

Sequential and parallel implementation of bigrams and trigrams, Java and C++

Giulio Calamai
Marco Loschiavo

Introduction

The main goal is to compute and estimate occurrences of bigrams and trigrams in a certain text.

More in general:

- A sequence of **two** letters (e.g. of) is called a **bigram**.
- A **three**-letter sequence (e.g. off) is called a **trigram**.
- The general term **n-gram** means '**sequence of length n**'.

Languages

We have implemented the task in 2 versions using different technologies:

Sequential version:

- **Java**
- **C++**

Parallel version:

- **Java Thread**
- **C++ Thread (Pthread)**



Computation

- Sequential version:

Algorithm 1: Sequential version

Data: n , filestring

```
1 for i = 0 to filestring.length-n+1 do
2   key = "";
3   for j = 0 to n - 1 do
4     key = key + filestring[i+j];
5   end
6   // insert key in a data structure;
7 end
```

In particular

- n : Number of grams
- filestring: contains the characters from txt file

- Parallel version

Algorithm 2: Parallel version

Data: id , k , n , begin, stop, filestring

```
1 for i = this.begin to (this.stop - this.n+1) do
2   key = "";
3   for j = 0 to this.n - 1 do
4     key = key + filestring[i+j];
5   end
6   // insert key in a data structure;
7 end
```

In particular

- id = idthread
- k = $\text{floor}(\text{text.length}/n\text{Threads})$ // Dimension of test splits
where $n\text{Thread}$ = Number of threads
- n = $n^{\circ}\text{grams}$
- $\text{begin} = (k * i)$
- $\text{stop} = (i+1)*k + ((n-1)-1)$
- $\text{filestring} = \text{txt}$

Thread's attributes

- id
- begin
- stop
- filestring

Sequential implementation - Java

Data preprocessing

- ***readTextFromFile()***: read the text to analyze and replace opportunely those characters defined *invalid*.

Data structure

- **HashMap**: constant time performance for the basic operation (*get()*, *put()*) for large sets.

Computation

- ***computeNgrams()***: follows the sequential computation explained above.
 - *int n*: identifies the number of grams (two or three).
 - *char [] filestring*: text to analyze.

Sequential implementation - C++

Data preprocessing

- ***readTextFromFile()***: read the text to analyze and replace opportunely those characters defined *invalid*.

Data structure

- **UnorderedMap**: is the chosen structure to replace the java Hash Maps.

Computation

- ***computeNgrams()***: follows the sequential computation explained above.
 - *int n*: identifies the number of grams (two or three).
 - *char * filestring*: text to analyze.

Parallel implementation

Idea

- Divide the text in as many parts as are the thread instances.
- Entrust the search of bigrams and trigrams on a single part to a single thread.

Example

Il 5 luglio 2018 il parlamento europeo in seduta plenaria
 deciderà se portare avanti o meno una proposta di direttiva sul
 diritto d'autore unanimemente avversata dalla società civile e
 persino da alcune commissioni del Parlamento.

Il 5 luglio 2018 il parlamento europeo in seduta plenaria
 deciderà se portare avanti o meno una proposta di direttiva sul
 diritto d'autore unanimemente avversata dalla società civile e
 persino da alcune commissioni del Parlamento.

Il 5 luglio 2018 il parlamento europeo in seduta plenaria
 deciderà se portare avanti o meno una proposta di direttiva sul
 diritto d'autore unanimemente avversata dalla società civile e
 persino da alcune commissioni del Parlamento.

T1

Il 5 luglio 2018 il parlamento europeo in seduta plenaria
 deciderà se portare avanti o meno una proposta di direttiva sul
 diritto d'autore unanimemente avversata dalla società civile e
 persino da alcune commissioni del Parlamento.

T1

Il 5 luglio 2018 il parlamento europeo in seduta plenaria
 deciderà se portare avanti o meno una proposta di direttiva sul
 diritto d'autore unanimemente avversata dalla società civile e
 persino da alcune commissioni del Parlamento.

T2

Il 5 luglio 2018 il parlamento europeo in seduta plenaria
 deciderà se portare avanti o meno una proposta di direttiva sul
 diritto d'autore unanimemente avversata dalla società civile e
 persino da alcune commissioni del Parlamento.

T1

T2

T3

Il 5 luglio 2018 il parlamento europeo in seduta plenaria
 deciderà se portare avanti o meno una proposta di direttiva sul
 diritto d'autore unanimemente avversata dalla società civile e
 persino da alcune commissioni del Parlamento.

1 thread

2 threads

3 threads

Parallel implementation - Java thread

Some features of `java.util.concurrent` package have been used:

- **Future**
- **ExecutorService**

Data preprocessing

- `readTextFromFile()`

Data structure

- **ConcurrentHashMap:**
 - Allows concurrent modifications of the map from several threads without blocking them.
 - It's lock-free on reads.
 - Lock only the bucket that's being altered.



