Dataflow Plan Execution for Software Agents

Greg Barish, Daniel DiPasquo, Craig A. Knoblock, and Steven Minton
Information Sciences Institute, Integrated Media Systems Center,
and Department of Computer Science
University of Southern California
4676 Admiralty Way, Marina del Rey, CA 90292
{barish, dipasquo, knoblock, minton}@isi.edu

ABSTRACT

Recent research has made it possible to build information agents that retrieve and integrate information from the World Wide Web. Although there now exist solutions for modeling Web sources, query planning, and information extraction, less attention has been given to the problem of optimizing agent execution. In this paper, we describe Theseus, an efficient agent plan execution system. Through its pipelined, dataflow-based architecture, Theseus offers a high degree of parallelism and asynchronous information routing, features that can substantially improve performance. Theseus differs from prior work in reactive planning systems and parallel databases because it gathers information from the Web, a domain where information retrieval is a problem that is network-bound and is often based on interleaved data gathering and navigation. The Theseus plan language and architecture directly address these issues, resulting in a high-performance execution system.

1. INTRODUCTION

Gathering information from the World Wide Web is a research problem that has received substantial attention in recent years. There now exist a number of systems [7, 10, 13] and approaches towards automating this process, including work on data extraction [11, 14], query planning [1, 12], data materialization [2], and methods for handling data inconsistency [4]. Today, it is possible to construct useful agents that rely on these technologies as tools to perform automatic and intelligent data integration [3].

Although these individual technologies may each be efficient, overall end-to-end agent execution performance is often less than optimal. This is primarily because Web-based data integration is a process that is network-bound and masks the efficiencies of individual technologies (such as data extraction). Complicating matters is the fact that building useful agents often requires larger, more complex plans. For example, consider how people commonly use the Web to locate houses for sale, which meet a particular set of criteria (*e.g.*, price and location). This process means more than simply executing a particular query once and then returning a long list of data. More often, searching for a house means executing the same or similar queries periodically, say on a daily basis, over the course of a few weeks or months. Furthermore, a "useful" search process means gathering only new or updated listings (meeting the specified criteria) for each query execution. Users are rarely interested in being reminded of houses about which they have already been notified. Also, with the explosive growth in mobile networking, there are many users who would prefer to have their query results distributed through different messaging means (*i.e.*, e-mail, cellular phone, fax) and reported using a variety of formats (*i.e.*, XML, HTML, WML, text, voice). Finally, in addition to message notification, it is often desirable to have newly gathered information trigger a variety of other actions. For example, if a search for a house yields a result, a user may want to immediately send an automated e-mail to the corresponding real-estate agent suggesting a meeting time (based on the user's personal schedule, also kept online).

Thus, while gathering data is unquestionably an important task, there are also challenges related to useful processing of this data. We believe that information gathering is a piece of a larger puzzle called *information management*, a problem that involves conditional plan execution, continuous querying, query result accumulation, local persistent storage, and the linking of other actions to the results of queries. This problem encompasses issues that are at the heart of how users query the Web today to retrieve meaningful information and the way such data is put to practical use. Searching for a new house is merely one type of application. There are numerous other instances where such automation is not only useful, but perhaps essential: newswire tracking, online auction participation, and stock/portfolio management, to name a few. In these scenarios, users want more than to just query and retrieve data once - they want to be able to monitor Web sites. The dynamic nature of the Internet invites this approach. However, a means for building high-performance agents for this type of information management remains a relatively open issue.

2. THESEUS

This paper outlines our approach to efficient information management in the presence of heterogeneous and distributed databases. In particular, our approach benefits from combining features found in both parallel databases and general plan execution systems, marrying the efficiency found in former with the generality and flexibility found in the latter. Parallel database research [5, 8, 17] has shown that it is possible to build highly efficient query execution systems for local databases. Complementarily, existing plan execution systems [6, 15] have proven to be more generally applicable to a wide-range of planning problems and often provide more flexibility in terms of plan control flow (*i.e.*, support for loops and conditionals). Motivated by this research, we have implemented the combination of these features in the Theseus agent plan execution system. Based on a dataflow processing architecture, Theseus is an approach well-suited for efficient information integration and management of Internet data sources.

The Theseus plan language allows plans to be specified as a network of *operators* connected through sets of *enablements*. Operators can be thought of as finite state machines that, when enabled, perform a specific type of information management action. Enablements are similar to planning pre/post-conditions, except that they can also carry data, thus allowing information to be easily routed between operators. Declaring plans with operators and enablements allows execution to be specified in terms of those control and data dependencies necessary to ensure correct execution. Theseus operators include those useful for data processing, remote information retrieval, local storage (*i.e.*, in a local relational database), and those for flexible communication of plan results (*i.e.*, via e-mail). Leveraging these operators and built-in support for loops and conditionals allows powerful, practical information management plans to be specified.

The Theseus execution engine is designed to function like a hybrid dataflow machine [16]. The availability of enablements determines when various operators execute. Thus, as is the case with most dataflow systems, parallelism and synchronization are realized automatically. Operators exist as threads, so Theseus can theoretically achieve as much true parallelism as there are CPUs. The execution system also supports *pipelining*, in which producer operators asynchronously propagate enablements to consumer operator queues. Without the synchronization overhead, opportunities for parallel execution are increased.

