SAND2005-5600
Unlimited Release
Printed September 2005

Algorithms and Architectures for High Performance Analysis of Semantic Graphs

Bruce Hendrickson

Sandia National Labs

Semantic graphs offer one promising avenue for intelligence analysis in homeland security. They provide a mechanism for describing a wide variety of relationships between entities of potential interest. The vertices are nouns of various types, e.g. people, organizations, events, etc. Edges in the graph represent different types of relationships between entities, e.g. "is friends with", "belongs-to", etc. Semantic graphs offer a number of potential advantages as a knowledge representation system.

- They allow information of different kinds, and collected in differing ways, to be combined in a seamless manner.
- A semantic graph is a very compressed representation of some of relationship information. It has been reported that the semantic graph can be two orders of magnitude smaller than the processed intelligence data. This allows for much larger portions of the data universe to be resident in computer memory.
- Many intelligence queries that are relevant to the terrorist threat are naturally expressed in the language of semantic graphs. One example is the search for "interesting" relationships between two individuals or between an individual and an event, which can be phrased as a search for short paths in the graph. Another example is the search for an analyst-specified threat pattern, which can be cast as an instance of subgraph isomorphism. It is important to note than many kinds of analysis are not relationship based, so these are not good candidates for semantic graphs. Thus, a semantic graph should always be used in conjunction with traditional knowledge representation and interface methods.
- Operations that involve looking for chains of relationships (e.g. friend of a friend) are not efficiently executable in a traditional relational database. However, the semantic graph can be thought of as a pre-join of the database, and it is ideally suited for these kinds of operations.

Researchers at Sandia National Laboratories are working to facilitate semantic graph analysis. Since intelligence datasets can be extremely large, the focus of this work is on the use of parallel computers. We have been working to develop scalable parallel algorithms that will be at the core of a semantic graph analysis infrastructure. Our work has involved two different thrusts, corresponding to two different computer architectures.

The first architecture of interest is distributed memory, message passing computers. These machines are ubiquitous and affordable, but they are challenging targets for graph algorithms. Much of our distributed-memory work to date has been collaborative with researchers at Lawrence Livermore National Laboratory and has focused on finding short paths on distributed memory parallel machines. Our implementation on 32K processors of BlueGene/Light finds shortest paths between two specified vertices in just over a second for random graphs with 4 billion vertices.

SAND2005-5600
Unlimited Release
Printed September 2005
We are currently working to extend these initial efforts in three directions, each involving parallel implementation on distributed memory computers. First, we will build upon the short paths work to find what the literature calls “connection subgraphs” – a subgraph that best captures the salient relationships between two entities. Note that the paths in such a graph will generally be short, but not necessarily the shortest paths. This is because a path through a very high degree vertex (e.g. “is a citizen of the U.S.”) is likely to be less important than a longer path through lower degree vertices. Second, we will finish our work on parallel partitioning of semantic graphs by merging with the two-dimensional hypergraph partitioning work underway at Sandia. Third, we will implement algorithms for subgraph isomorphism.

This distributed memory algorithm development is complemented by another project that has the same objectives, but is focused on a different parallel computer architecture. Graph algorithms are generally dominated by memory access time. Most operations involve following links in the graph, and are not computationally intensive. Highly unstructured semantic graph operations cannot benefit from caches and other multilevel memory features of contemporary processors. Hence, most memory accesses are to the slowest, most distant memory layer, and the processor is starved for data. The latency of the most distant memory determines the performance of the processor, and caching operations are a non-beneficial overhead. In recent graph benchmarks, a novel architecture dramatically outperformed traditional processors. The Cray/Tera MTA was designed to tolerate latency through massive multithreading. Specifically, an MTA processor has 128 streams, each capable having of up to 8 outstanding memory requests. Context switches between streams take single clock cycle. In this way, memory accesses are parallelized and the processor can be kept busy. Although the clock rate of the MTA processor is an order of magnitude slower than current commodity microprocessors, it executes graph algorithms much faster than they can.