Parallel implementation - Java thread

Computation

- **Declare** a thread class which implements *callable*.
- **Implement** the *call()* method which computes bigrams and trigrams as described before.
- **Implement** a *HashMerge()* function to merge the maps returned from threads.
- **Instantiate** a Future array and an ExecutorService specifying the thread pool size.
- **Use** the ExecutorService object to submit the compute method and get the results through *.get()* Future method.

Parallel implementation - C++ thread

Pthread API have been used to parallelize the C++ code.

Data preprocessing

- *readTextFromFile()*

Data structure

- UnorderedMap



Parallel implementation - C++ thread

Computation

- **Implement** the Thread class including <pthread.h>
- **Extend** the class above, including <Thread.h> defining ParallelThread class.
- **Implement** a ParallelThread method *int *run()* which computes bigrams and trigrams as described before. A getter method provides to return the map.
- **Define** a *HashMerge()* function to merge the returned maps from threads.
- **Declare** an array of threads and a vector of UnorderedMap. The threads are instantiated into the array and started. After the *join()* the returned maps are pushed back into the vector and merged.

Tests

Sequential version:

- The computational time has been collected 100 times vary with the text (from 50Kb to 150Mb) and averaged.

Parallel version:

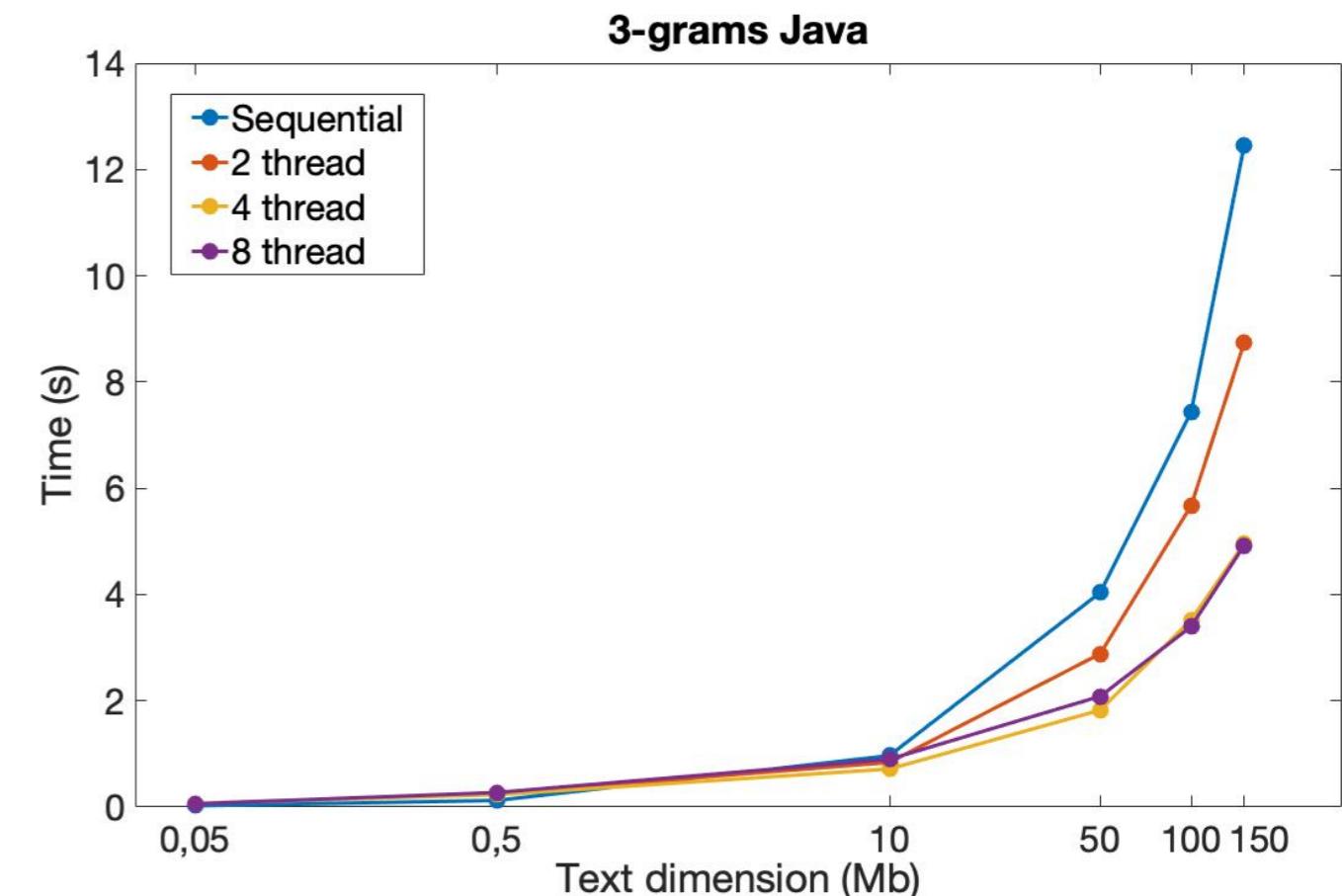
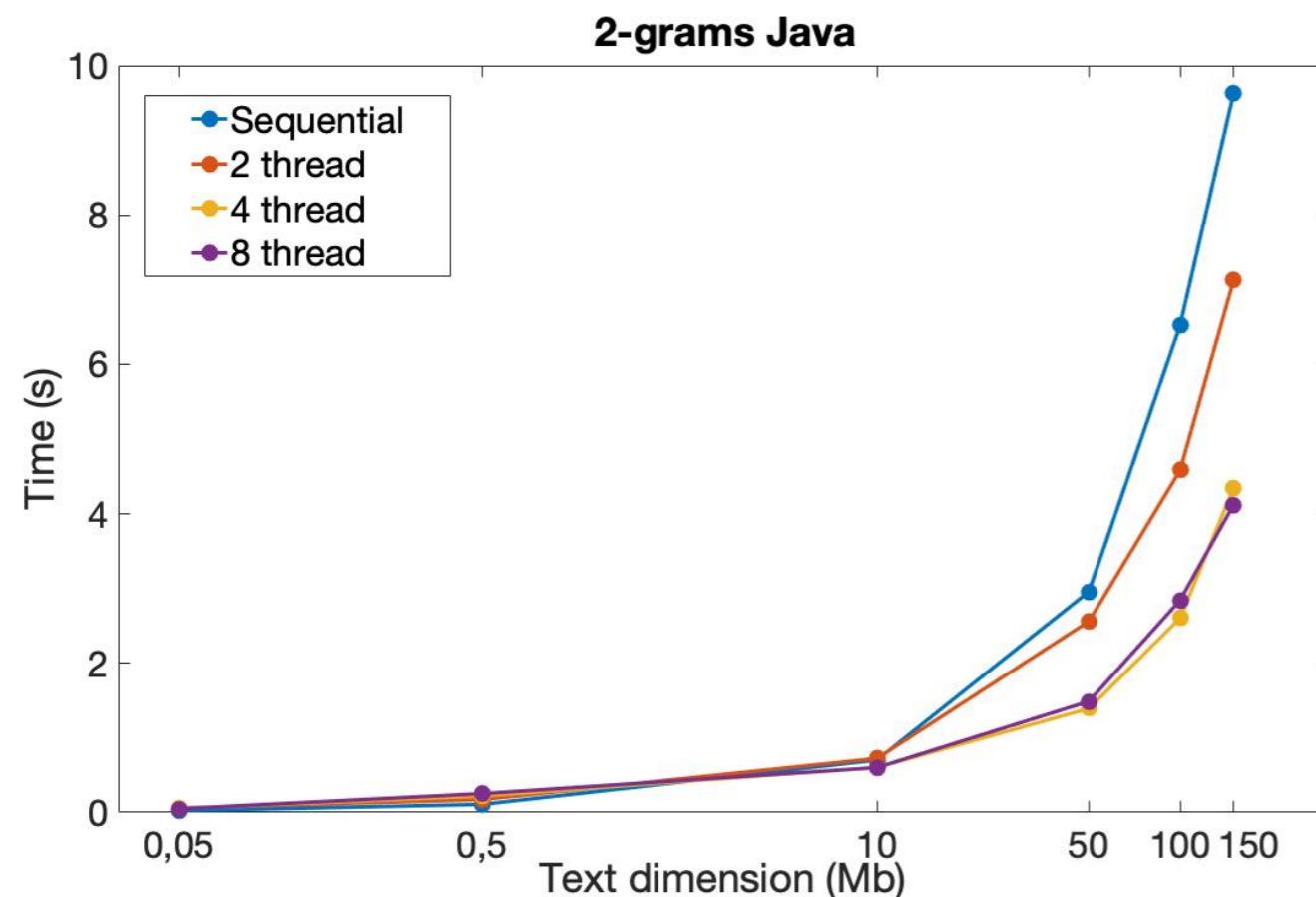
The number of tests is depending on the number of threads used.

