Processing High-Speed Intelligence Feeds in Real-Time

Alan Demers, Johannes Gehrke, Mingsheng Hong, and Mirek Riedewald

Department of Computer Science, Cornell University
{ademers, johannes, mshong, mirek}@cs.cornell.edu

Intelligence organizations face the daunting task of collecting all relevant pieces of information and to draw conclusions about potential threats in a timely manner. Typical information sources range from news tickers, financial transaction logs and message logs to satellite images and speech recordings. This wealth of data is continuously updated and arrives in high-speed data streams; it needs to be analyzed both in real-time (e.g., to estimate the importance of the information and to generate early threat alerts) and offline by sophisticated data mining tools. This work focuses on the real-time aspects of processing these massive streams of intelligence data. We also show how real-time and data mining components can interact effectively.

Current database and data warehousing technology supports expressive queries, but is not designed for real-time processing of data streams [1]. At the other end of the spectrum are publish/subscribe (pub/sub) systems. They can support very high throughput even for millions of active subscriptions [2], but have only limited query languages. It is not possible to search for patterns involving multiple incoming messages and across streams.

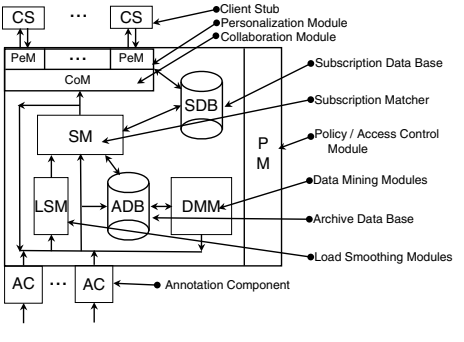

Fig. 1. Cornell Knowledge Broker

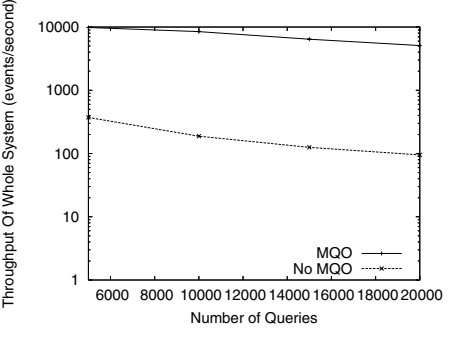

Fig. 2. Throughput vs. number of active monitoring queries

We are building the Cornell Knowledge Broker (CKB), a system for processing high-speed streams of intelligence data. The architecture of the CKB provides a unique combination of new technology for data stream processing, data mining, and privacy/access control (see Figure 1). The *Subscription Matcher (SM)*

P. Kantor et al. (Eds.): ISI 2005, LNCS 3495, pp. 617–618, 2005.

component lies at the heart of the CKB.[1] The SM module manages expressive monitoring queries, which are continuously evaluated for all incoming messages. As soon as a relevant pattern is detected, it generates an alert message, which in turn is either processed by another query or directly forwarded to an analyst. We have developed a powerful query language, which enables analysts to search for complex patterns in the incoming data.

Let us illustrate the functionality of the SM module through one example query: "Notify me whenever Alice passes sensitive information to Bob, either directly or through several intermediate parties". The SM module monitors all incoming messages, building up internal data structures that capture the relevant network of information flow. As soon as a path with sensitive information flow between Alice and Bob has been established, an analyst is notified. The sensitivity of a message could be determined based on simple keywords or by using a sophisticated text classification approach [4]. Note that it is easy to see through this query that the expressiveness of the SM module goes far beyond the simple predicate filtering of current pub/sub technology. The SM module also supports queries that can search for patterns like a sequence of phenomena that indicate the outbreak of an epidemic and many others.

We have built a running prototype of the SM module, which can effectively interact with text analysis modules. The system has an easy-to-learn user interface, enabling users to formulate sophisticated monitoring queries in a high-level language. Our system automatically detects and exploits commonalities between different monitoring queries. Figure 2 shows the throughput (number of relevant events or messages per second) with varying number of concurrently active monitoring queries. Notice the logarithmic scale of the y-axis. The MQO curve is for our system, while "no MQO" is for a traditional implementation where the query functionality is located in an application server.

References

1. Carney, D., Çetintemel, U., Cherniack, M., Convey, C., Lee, S., Seidman, G., Stonebraker, M., Tatbul, N., Zdonik, S.: Monitoring streams — a new class of data management applications. In: Proc. Int. Conf. on Very Large Databases (VLDB). (2002)
2. Fabret, F., Jacobsen, H.A., Llirbat, F., Pereira, J., Ross, K.A., Shasha, D.: Filtering algorithms and implementation for very fast publish/subscribe. In: Proc. ACM SIGMOD Int. Conf. on Management of Data. (2001) 115–126
3. Demers, A., Gehrke, J., Riedewald, M.: The architecture of the cornell knowledge broker. In: Proc. Second Symposium on Intelligence and Security Informatics (ISI-2004). (2004)
4. Joachims, T.: Learning to Classify Text Using Support Vector Machines – Methods, Theory, and Algorithms. Kluwer (2002)

[1] The functionality of the other components is indicated by their names, details can be found in our previous work [3].