Parallel Clustering of English Verbs into Levin Classes

18.337/6.338 Final Project Progress Report

SUPPLEMENT

Mark Brigham Melanie Goetz Andrew Hogue
{brigham, melg, ahogue}@mit.edu

April 21, 2004

This document is a supplement to our originial progress report, to address issues about the parallelization of our code.

Our current plan is as follows:

1. Parse a large corpus (possibly from a World Wide Web crawl), in parallel, and build a sparse incidence matrix between important words which co-occur in the sentences of the corpus. We will pull out verbs, prepositions, subject nouns, object nouns, and prepositional object nouns.
2. Merge the separate incidence matrices into a single, sparse, distributed incidence matrix. We plan to redistribute the matrix *without* bringing all the data to the front end.
3. Partition the graph using a parallel version of spectral graph partitioning implemented in STAR*P. This partitioning should provide us with semantically clustered verbs. We may also try partitioning with other methods, such as geometric partitioning, to allow for comparison data.

Each of these components has a parallel part. The first is "embarassingly" parallel, as the processes doing the parsing do not have to communicate. If we do a web crawl, there may be some interprocess communication to coordinate the page queue.

To create a matrix suitable for the partitioning algorithms, part 2 will have to combine the separate matrices from each process into a single, sparse, distributed matrix. We hope to be able to perform this combination without bringing the data to the front end, as that would incur a large commuication penalty. Instead, we plan to coordinate a "vocabulary list" between the processors, which will allow each to index its words in a coherent manner while keeping communcation to a minimum.

The indexing scheme from part 2 will allow us to create a sparse distributed matrix using STAR*P's `sparse()` function on distributed vectors. For part 3, we then plan to port the Meshpart toolkit's `specpart()` spectral partitioning method to STAR*P to allow us to partition the graph in parallel.

Currently, there is no existing code for the reshuffling operation of part 2. Part 3 will be based on the existing serial code in the Meshpart toolkit and on the STAR*P sparse matrix implementation currently being worked on by Viral Shah. We have spoken with Viral and will work closely with him to implement any missing functionality in the current sparse methods.