- 2, 4 and 8 threads for the java parallel version
- 2, 4 and 8 threads for the C++ parallel version

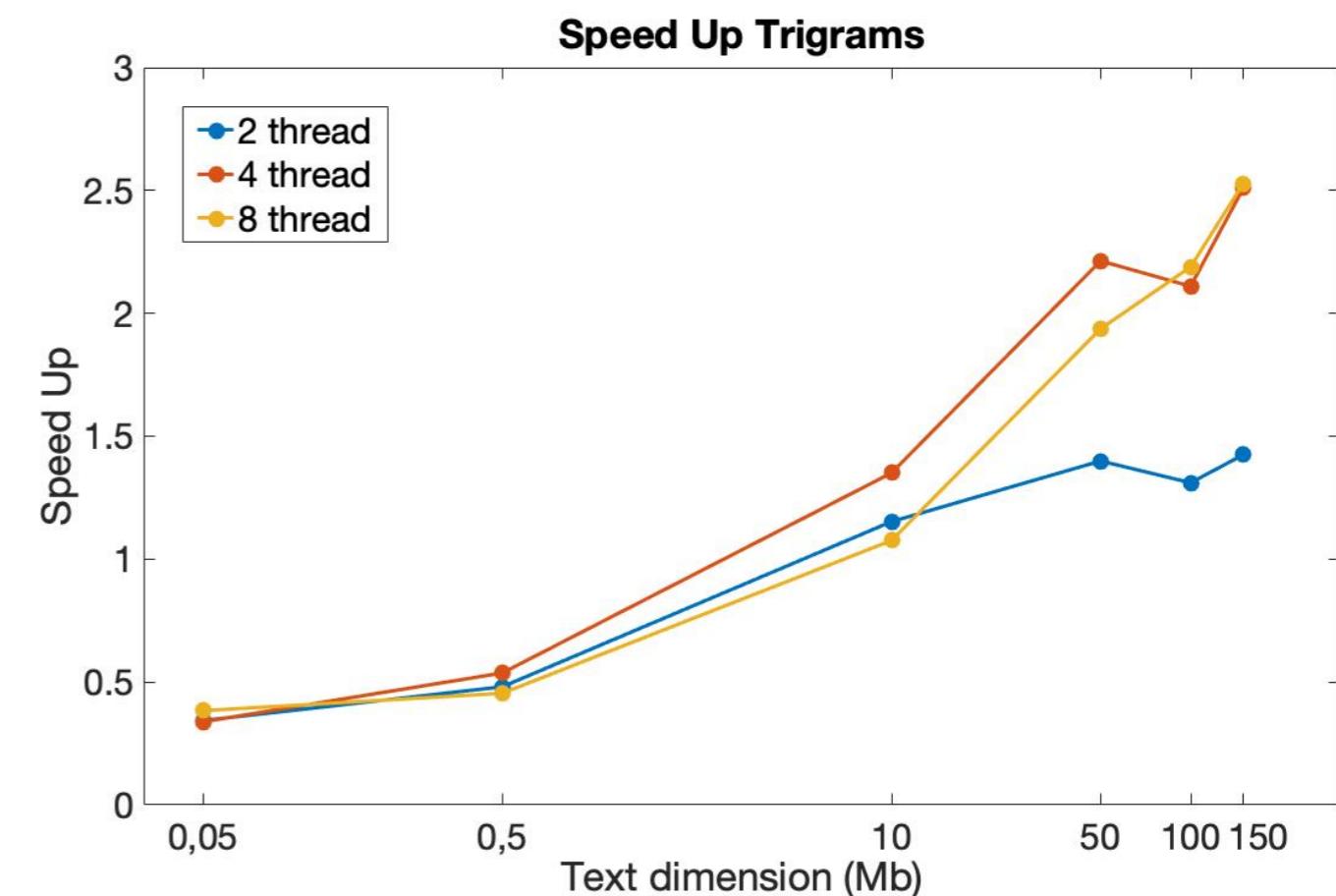
