Monitoring Network Routing and Traffic with Low Space MIT2001-09

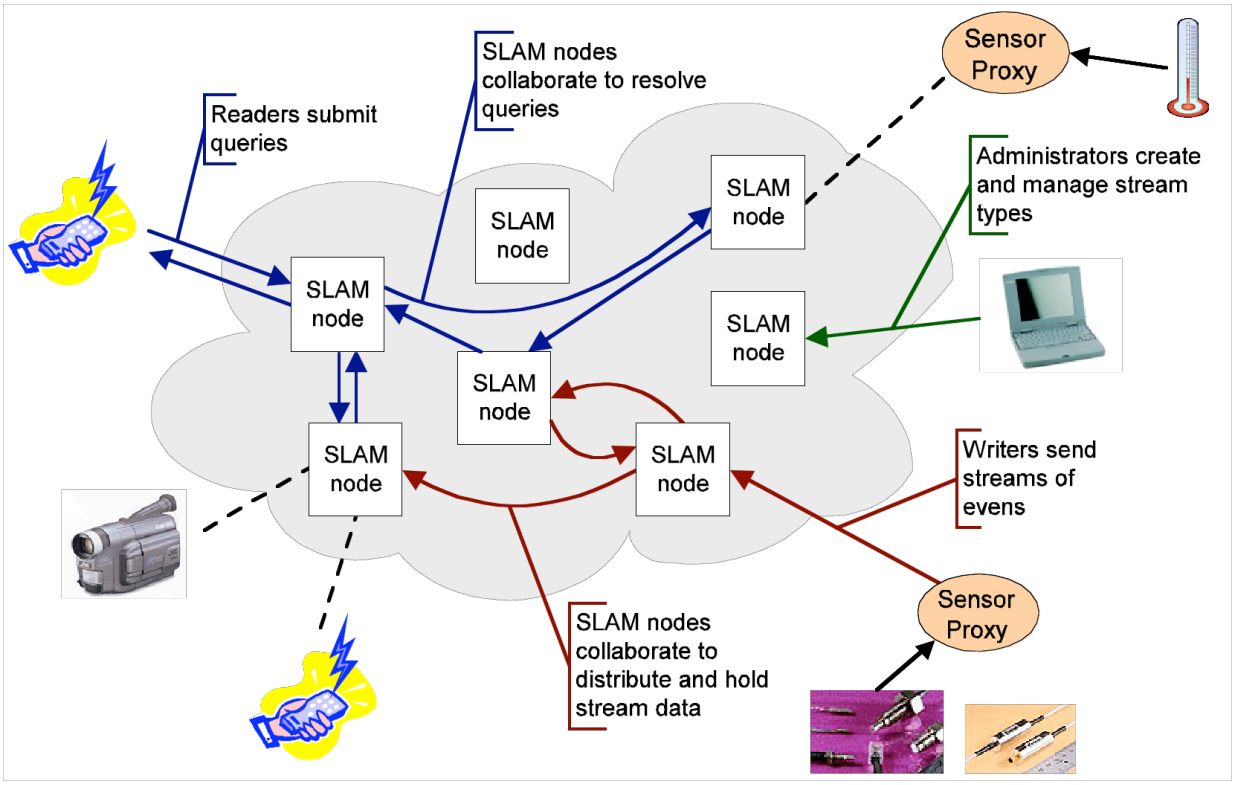
Progress Report: July 1, 2002—December 31, 2002

Erik D. Demaine

Project Overview

SLAM (Scalable Location-Aware Monitoring) is a new collaborative effort between MIT professors in multiple areas—Hari Balakrishnan (networks), Erik Demaine (algorithms), Michael Stonebraker (databases), and Seth Teller (graphics/vision). The goal of the SLAM project is to build a scalable location-aware computational network architecture integrating millions of real-world sensors with actuators and software applications. SLAM will enable a broad variety of novel monitoring and control applications including rapid disaster response, traffic management, network monitoring, and many others. A key component of SLAM is a peer-to-peer sensor network, called ***Medusa***, which will (a) route data streams from sensors and sensor proxies to nodes that need the data, and (b) resolve stream-style database queries on-the-fly by distributed computation. Medusa will support three main types of queries:

- *persistent queries*, which ask about current data and whose answer requires constant updates (e.g., notify me if the average temperature in any room goes beyond a threshold, or if my laptop leaves campus);
- *ad-hoc queries*, which ask similar questions but require only a single answer (e.g., where are my books about geometry currently?);
- *historical queries*, which ask questions about past recorded data (e.g., where have I been in the last hour that I stayed for at least 5 minutes (and might have dropped my keys)?).



The principle challenge is formed by persistent queries. In this setting, the roles in the database are completely reversed: in a traditional database, data is relatively static and queries come online and pass through the database; while in Medusa, the queries are relatively static and data comes online and passes through the queries. This reversal of roles is an essential difference between traditional databases and stream-oriented databases, and means that Medusa must face many new research challenges.

Progress Through December 2002

The Medusa project has matured to involve 4 faculty, 1 research staff member, 7 graduate students, and 1 undergraduate student. We have completed a high-level Medusa system architecture design, and have already drafted a document describing this design. We have also built a preliminary implementation of this architecture, and can use the implementation for basic Medusa functionality. One application which already runs is an RFID tracking application, which labels a graphical representation of the environment with locations of recently seen RFID (Radio-Frequency Identification) tags.

In addition, we have initiated a productive collaboration with the Brown University database group and their ***Aurora*** project. Together, we are designing a powerful framework for stream-based database processing, and implementing a single-node engine for processing these queries. This single-node engine will form the core of a single Medusa node; the rest of the Medusa system connects these cores together to form an efficient, scalable, and fault-tolerant distributed database.

Another important advance that we made most recently is the design of an economic model in which multiple Medusa networks can cooperate to share data, load, storage, etc. safely and fairly. One key idea is to assign each computational "box" (a persistent procedure with input and output stream(s)) a particular *evaluation cost* per input, which can be negotiated between networks. Such a box can be moved to another Medusa network, but the originating network must pay the corresponding amount for balance. The resulting "cash" can be used for moving boxes in the other direction at a later time. The idea is that this ability for load-balancing allows for more efficient use of a multinetwork universe, where some networks may be more heavily loaded than others, and this distribution may change over time. Nodes *bid* on the price for implementing boxes so that the most effective assignment of boxes to nodes can be found, and nodes aim to earn the maximum amount of cash.

Research Plan for the Next Six Months

One main algorithmic problem that has arisen in this work is to design an efficient *load-balancing* algorithm for dynamically distributing load (by way of the "boxes" described above) among nodes within a single Medusa networks and across multiple Medusa networks. Here there are two main factors to take into account: the relative loads of various machines, particularly when one machine is above a maximum load threshold; and the cost of moving a box from one node to another. The box-moving cost can be substantial, possibly involving the transference of all relevant state, historical data, code, and active query processing. In particular, we must avoid constant oscillation of boxes between two or more nodes when the network remains in a steady state; only when load varies substantially is it worthwhile moving boxes. Another key challenge is that nodes cannot afford to communicate with all other nodes; instead, each node can communicate only with its neighbors (nodes nearby in the physical network), and yet still must guarantee eventually balanced load throughout the network in a steady state.

In addition, we are currently working on extending the preliminary Medusa implementation, and completion and incorporation of the Aurora engine, to build a fully functional Medusa prototype. We will also build a stress-test benchmark, involving a realistic highway traffic simulator inspired by a recent highway-charging mechanism introduced in California. The goal is for the benchmark to be sufficiently stressful that traditional database systems, such as Oracle, will grind to a halt, proving the need for distributed stream-based databases. Then we will evaluate our Medusa implementation on this benchmark; we expect at least one order of magnitude in speedup, making previously impractical applications a reality.