Towards parallel computation of linguistic concordance procedures

Master project (2 credit points)

Natural Science Faculty of the University of Basel)
Department of Mathematics and Computer Science)
High Performance Computing
https://hpc.dmi.unibas.ch/en/

Examiner: Prof. Dr. Florina Ciorba
Supervisor: Ahmed Hamdy Mohamed Eleliemy

Lesther Zulauf- Bal-ut
lesther.zulauf@unibas.ch
13-053-863

15 January 2021

Acknowledgments

I would like to thank Prof. Dr. Florina Ciorba and my supervisor Ahmed Eleliemy for their input, for their encouragements and for making this project possible.

Table of Contents

1 Introduction

Common procedures in Corpus Linguistics (e.g. ngram search, keyword comparison, concordance, and collocational search) rely on term frequency counts and pattern matches provided by exhaustive search algorithms [4]. As datasets of widespread linguistic corpora continue to increase in size [3], it is unsurprising that larger datasets incur higher computational costs. However, longstanding and widely-available linguistic concordancers are not designed with performance in mind, and/or the underlying engines used for search are not evaluated in terms of their performance [1] [5]. This project fills this twofold research gap by extending the existing BalConc concordance engine [6] such that it exploits the affordances provided by modern high performance computing (HPC) architectures.
In this report, I present the results of the strong- and weak-scaling analysis of various HPC systems (i.e. shared-memory, distributed-memory, hybrid) to assesses the performance of the reimplemented engine's core search functionality. In doing so, the possible benefits of parallel programming in this problem domain in terms of reduced execution time are assessed.

2 Related Work

Concordance search and collocational search are functions offered by Corpus Linguistic software (which are typically referred to as concordancers) such as AntConc[1], Wordsmith [5], and Corpus Workbench [2]. Both of these search functions rely on finding all matches in the documents of a corpus (i.e. a collection of documents containing naturally-occurring language) given a pattern containing a span of token and/or annotations. In concordance search, linguists are interested in studying matches in context (i.e. neighbouring tokens to the left and to the right). In collocational search, linguists are interested in the collocates which are the neighbours which co-occur with a given pattern with high relative frequency. Both of these search functions rely on performing an exhaustive search on the documents of the corpus to obtain the matches. Each document of a given corpus has to be traversed all (series of) tokens matching a certain pattern.

3 Proposed approach

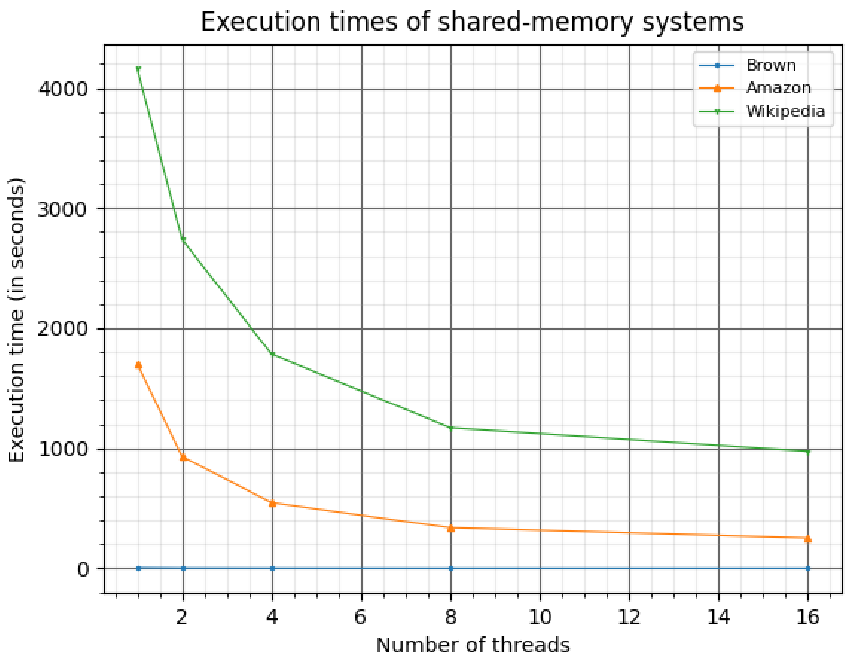
The approach explored by this paper is to traverse the documents of a given corpus in parallel using shared-memory systems, distributed memory systems and hybrid systems. The proposed approach is a straightforward one: instead of traversing the corpus' documents sequentially, the documents of a corpus are distributed to available processing elements and are traversed in parallel.