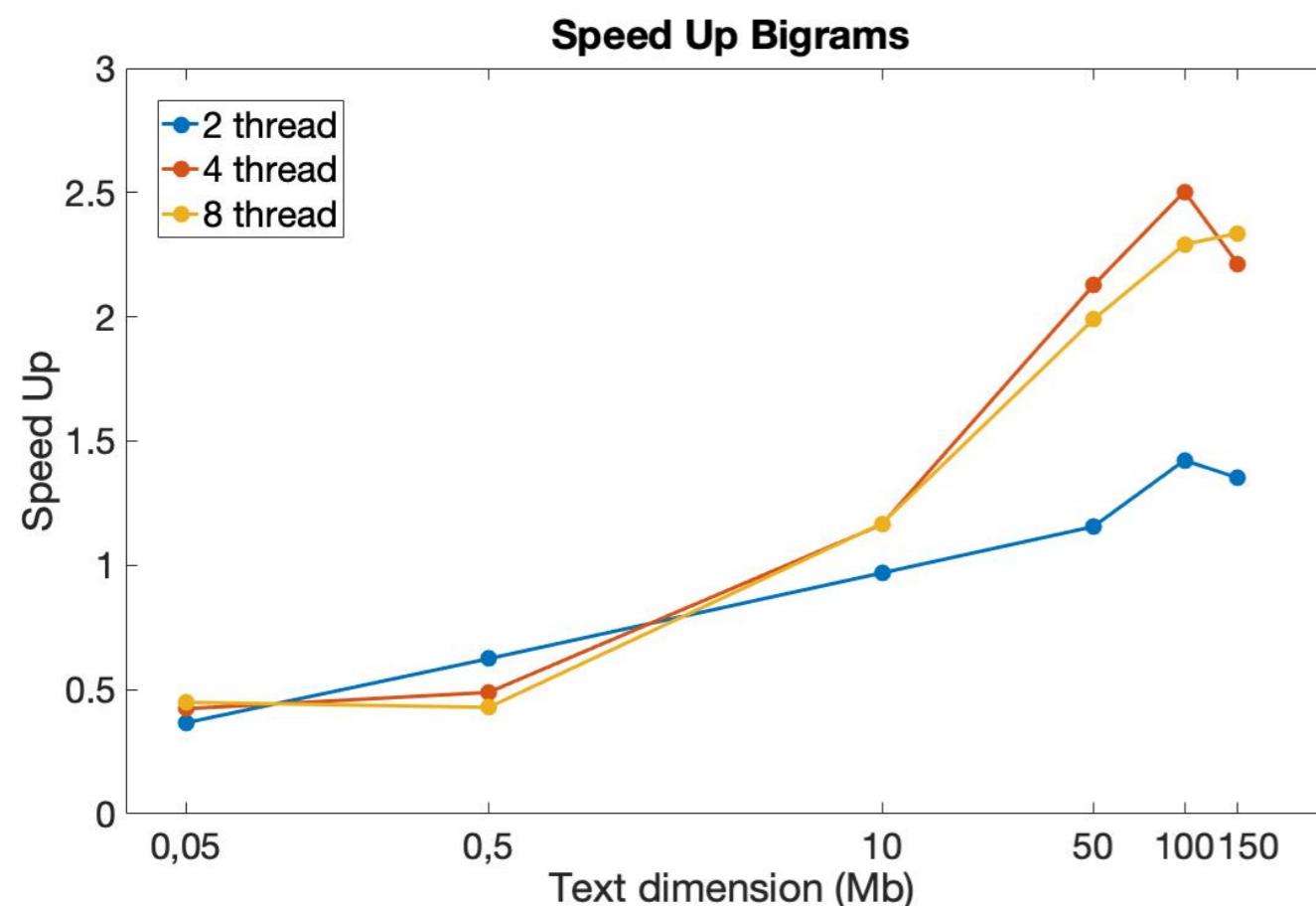
For every thread number the computational time has been collected 100 times and averaged.

