Forgotten Treasures: MPI, Algorithms, Data Structures and The C Language To Empower Distributed Web Crawler
Seiya Kawashima, The University of Chicago, Radiology Department

Conclusion

a) MPI gives controls to design and manage multiple processes and machines anyway you imagine.
b) Efficiencies are not magical. They are from the right data structures and algorithms.
c) It works. It’s consisted of 40,998 lines of C code so far.
d) No memory leaks so far verified by a memory detector called Valgrind.

Introduction

This project answers the questions like below:

a) Is Map/Reduce from Google the only way to distribute tasks among computers?
b) Would like to use your knowledge about data structures and algorithms to build something big?
c) Do you feel that the languages, Java, C#, Node.js Ruby, Python are ok but wonder what C gives you at scale?
d) Would you like to know how a web crawler is designed and implemented?

Aim

a) Control threads, processes and multiple computers under MPI.
b) Think scale in a bottom up way but not top down.
c) Understand how important data structures and algorithms at scale.
d) Create zero memory leak.

Method

a) Understand the characteristics of threads and processes in C in user space and kernel space as well.
b) Understand the efficiency of text analysis including searching, storing and retrieving.
c) Optimize each component using the right data structures and algorithms.
d) Use a memory detector to detect memory leaks and invalid memory access.

Tips

a) To optimize at scale, make sure that each component is optimized before employing multiple threads, processes and machines.
b) Slow algorithms don't get any faster employing multiple threads, processes and machines.

Approximate number of pages to crawl a day

<table>
<thead>
<tr>
<th>Crawled Delay (sec)</th>
<th># of times to crawl a day</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>80,400</td>
</tr>
<tr>
<td>30</td>
<td>2,890</td>
</tr>
<tr>
<td>60</td>
<td>1,440</td>
</tr>
</tbody>
</table>

Approximate number of pages to crawl a day based on web server’s instantaneous pages crawled because each page is crawled at the web page’s absolute URL. The crawled delay is the time between two crawling at the next page. If the delay is small enough, the crawled time between two pages will be small enough. If a page takes a long time to be crawled, the crawled time between two pages will be long enough. This paper uses an un-crawled empty string as the default value if the crawled time between two pages is not available.

Based on the technologies, the author is trying to create a web search engine. If you are interested in it, please contact the author at skawashima@uchicago.edu.