Needleman-Wunsch Algorithm on a large phonetic dataset

Dominic Plein

October 19, 2024

Abstract

This project is part of the GP-GPU Computing course at UGA by Christophe Picard. Here, we present a proposal for the final project. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

Contents

1 Introduction

The International Phonetic Alphabet (IPA) uses special symbols[1] to represent the sound of a spoken language. This is useful for language learners since the pronunciation of a word can be significantly different from its written form. For example, the French word *renseignement* (information) is pronounced /ʁɑ̃.sɛɲ.mɑ̃/. Based on this alphabet, one might wonder if we can construct a metric that quantifies the **distance between two words based on their phonetic transcription**. This would allow to construct a graph where nodes are words and edges are weighted by the distance between the words. This graph could then be used to find neighbors of a word based on their phonetic similarity. This opens up the possibility to use clustering algorithms and other methods from graph theory to analyze the phonetic structure of a language.

In the following, by *word* we always refer to its phonetic transcription. That is, homophones like the French words *vert* /vɛʁ/ (green) and *verre* /vɛʁ/ (glass) are considered the same word. We employ the Needleman-Wunsch algorithm to calculate the “score” (distance) between two words. After an introduction to this algorithm in section 2, we present ways to implement this in parallelized ways. First, on the CPU in Rust using the *Rayon* library, and then on a consumer Nvidia GPU in the CUDA framework by means of the *cudarc* Rust library. Finally, we discuss some applications and present excerpts from the obtained graphs that we visualized in Gephi.

[1] See for example the French list here.

2 Needleman-Wunsch algorithm

which calculates the global alignment of two sequences and was originally used in bio-informatics to compare DNA sequences. Here, the alphabet will instead consist of the phonetic symbols. Out of all possible alignments of two words (including gaps), the Needleman-Wunsch algorithm finds the one with the smallest distance, i. e. the alignment with the highest score.

A good introduction to the algorithm can be found on the respective Wikipedia page.

Algorithm 2.1: Needleman-Wunsch

Input: $A = \{ A_0, \ldots, A_{\texttt{len}(A)-1} \}$, $B = \{ B_0, \ldots, B_{\texttt{len}(B)-1} \}$,
similarity: similarityScoreFunc, p: GapPenalty
Output: score

```
1  Function calculateScore():
2      Init scoreMatrix with dimensions (len(A) + 1) × (len(B) + 1)
3      for i ∈ {0, ..., len(A)} do
4          scoreMatrix[i][0] ← p · i
5      for j ∈ {0, ..., len(B)} do
6          scoreMatrix[0][j] ← p · j
7      for i ∈ {1, ..., len(A)} do
8          for j ∈ {1, ..., len(B)} do
9              cost ← similarity(A_i, B_j)
10             matchScore ← scoreMatrix[i − 1][j − 1] + cost
11             deleteScore ← scoreMatrix[i − 1][j] + p
12             insertScore ← scoreMatrix[i][j − 1] + p
13             scoreMatrix[i][j] ← max(matchScore, deleteScore, insertScore)
14     return scoreMatrix[len(A)][len(B)]
```

And we go on

Glossary

IPA International Phonetic Alphabet. 1