Over the maximum number of threads chosen, computational times don't improve more significantly.

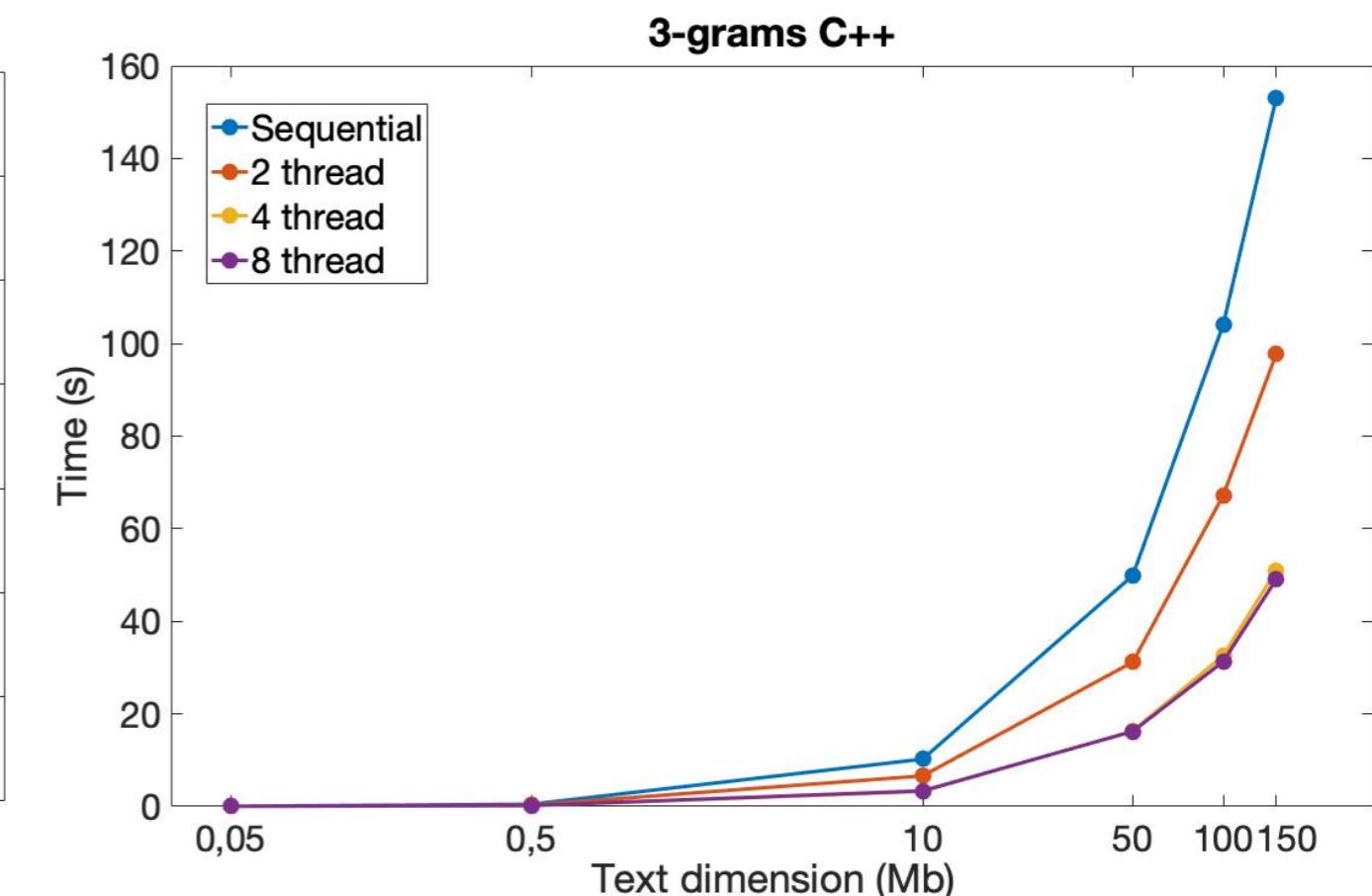
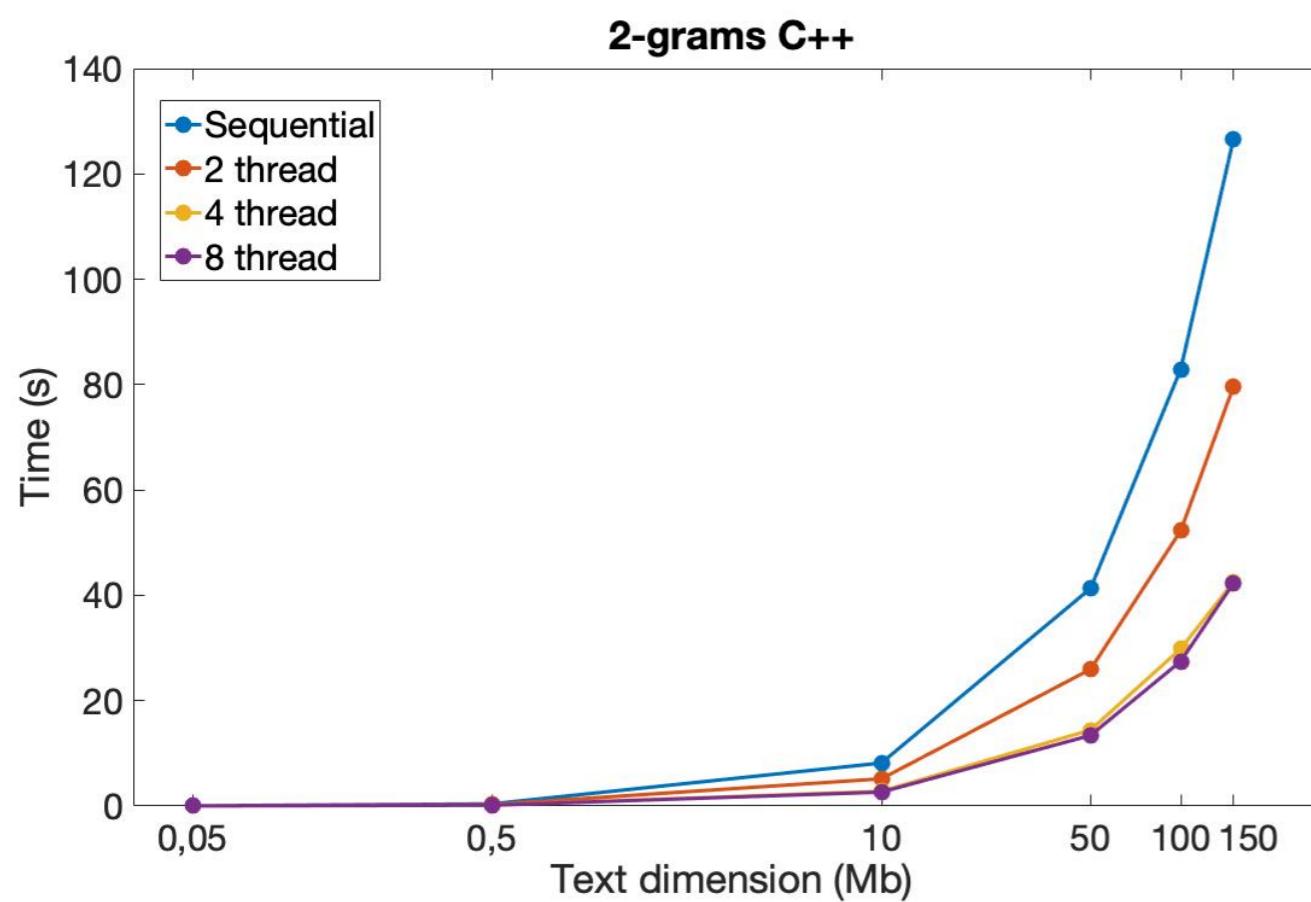
Results - Java