4 Design of the experiments

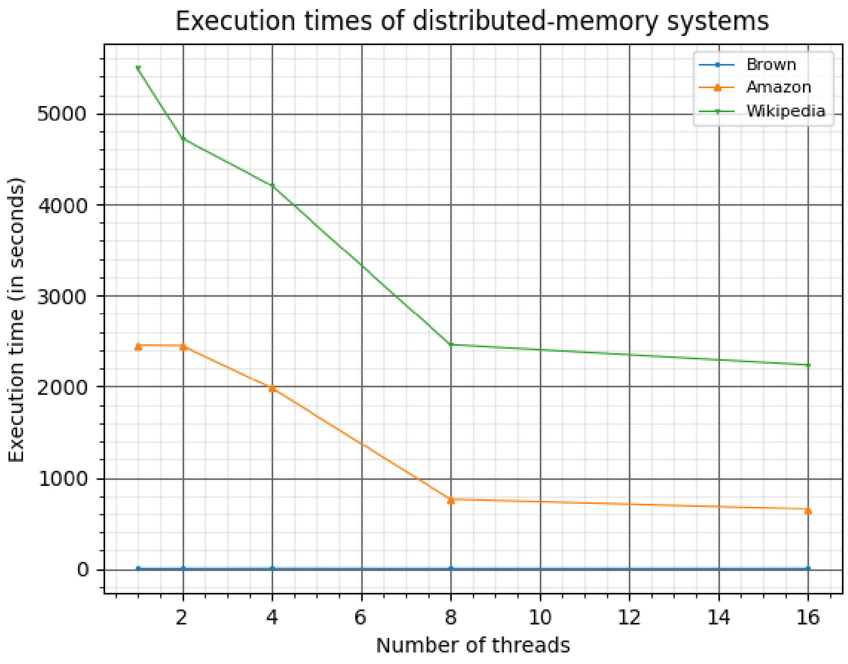
The search engine was reimplemented to run on shared-memory systems, distributed-memory systems and hybrid systems. For experiments using shared-memory systems, we use $\mathrm{t} \in \{1, 2, 4, 8, 16\}$ threads. For experiments using distributed-memory systems, we use $\mathrm{m} \in \{1, 2, 4, 8, 16\}$ processors. For experiments using hybrid systems, we use $\mathrm{n} \in \{1, 2, 4, 8, 16\}$ processors each with 16 threads. The programming language used was Python and the execution times were measured five times for each search system.