Through its language and execution system, Theseus enables agents to perform useful information management tasks, such as periodic execution, query result accumulation, and flexible result communication. Most importantly, through properties of its architecture, Theseus reduces the overall effect of network latencies on data integration, providing increased parallelism and asynchrony during execution so that the overall end-to-end agent execution process is substantially faster.

Theseus has evolved from research related to the Ariadne [10] information mediator project. Ariadne enabled the integration of multiple heterogeneous data sources, so that the combined data can be accessed from a single, logical model. We believe Theseus is a logical next step: it builds on integration, allowing users to do something useful with information that is gathered.

3. RELATED WORK

As described earlier, Theseus can be viewed as a cross between general plan executors and parallel database systems. The key differences are that (a) unlike general plan executors, Theseus is optimized for the information processing domain and that (b) unlike parallel databases, standard techniques for achieving high-performance (such as the shared-nothing approach) are simply not applicable to information management on the Internet, which consists of heterogeneous and distributed data sources, beyond the administrative domain of the execution engine.

Theseus can also be compared with Tukwila [9], which supports efficient query execution on remote, heterogeneous data sources. Like Theseus, Tukwila is interested in data integration, especially the ability to gather information from web sites as if they were databases. The main difference between Theseus and Tukwila is that Theseus uses a hybrid dataflow model of execution while Tukwila uses standard (von-Neumann) control flow model. Key in understanding this difference is in the tradeoffs between dataflow and control flow systems. The former allows implicit, on-demand parallelism with minimal synchronization penalty, while the latter manages parallelism manually, usually with higher synchronization overhead.

4. CONCLUSION

We have provided an overview of Theseus and how it is a useful approach for building efficient information agents that can gather and manage data from the Web. Because our planning language allows complex plans for managing information to be easily expressed, users can build powerful agents. Furthermore, since the execution system described is based on a dataflow paradigm, and operates with substantial asynchrony and parallelism, these agents can realize a high level of performance.

5. ACKNOWLEDGEMENTS

This work was supported in part by the United States Air Force under contract number F49620-98-1-0046, by the Rome Laboratory of the Air Force Systems Command and the Defense Advanced Research Projects Agency (DARPA) under contract number F30602-98-2-0109, and by the Integrated Media Systems Center (an NSF Engineering Research Center). The views and conclusions contained in this article are the authors' and should not be interpreted as representing the official opinion or policy of any of the above organizations or any person connected with them.

6. REFERENCES

[1] Ambite, J.L. and Knoblock, C.A. 1997. Planning by Rewriting: Efficiently Generating High-Quality Plans. *AAAI-1997.*

[2] Ashish, N.; Knoblock, C.A.; and Shahabi, C. 1999. Selective materializing data in mediators by analyzing user queries. *COOPIS-99*

[3] Barish, G.; Knoblock, C.A.; Chen, Y-S.; Minton, S.; Philpot, A.; Shahabi, C. 1999. TheaterLoc: A Case Study in Information Integration. *IJCAI-99 Workshop on Information Integration*.

[4] Cohen, W. W. 1998. Integration of Heterogeneous Databases Without Common Domains Using Queries Based on Textual Similarity. *SIGMOD-1998.*

[5] DeWitt D.J. and Gray, J. 1992. Parallel Database Systems: The Future of High Performance Database Systems. *Comm of ACM 35(6).*

[6] Firby, R.J. 1994. Task Networks for Controlling Continuous Processes. *Proceedings of the 2nd Intl Conf on AI Planning Systems*.

[7] Genesereth, M.R.; Keller, A.M.; and Duschka, O.M. 1997. Infomaster: An information integration system. *SIGMOD-1997.*

[8] Graefe, G. 1994. Volcano-An Extensible and Parallel Query Evaluation System. *IEEE Trans Knowledge Data Engineering 6(1).*

[9] Ives, Z; Florescu, D.; Friedman, M.; Levy, A.; Weld , D. 1999. An Adaptive Query Execution Engine for Data Integration. *SIGMOD-99*.

[10] Knoblock, C.A.; Minton, S; Ambite, J.L.; Ashish, N.; Modi, J.; Muslea, I.; Philpot, A. and Tejada, S. 1998. Modeling Web Sources for Information Integration. *AAAI-1998*.

[11] Kushmerick, N. 1997. *Wrapper Induction for Information Extraction*. PhD Thesis, Computer Science Dept. University of Washington.

[12] Kwok, C.T and Weld, D.S. 1996. Planning to gather information. *AAAI-1996.*

[13] Levy, A.Y.;Rajaraman, A; Ordille, J.J. 1996. Querying Heterogeneous Information Sources Using Source Descriptions. *VLDB 1996*.

[14] Muslea, I.; Minton, S.; and Knoblock, C.A. 1998. STALKER: Learning Extraction Rules for Semistructured, Web-based Information Sources. *AAAI-98 AI & Information Integration Wkshp*

[15] Myers, K. 1996. A Procedural Knowledge Approach to Task-Level Control. In *Proc of the Third Intl Conf on AI Planning Systems*.

[16] Papadopolous, G.M. and Traub, K.R. 1991. Multithreading: A revisionist view of dataflow architectures. *Proc 18th Intl Symp on Computer Architecture.*

[17] Wilschut, A.N. and Alpers, P.M.G. 1991. Dataflow query execution in a main memory environment. *Proc of 1st PDIS Conference*.