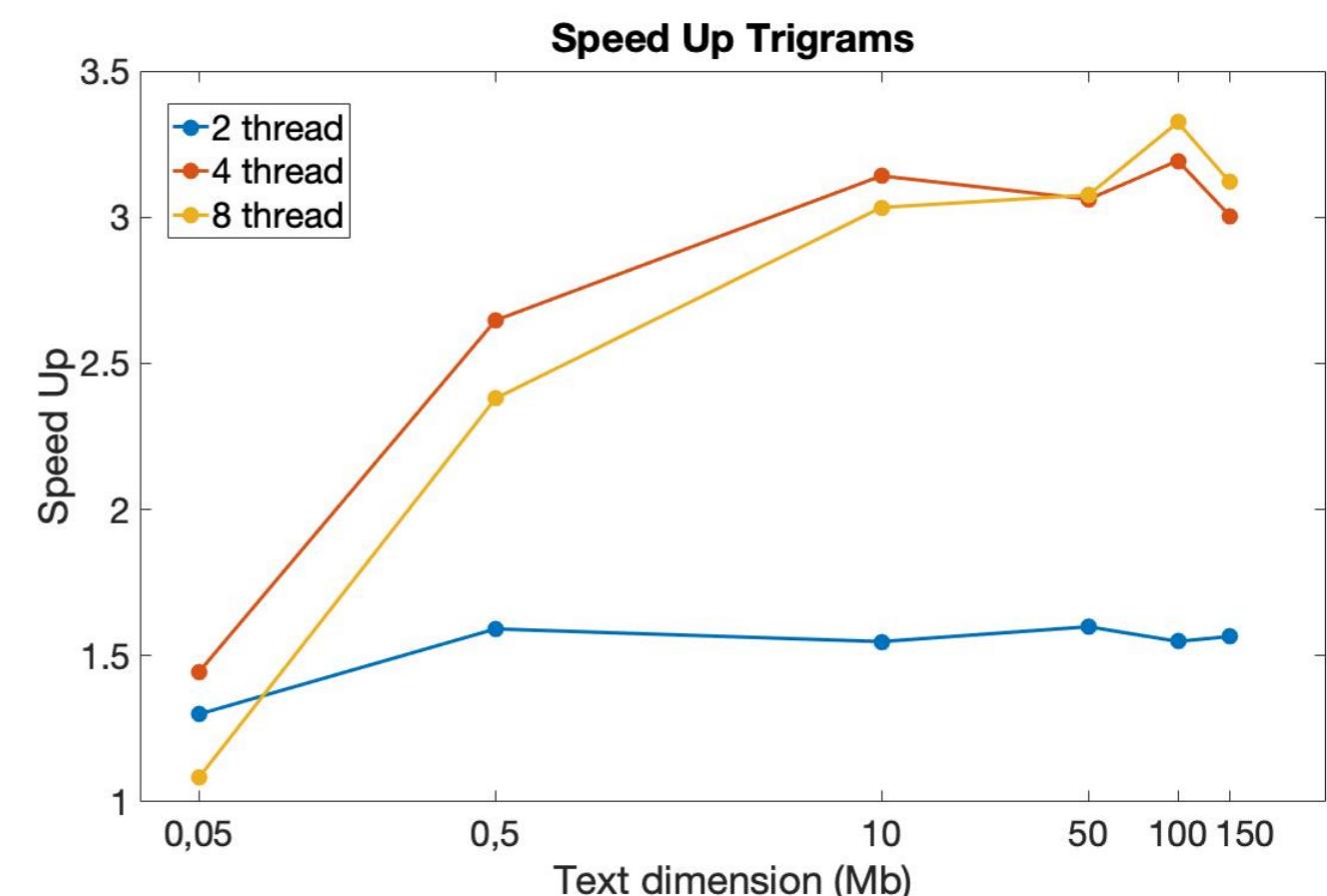
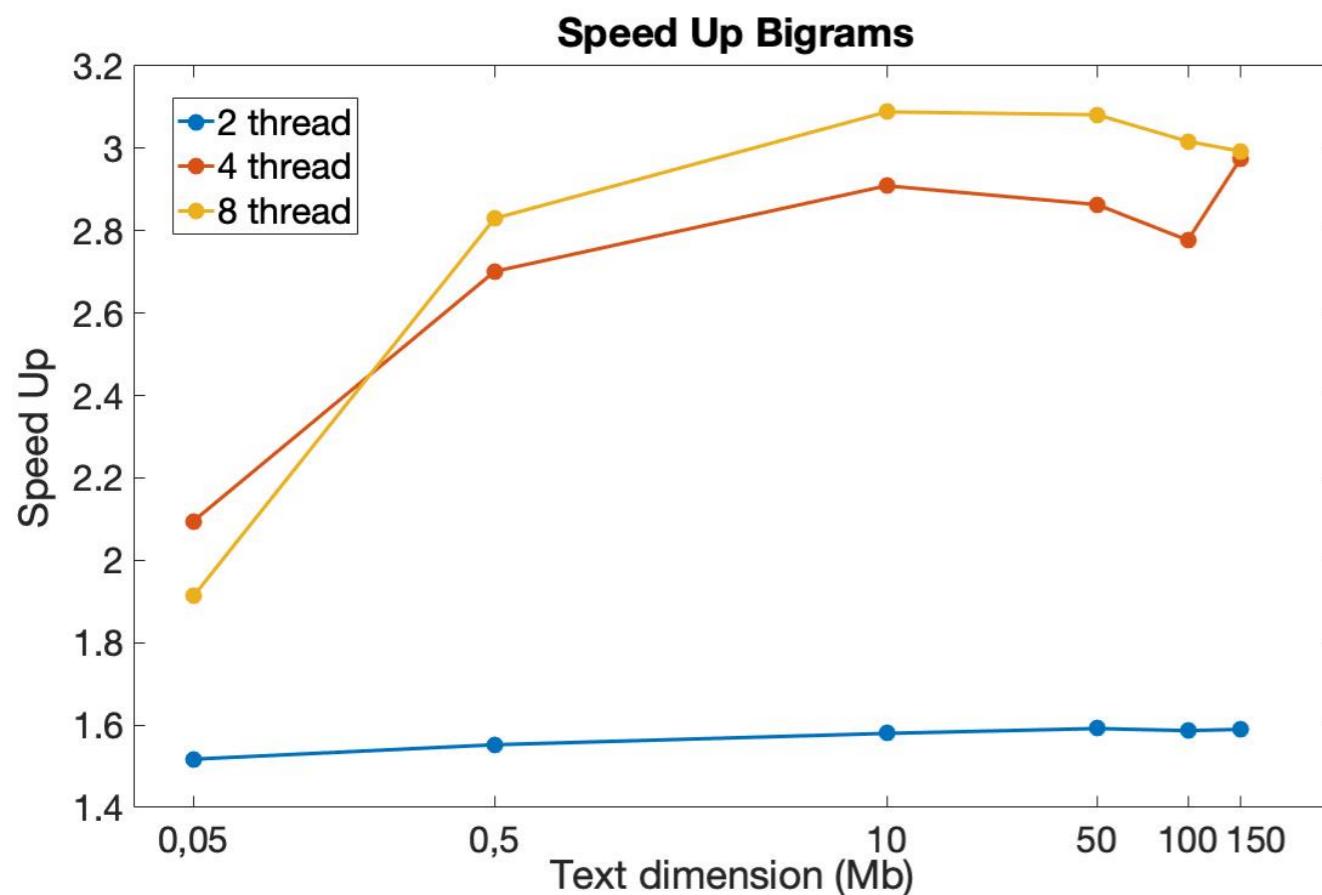
Results - Java



Results - C++



Results - C++





Sequential and parallel implementation of bigrams and trigrams, Java and C++

Giulio Calamai
Marco Loschiavo