Expert Systems With Applications 105 (2018) 77–88

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Parallel SAX/GA for financial pattern matching using NVIDIA's GPU

João Baúto, António Canelas, Rui Neves, Nuno Horta*

Instituto de Telecomunicações, Instituto Superior Técnico, 1049-001 Lisboa, Portugal

ARTICLE INFO

Article history:
Received 3 December 2017
Revised 21 February 2018
Accepted 14 March 2018
Available online 20 March 2018

Keywords:
Genetic Algorithms
Finance computation
Pattern matching
Symbolic Aggregate ApproXimation
GPU
CUDA

ABSTRACT

This paper starts by presenting a study from a computational performance standpoint of SAX/GA, an algorithm that uses the Symbolic Aggregate approXimation (SAX), to dimensionally reduce time series, and the Genetic Algorithm (GA) to optimise market trading strategies. This study highlights how the sequential implementation of SAX/GA and genetic operators works. This study is later used as the baseline for the development of parallel techniques capable of exploring the identified points of parallelism that simply focus on accelerating the heavy-duty fitness function to a full Graphical Processing Unit (GPU) accelerated GA. The implemented solutions accelerated the sequential single-core SAX/GA solution in about 30 times with a maximum of nearly 180 times.

1. Introduction

The financial system and, particularly, the stock markets represent a huge amount of real and dynamic data resources for researchers allowing the true exploration of new algorithms and methodologies. Technical analysis is a well-known tool used by traders to identify market decision points, i.e., buy and sell opportunities. Pattern recognition plays a key role when identifying market and/or stock trends, however, this is not feasible by hand anymore due to the huge size of financial time series and the large number of available stocks. Therefore, it is clear that creating a custom trading strategy that uses patterns as decision points of entry or exit on the market can be a tedious and long process of optimisation where multiple possible decisions sets must be tested against large time series. Researchers have been trying to ease the optimisation process by reducing datasets while maintaining a precise representation with minimal data loss, however, this is a trade-off between lower execution time or less accurate trading decision set. This paper presents a study from a computational performance point of view on how to accelerate the SAX/GA algorithm, an algorithm designed for financial trading over long periods, which uses the SAX representation to dimensionally reduce time series without significant information loss, providing a valid method for comparing two time series, while the GA optimizes a pool of possible solutions. The main downfall of using GA's or other type of "learning" algorithms is that the optimization process of solutions can be very time consuming even for modern Central Processing Units (CPU). Exploring an alternative execution system, like the Graphical Processing Unit (GPU), is the main ideal behind this paper however such task is not straightforward given the architectural differences between the CPU and GPU.

Section II presents a review of related work where the application of pattern recognition, matching or discovery techniques in the domain of computational finance. Then in Section III, the sequential CPU SAX/GA implementation is described and analysed to identify the execution bottlenecks to be consider in the parallelized solution. Moreover, the sequential SAX/GA implementation is considered for a performance benchmark, which is later used in a baseline comparison with the parallelized implementation, discussed in section IV. Based on the findings of the performance benchmark, Section V presents three solutions to accelerate the SAX/GA algorithm using a NVIDIA GPU. Solution A focuses on the fitness operator and, with that, minor modifications are proposed to the original fitness function, inspired by the out of order and speculative execution on the CPU architecture, to take advantage of the GPU. Further changes were implemented in solution B to understand what is the impact of knowledge transfer between generations which led to a complete parallel fitness process where all generations are processed at once. And finally, solution C is introduced to solve the performance issues directly caused by the parallel fitness process and the memory transfers associated, leading a fully accelerated GPU SAX/GA algorithm. The results obtained with each solutions are discussed in Section VI with particular interests in two metrics, the Return on Investment (ROI) and the speedup achieved when compared to the original algorithm. The ROI met-

* Corresponding author.
E-mail addresses: joao.bauto@tecnico.ulisboa.pt (J. Baúto), antonio.canelas@tecnico.ulisboa.pt (A. Canelas), rui.neves@tecnico.ulisboa.pt (R. Neves), nuno.horta@lx.it.pt (N. Horta).

https://doi.org/10.1016/j.eswa.2018.03.026
0957-4174/

Table 1
Implementations of pattern recognition techniques on the literature.

Refs.	Technique	Architecture	Financial market	Dataset	Performance
Fu et al. (2007)	PIP Template	CPU	HSI	2532 points	~96% accuracy
Fu et al. (2007)	PIP Rule	CPU	HSI	2532 points	~38% accuracy
Zapranis and Tsinaslanidis (2010)	PIP-SOM	CPU	N/A	N/A	97.1% accuracy in HS pattern
Li et al. (2011)	MPLA-DDTW	CPU	N/A	2000 points	2x lower error than PAA
Chang, Deka, Hwu, and Roth (2012)	Shapelets	GPU	N/A	N/A	~15x speedup
Chen, Tseng, Yu, and Hong (2013)	PIP-GA	CPU	TAIEX	N/A	Correct identification of 4 patterns
Canelas et al. (2013b)	SAX-GA	CPU	S&P500	Jan 2005 Apr 2010	67.76% average return (ROI)
Bakhach et al. (2016b)	DC	CPU	Forex	Jan 2015 Jul 2015	67–78% Profitability

ric serves as the financial baseline that verifies that the developed solutions and modifications introduced did not have a negative impact while the speedup is used to create the performance ground truth of the algorithm. Section VII concludes the proposed work with some final remarks regarding the results obtained.