Besides faster single processor performance, massive multithreading offers a number of additional advantages for semantic graph analysis. First, the MTA supports a global address space with uniform memory access. This greatly simplifies the programming model. With the exception of a few special language features, parallel MTA software runs on serial workstations. The complexity of explicit message passing and memory management is absent in the programming model. Parallelism is loop based, often automatically identified by the compiler, and customizable via simple programmer specified pragmas. Second, load balancing is comparatively easy on the machine. The mixture of global shared memory and light-weight threads allow the operating system to assign new tasks to underutilized processors. Third, and closely related, streams of jobs are easily handled via assignment to idle processors. No data remapping is needed. This is likely to be useful in handling unpredictable streams of analyst queries.

Sandia is exploring the capabilities of the MTA for graph-based informatics problems. Specifically, we are developing a flexible and extensible infrastructure to support efficient graph queries on semantic graphs. In our experience, the programming flexibility of the MTA allows for much more rapid algorithm and code development than on traditional parallel machines. This is not to say that MTA programming is easy. The programmer must expose parallelism and avoid subtle race conditions and memory hotspots. But in our design, the infrastructure will encapsulate many of these challenges, greatly easing the task of the programmer/analyst.

SAND2005-5600
Unlimited Release
Printed September 2005
In 2006, Cray will deliver a successor to the MTA which they are calling Eldorado. This new machine will be built as a modification to the Red Storm architecture that was jointly developed by Cray and Sandia. Red Storm is a traditional distributed memory parallel computer with AMD Operton microprocessors. In Eldorado, the Opterons will be replaced with pin-compatible versions of the MTA processor. This will allow for the extensive reuse of previously capitalized technology, but it will lead to a less balanced machine than the original MTA. Specifically, the processor/network performance ratio will be a factor of 4-5 lower on Eldorado. Thus the number of active threads will need to grow to keep the processors busy. Also, the machine will no longer have uniform memory access. Local memory will be notably quicker to read and write than distant memory. Although Cray would like to retain the MTA programming model, this is likely to come at a performance cost. Paying explicit attention to memory locality may well lead to substantially greater performance.

We plan to continue development of our infrastructure, and our research on graph and knowledge discovery algorithms. However, as discussed above, we have concerns that the MTA programming model may be inefficient for the Eldorado machine. We would like to investigate alternative methods for graph algorithm development on Eldorado. Specifically, we will investigate treating Eldorado as a distributed memory machine (which in fact, it is), yet one that supports global addressing. Each processor would benefit from the latency tolerance afforded by the MTA processor, and so should get better much performance than traditional microprocessors on graph operations. Instead of MPI communications, we would exploit the shared global address space to exchange information. In this way, we would leverage much of our prior work on distributed memory graph algorithms. Specifically, we would study short paths and subgraph isomorphism using this approach. If the performance is compelling, we could then begin to study alterations to our existing infrastructure to support this programming style.

Yet another area of interest to us is the development of database technology on the Eldorado which may allow for a merging of the database-centric and the memory-centric alternatives described above. We have begun a small study of databases on the MTA but this is not the focus of our current funding. Latency bottlenecks are a significant performance-limiting aspect of current database machines, and our initial results suggest that massive multithreading might provide an attractive alternative.

Another important open question of interest to us is the overall architecture of a high-performance graph query system. Two basic approaches seem possible, with differing merits. Both approaches begin with a traditional intelligence database (or databases) from which the semantic graph is constructed. One approach is to keep the graph in a database as well (the *database-centric* approach). An analyst query is sent to the database, which leads to a portion of the graph being extracted and, if needed, sent to a special graph-processing supercomputer for analysis. The output of this analysis is then communicated to the analyst. In an alternative approach (the *memory-centric* approach), the graph resides permanently in the memory of the supercomputer. The analyst can still access the traditional database, but graph queries needn't rely on it. The memory-centric approach is likely to result in higher performance for two main reasons: database operations are avoided and the graph doesn't need to be extracted from slower disk storage before processing. However, the database-centric approach has the advantage of allowing the database to apply access control protocols. If the graph grows beyond the

SAND2005-5600
Unlimited Release
Printed September 2005
memory capacity of the graph-processing supercomputer and out-of-core algorithms can be developed, then the database centric may prove to be more flexible.

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed-Martin Company, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.