Available online at www.sciencedirect.com
SCIENCE DIRECT®
Information Processing Letters 93 (2005) 307–313
Information Processing Letters
www.elsevier.com/locate/ipl

# A Coarse-Grained Multicomputer algorithm for the detection of repetitions

Thierry Garcia, David Semé *

*LaRIA: Laboratoire de Recherche en Informatique d'Amiens, Université de Picardie Jules Verne, CURI, 5, rue du Moulin Neuf, 80000 Amiens, France*

Received 17 October 2003

Available online 11 January 2005

Communicated by F. Dehne

**Abstract**

The paper presents a Coarse-Grained Multicomputer algorithm that solves the problem of detection of repetitions. This algorithm can be implemented in the CGM model with $P$ processors in $O(N^2/P)$ in time and $O(P)$ communication steps. It is the first CGM algorithm for this problem. We present also experimental results showing that the CGM algorithm is very efficient.


*Keywords:* Parallel algorithms; Coarse-Grained Multicomputers; Dynamic programming; String matching

## 1. Introduction

This paper describes a Coarse-Grained Multicomputer solution for the detection of repetitions. Strings of symbols containing no consecutive occurrences of the same pattern have attracted the attention of researchers in diverse fields for a long time. For example, repetitions play a crucial role in many domains [3,19] such as formal language theory, term rewriting, data compression and computational molecular biology. Perhaps their first appearance dates back to the work of Thue [23], who is generally credited with the discovery of arbitrary long streams of symbols from a finite alphabet that do not contain any "square" substrings, i.e., subpatterns formed by the concatenation of some substring with itself.

In recent years, the study of such "square-free" strings has been found relevant to automata and formal language theory, algebraic coding and more generally in systems theory and combinatorics, and we shall make no attempt to refer to the existing copious literature. Suffice it to mention that papers have been devoted to the construction of arbitrarily long square-free (as well as other related repetition-constrained)

* Corresponding author.
*E-mail addresses:* garcia@laria.u-picardie.fr (T. Garcia), seme@laria.u-picardie.fr (D. Semé).

0020-0190/$ – see front matter 
doi:10.1016/j.ipl.2004.12.004

strings [18] over alphabets of fixed cardinality. In a related endeavor, the complementary notions of periodicity and overlaps of strings have been extensively investigated and still are an active research subject (see Duval [11] for an extensive bibliography).

In the framework of pattern matching [2], some classic results on string periodicity [13] have been used to develop clever techniques for the detection of assigned patterns in text-strings in time linear in the string length [7].

The problem of the efficient recognition of the occurrence of substring squares in a string $X$ stems quite naturally from the preceding remarks, and it is certainly relevant to a variety of practical applications as well [1]. $O(N^2)$ time algorithms can be quickly developed on the basis of existing pattern matching techniques and tools ($N$ is the length of the string $X$). Recently, an $O(N \log N)$ algorithm has been proposed to determine whether a given text-string over a finite alphabet contains a repetition [21]. Crochemore [5] developed an $O(N \log N)$ algorithm to determine all repetitions in a text-string $x$. Crochemore's approach essentially relies on the well-known minimization algorithm for finite state automata [2] and exploits the theoretical bound of $N \log N$ repetitions in a string [20] as a terminating condition. Apostolico and Negro [4] presented a systolic algorithm for the detection of repetitions running in time $O(N)$ (with $4N - 3$ steps) and using $N$ processors, whereas the linear systolic array for the same problem presented in [22] requires $(5N/4) - 1$ steps on $N/4$ processors. The only known algorithm for the PRAM model can be obtained by simulating the linear systolic array on the PRAM. A constant time parallel detection of repetitions with a BSR (Broadcasting with Selective Reduction) solution has been developed by [10]. All known parallel algorithms require a work of $O(N^2)$.

In this paper we show an algorithm which solve this problem in the CGM model but this algorithm is not work optimal. Indeed, the optimality in work to solve this problem is $O(N \log N)$. Here, our algorithm has a work of $O(N^2)$ but our goal is to obtain a CGM algorithm from a systolic solution having a work of $O(N^2)$. This is a contribution to a more larger work on the adaptation of linear systolic solutions in CGM model.

In recent years several efforts have been made to define models of parallel computation that are more realistic than the classical PRAM models. In contrast of the PRAM, these new models are coarse grained, i.e., they assume that the number of processors $P$ and the size of the input $N$ of an algorithm are orders of magnitudes apart, $P \ll N$. By the precedent assumption these models map much better on existing architectures where in general the number of processors is at most some thousands and the size of the data that are to be handled goes into millions and billions.

This branch of research got its kick-off with Valiant [24] introducing the so-called Bulk Synchronous Parallel (BSP) machine, and was refined in different directions, for example, by Culler et al. [6], LogP, and Dehne et al. [9], CGM extensively studied in [8,12, 14–16].

CGM seems to be the best suited for a design of algorithms that are not too dependent on an individual architecture.

We summarize the assumptions of this model:

- all algorithms perform in so-called supersteps, that consist of one phase of interprocessor communication and one phase of local computation,
- all processors have the same size $M = O(N/P)$ of memory ($M > P$),
- the communication network between the processors can be arbitrary.

The goal when designing an algorithm in this model is to keep the individual workload, time for communication and idle time of each processor within $T/s(P)$, where $T$ is the runtime of the best sequential algorithm on the same data and $s(P)$, the speedup, is a function that should be as close to $P$ as possible. To be able to do so, it is considered as a good idea the fact of keeping the number of supersteps of such an algorithm as low as possible, preferably $O(M)$.

As a legacy from the PRAM model it is usually assumed that the number of supersteps should be polylogarithmic in $P$, but there seems to be no real world rationale for that. In fact, algorithms that simply ensure a number of supersteps that are a function of $P$ (and not of $N$) perform quite well in practice, see Goudreau et al. [17].

The paper is organized as follows. In Section 2, we present the detection of repetition problem. The CGM solution of the problem is described in Section 3. Section 4 presents some experimental results and the conclusion ends the paper.