3,4] and multiway stream joins [1]. Furthermore, StreaMon monitors stream properties (e.g., sortedness) and exploits them to reduce run-time memory requirements for execution plans [12].

Multi-objective Scheduler: The scheduling strategy for the operators in a DSMS query plan can have a significant impact on the system's peak resource usage as well as its ability to produce results in a timely fashion. I participated in the design and implementation of Chain, a multi-objective operator scheduling strategy that minimizes run-time memory requirements for a large class of execution plans, while guaranteeing that answers are generated within required latency limits [5,11].

System Prototype: I was one of the lead architects of STREAM's comprehensive prototype DSMS [6,16,24]. The system fully supports CQL queries and StreaMon execution strategies, along with many other features. It was recently released publicly and is gaining outside users.

Adaptive Query Processing in Database Management Systems
In a separate but somewhat related research thrust, in collaboration with a University of Wisconsin graduate student I developed new adaptive query processing techniques to improve overall performance in a conven-
tional DBMS [25,26]. DBMSs choose efficient execution plans for queries based on statistical estimates of the input data characteristics. In practice, these estimates are often erroneous, leading to suboptimal plan choices. To address this problem, we developed a new approach that takes the confidence in statistical estimates into account to choose execution plans that are less sensitive to errors in estimates, as well as plans where execution choices can be delayed until accurate statistics can be collected at run-time. Using an open-source DBMS, we implemented our new adaptive techniques as part of an overall improved query optimization strategy [26]. We obtained significant performance gains over previous optimization techniques, including adaptive ones.

Future Research Agenda

My long term research goal is to make data management systems efficient and self-manageable for new usage scenarios. I plan to take the following initial steps in this direction:

**New Steps in DSMSs:** While first-generation DSMS prototypes deliver on some of their original promises, plenty of work remains to be done. For example, STREAM’s prototype DSMS handles complex continuous queries on fast streams by restricting all data accesses to main memory. I plan to develop new stream processing techniques that generate predominantly sequential data access patterns, and use them in conjunction with high-bandwidth disk subsystems. As another example, current DSMS prototypes cannot run complex continuous queries on streams that arrive at modern high-end network speeds. I plan to explore how the similarities between stream processing in a DSMS and graphics processing can be exploited to execute stream processing primitives efficiently on modern graphics processing units that have significantly higher memory bandwidth and parallelism compared to general-purpose processors.

**New Steps in Adaptive Processing:** Adaptive processing is integral to the current Computer-Science-wide push towards *autonomic computing*, which aims to minimize human effort in running systems efficiently. I intend to explore whether the adaptive techniques that I have developed in StreaMon can benefit other fields where adaptive processing is being used today, e.g., in computer architecture to configure computational units on a chip dynamically based on application needs, and in distributed systems to allocate hardware resources dynamically in server farms. Within databases, it is challenging to scale adaptive engines to handle complex queries and large *plan spaces*—e.g., to simultaneously consider data access methods, plan shapes, memory allocation to plans, parallelism, and sharing of data and computation—because the run-time overhead to support adaptive processing increases significantly as plan spaces expand. I plan to apply a new overall approach to address this problem, where plan choices made for execution under different statistical conditions are logged over time [2]. One of the many uses of this technique is to enable efficient plans to be chosen quickly whenever statistical conditions change in a periodic manner.

**New Usage Scenarios for DBMSs:** DBMSs are being used increasingly in new application domains, e.g., involving remote web-services, thousands of tiny sensors in an ad-hoc network, or grid-based scientific analysis. Many of these domains challenge some of the basic assumptions underlying DBMS technology. Overall, I am interested in how DBMS technology must evolve—or be completely reconsidered—when basic assumptions are broken. As just one example, with web-services there may be no useful prediction of data statistics to perform query planning, and with sensors the data may be heavily error-prone or uncertain. As a first step in handling these types of applications, I hope to develop query processing techniques targeted at reducing uncertainty in statistics and data as quickly as possible, even at the expense of early answers—an objective not considered in more traditional application domains [2].