2. Related work

Pattern recognition, matching or discovery are terms associated with the identification of specific sequences that are either known a priori (recognition or matching) or are found and represent an important discovery. Although the focus will be on pattern matching techniques applied to financial time series (Table 1), these techniques proved to be very versatile and expandable to different areas, from the medical sector with applications in Electrocardiogram (ECG) (Chen, Hsieh, & Yuan, 2004) to the energy sector with forecasting and modelling of building energy profiles (Iglesias & Kastner, 2013).

The Middle curve Piecewise Linear Approximation (MPLA) approach looks to evade a problem associated with the simple Piecewise Linear Approximation (PLA), the delayed representation of a trend inversion, by finding inversion points based on the amplitude of three consecutive points, q_{i-1}, q_i and q_{i+1}. Once an inversion point is located (p_j), it is saved to be used in the future to create a so-called middle curve of the time series which is then passed as input to a different PLA technique, the Divisive Piecewise Linear Approximation (DPLA). Li, Guo, and Qiu (2011) evaluated the MPLA techniques in an unknown market and period of time. A subsequence Q of length 50 was selected from a time series P and the MPLA approach not only could search the input pattern Q but also discover identical sub-sequences in P.

A disadvantage of MPLA is the increase in time complexity comparatively to other techniques such as the SAX. Similarities can be found when comparing MPLA to the Directional Changes (DC) approach (Bakhach, Tsang, & Jalalian, 2016a; Bakhach, Tsang, Ng, & Chinthalapati, 2016b) which, not only searches for extreme points or trend inversion points, but also points that confirm a directional change. The decision on whether a point confirms a change is made based on the absolute percentage change of the price at the previous inversion point and the current point.

Much like the MPLA technique, the Piecewise Aggregate Approximation (PAA) uses an identical principle of dividing a time series into smaller frames, however, instead of a linear representation for each frame, the average of a frame's values is used as the representative value of that portion in the time series. To perform any type of comparison between two time series, these must be normalised (Eq. (1)) to a common baseline allowing a direct comparison.

$$x'_i = \frac{x_i - \mu}{\sigma} \tag{1}$$

where x_i represents a point, μ is the mean and σ is the standard deviation of the time series.

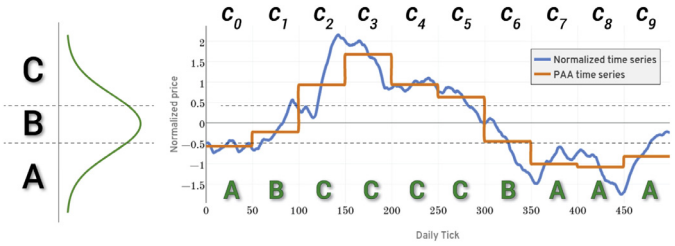

Fig. 1. Transformation of a PAA series into a SAX representation with 3 symbols.

The time series is then divided into N frames of size W and the mean value ($\bar{p}_i$) inside that frame is used as the approximation of that portion (Eq. (2)). Although PAA adopts frames with fixed size, it is possible applied this method to uneven frames where the border element of two frames split his value contribution between the neighbour windows (Wei, 2006).

$$\bar{p}_i = \frac{W}{n} \cdot \sum_{j=\frac{N}{W}\cdot(i-1)+1}^{\frac{N}{W}\cdot i} x'_j \tag{2}$$

To discover similarities in time series, PAA uses a Euclidean distance (ED) based formula where, instead of the point-to-point calculation, it uses the mean value of the reduced series (3).

$$Distance(P,Q) = \sqrt{\frac{n}{w}} \cdot \sqrt{\sum_{i=1}^{w} (\bar{p}_i - \bar{q}_i)^2} \tag{3}$$

Although PAA enables a direct comparison between two series, it does not indicate whether the distance obtained represents a low or high level of similarity.

The Symbolic Aggregate approXimation (SAX) sax can be viewed as an improvement to PAA as it still uses this method to obtain a dimension reduced time series but adds a new type of data transformation, numeric to symbolic representation. This transformation relies on a normal distribution, $N\sim(0, 1)$, with intervals where the probability of each interval is equivalent to the difference between the z-score (β_i and β_{i+1}) of both intervals, which must be equal to $\frac{1}{\alpha_N}$, and to each interval a symbol is assigned. For example, for 3 intervals, all with probability of 33.3% and with a symbolic representation,

$$\begin{aligned} \alpha &= \text{'A'} \quad \texttt{iif} \quad -\infty < c_i < \beta_1 \\ \alpha &= \text{'B'} \quad \texttt{iif} \quad \beta_1 < c_i < \beta_2 \\ \alpha &= \text{'C'} \quad \texttt{iif} \quad \beta_2 < c_i < +\infty \end{aligned} \tag{4}$$

In Fig. 1, frame 3 (c_3) has an average value of 1.5 and considering an alphabet with 3 letters ($\alpha = 3$), it is possible to assess that, through Table 2, is between c_3 and β_2 and therefore, the corresponding symbol is C.

This method ensures that, in a fixed size alphabet, each SAX symbol has equal probability allowing a direct comparison and indication of the similarity level.