5 Strong-scaling analysis

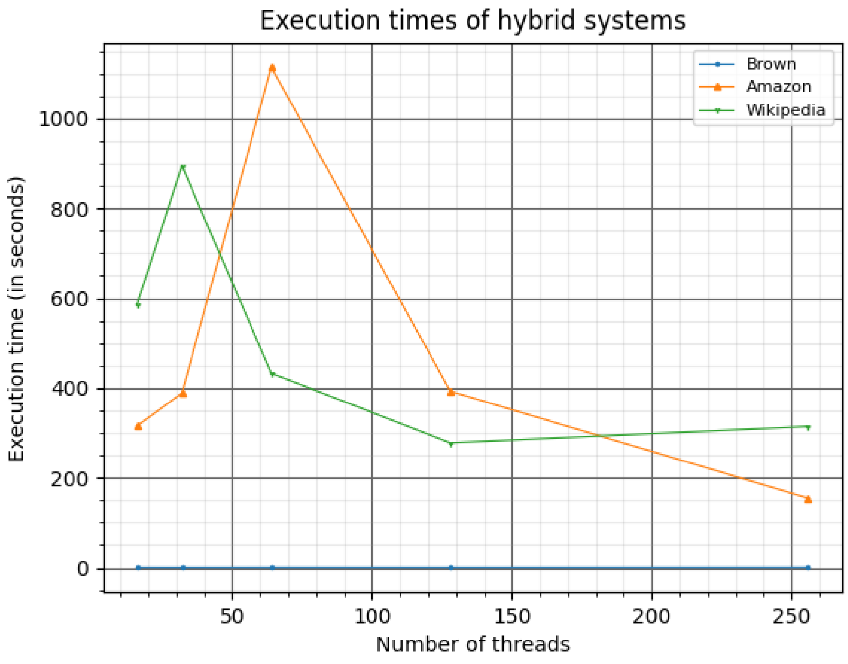
5.1 Shared memory system



5.2 Distributed memory system

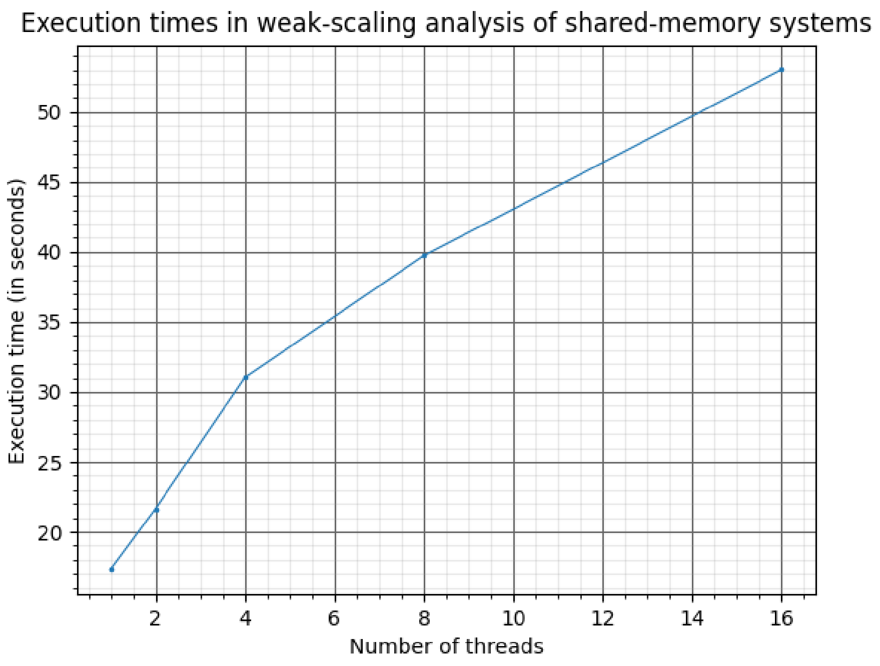


5.3 Hybrid system

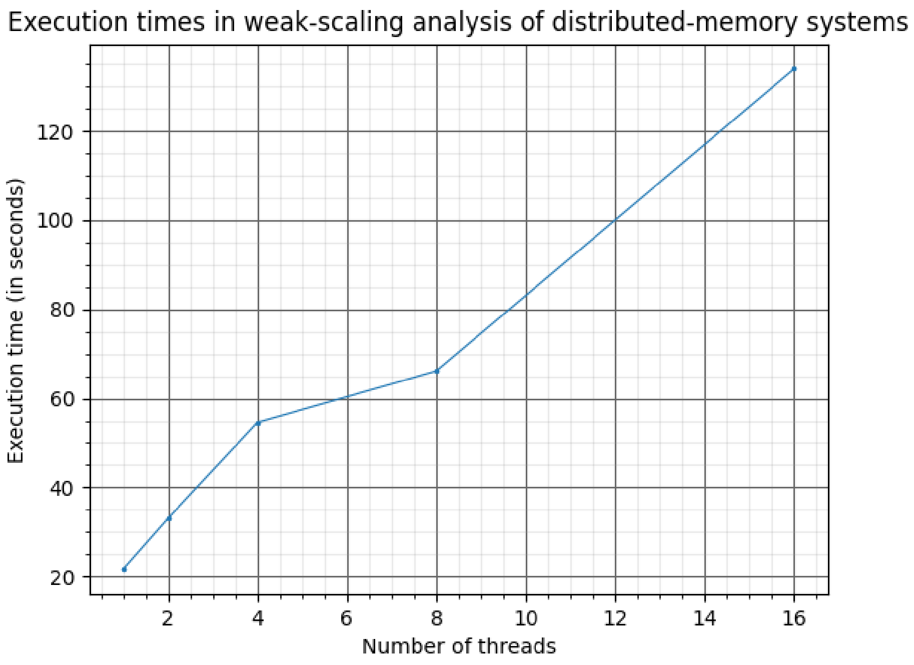


6 Weak-scaling analysis

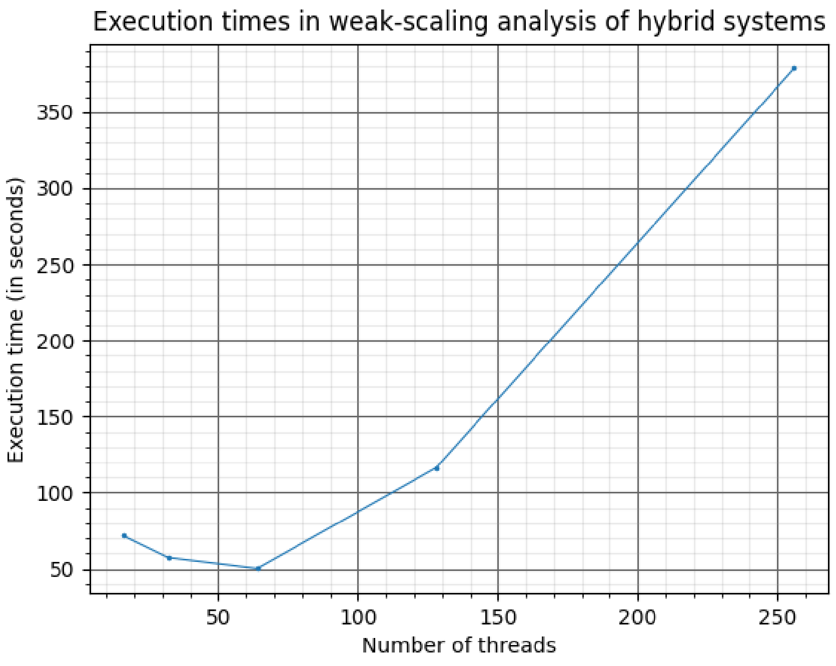
6.1 Shared memory system



6.2 Distributed memory system



6.3 Hybrid system



7 Conclusion

In this project, we measured the speedup of the execution times and reimplemented search functionality of the BalConc concordance engine to run on different HPC systems including distributed-memory and shared-memory systems. The experiments show that parallel programming reduces the required execution time.

All in all, while not every person doing Corpus-based research may have access to HPC systems, it is a worthwhile to look into the application of parallel algorithms in this domain because the vast majority of personal computers nowadays have multiple processors.

Bibliography

[1] Laurence Anthony. Antconc: A learner and classroom friendly, multi-platform corpus analysis toolkit. *proceedings of IWLeL*, pages 7–13, 2004.

[2] Oliver Christ, Bruno M Schulze, Anja Hofmann, and Esther Koenig. The ims corpus workbench: Corpus query processor (cqp): User's manual. *University of Stuttgart*, 8, 1999.

[3] Eric Friginal, Marsha Walker, and Janet Beth Randall. Exploring mega corpora: Google ngram viewer and the corpus of historical american english. *EuroAmerican Journal of Applied Linguistics and Languages*, 1(1):48–68, 2014.

[4] Christopher Manning and Hinrich Schutze. *Foundations of statistical natural language processing*. MIT press, 1999.

[5] AP Berber Sardinha. Wordsmith tools. *Computers & Texts 12 (1996)*, 1996.

[6] Lesther Zulauf-Bal-ut. Balconc. Computer software.