Efficient Time Series Search: Sequential and parallel Approaches Using C++ OpenMP and Python's Joblib and asyncio

Dylan Fouepe
E-mail address
dylan.fouepe@edu.unifi.it

Abstract

This project explores methods for searching a given time series within a larger dataset of time series using both sequential and parallel computing approaches. The parallel implementations utilize the OpenMP library in C++ and the Joblib and asyncio libraries in Python. The study aims to evaluate the performance improvements achieved through parallelization and to identify the most effective strategies for time series analysis

1. Introduction

Time series analysis is crucial in various fields, including finance, weather forecasting, and biomedical signal processing. A common problem within this domain is the search for a specific time series pattern within a larger set of time series data. Traditional sequential search algorithms can be slow and inefficient, especially with the increasing volume of data. This project aims to address this challenge by exploring both sequential and parallel search algorithms. Using the OpenMP library for parallel computing in C++ and Joblib and asyncio for parallelization in Python, we seeek to enhance the search process's efficiency and speed. The study will provide a comprehensive evaluation of the performance gains achieved through parallel computing and offer practical insights into the implementation of efficient time series search algorithms.

1.1. Time series

Time series data consist of sequential observations indexed by time, crucial for analyzing trends, patterns, and dependencies in various domains such as finance, weather forecasting, and healthcare. These data sequences enable researchers and analysts to discern temporal behaviors, predict future trends, and derive meaningful insights from historical data. In this project, the focus lies on developing efficient algorithms to search for specific patterns within extensive time series datasets. Traditional sequential search algorithms sequentially evaluate each data point, which can be inefficient and time-consuming for large datasets. Conversely, parallel search algorithms break down the search task into smaller sub-tasks, processed concurrently across multiple processors or threads, thereby significantly enhancing search speed and efficiency.

1.2. Enhancing Time Series Search Efficiency with C++ OpenMP and Python's Joblib/Asyncio

C++ and OpenMP are selected for their robustness and effectiveness in parallel computing.
C++ offers high performance and low-level control, making it suitable for optimizing computational tasks.
OpenMP simplifies the parallelization of C++ code by providing directives and APIs for multi-threading, enabling efficient utilization of multi-core processors.
This combination enhances the scalability and speed of time series search algorithms, crucial for handling extensive datasets and achieving

real-time performance.

In Python, the Joblib library is utilized for its simplicity and versatility in parallel computing tasks.

Joblib provides tools for efficient execution of parallel tasks, particularly beneficial for data-intensive operations such as time series analysis.

Asyncio complements Python's asynchronous programming capabilities, enabling non-blocking, concurrent execution of I/O-bound tasks.

This concurrency helps in efficiently managing time series data access and processing, enhancing overall performance and responsiveness in search operations.

By evaluating the effectiveness of both sequential and parallel approaches in time series search, this project aims to provide valuable insights into optimizing search algorithms for large-scale time series datasets.

The comparative analysis not only benchmarks performance gains achieved through parallel computing but also identifies best practices and strategies for efficient time series analysis and pattern recognition.