High Performance CGM-based Parallel Algorithms for the Optimal Binary Search Tree Problem

Vianney Kengne Tchendji, Department of Mathematics and Computer Science, University of Dschang, Dschang, Cameroon
Jean Frederic Myoupo, University of Picardie Jules Verne, Amiens, France
Gilles Dequen, University of Picardie Jules Verne, Amiens, France

ABSTRACT

In this paper, the authors highlight the existence of close relations between the execution time, efficiency and number of communication rounds in a family of CGM-based parallel algorithms for the optimal binary search tree problem (OBST). In this case, these three parameters cannot be simultaneously improved. The family of CGM (Coarse Grained Multicomputer) algorithms they derive is based on Knuth’s sequential solution running in $O(n^2)$ time and $O(n^2)$ space, where $n$ is the size of the problem.

These CGM algorithms use $p$ processors, each with $O(n/p)$ local memory. In general, the authors show that each algorithms runs in $O\left(\frac{n^2}{g}\right) \times R(p,g)$ communications rounds. $g$ is the granularity of their model, and $R(p,g)$ is a parameter that depends on $p$ and $g$. The special case of $g = \sqrt{2p}$ yields a load-balanced CGM-based parallel algorithm with $\sqrt{2p}$ communication rounds and $O(n^2 / \sqrt{2p})$ execution steps. Alternately, if $g = p$, they obtain another algorithm with better execution time, say $O(n^2 / p)$, the absence of any load-balancing and $p$ communication rounds, i.e., not better than the first algorithm. The authors show that the granularity has a crucial role in the different techniques they use to partition the problem to solve and study the impact of each scheduling algorithm. To the best of their knowledge, this is the first unified method to derive a set of parameter-dependent CGM-based parallel algorithms for the OBST problem.

KEYWORDS

Bulk Synchronous Parallel, Coarse-Grained Multicomputer, Optimal Binary-Tree Search, Parallel Processing

1. INTRODUCTION

The Optimal Binary Search Tree (OBST) problem is of great interest in data processing, as well as in distributed and centralized environments. The input is a sequence of $n$ weighted keys ($k_1, k_2, \ldots, k_n$), which are to be placed into a binary search tree in such a manner that the expected search cost of the resulting binary search tree is minimized (Cormen et al., 2001). The classical sequential algorithm for this problem is based on the dynamic programming technique (Aho et al., 1974). It requires $O(n^3)$ time steps and $O(n^2)$ memory space. By using the monotonicity property of optimal
binary search trees (defined later), Knuth (1972) derived an $O(n^2)$ time steps algorithm in the same space. Yao (1982) obtained the same result thanks to the quadrangle inequalities. Eppstein et al. (1988) put forth an $O(n \log n)$ algorithm using the restrictive assumption of convexity, whereas the parallelization of the classical version was extensively utilized by the community of parallel processing researchers for different parallel computing models: Bradford (1994), Guibas, Kung and Thompson (1979), Tang and Gupta (1995), Karypis et al. (1994, 1996), Rytter (1988), and Kechid and Myoupo (2008). Little work has been produced for the parallelization of the Knuth approach. An efficient parallelization of this version in an abstract model, such as PRAM (Parallel Random Access Machine), is difficult. In PRAM, it is unknown how to use the monotonicity property to reduce the number of processors in poly-logarithmic-time computations (Karpinski et al., 1994, 1996).

In this work, we tackle the problem of parallelizing the OBST algorithm on the Bridging Coarse Grain BSP/CGM (Bulk Synchronous Parallel/Coarse Grained Multicomputer) model (Dehne et al., 1994, 1996; Cáceres et al., 1997; Cheatham et al., 1995; Valiant, 1990). CGM seems best suited for the design of algorithms that are not too dependent on an individual architecture. A BSP/CGM machine is a set of $p$ processors, each having its own local memory of size $m$ (with $O(m) >> O(1)$) and connected to a router able to deliver messages in point-to-point fashion. A BSP/CGM algorithm consists of alternating between local computations and global communication rounds. Each communication round consists of routing a single $h$-relation with $h=O(m)$. A CGM computation/communication round corresponds to a BSP super-step with communication cost $g \times m$ (Bradford, 1994). $g$ is the cost of the communication of a word in the BSP model. To produce an efficient BSP/CGM algorithm, designers tend to maximize speedup and minimize the number of communication rounds (ideally independent from the problem size, and constant in the optimum case).

There are CGM-based parallel algorithms for various problems modeled by dynamic programming, among others, such as the longest common subsequence problem (Alves et al., 2003, 2006; Garcia et al., 2003; Krusche et al., 2006; Rytter, 1988) and the string editing problem (Alves et al., 2002). The special cases of the MCOP (Matrix Chain Ordering Problem), OBST and OSPP (Optimal String Parenthesizing Problem), which are modeled by the same dynamic programming algorithm, each with its own specification, were tackled in Kechid and Myoupo (2008) and Tchendji and Myoupo (2012).

The CGM-based parallel algorithm for MCOP (Kechid and Myoupo, 2008) requires $p$ communication rounds and, at most, $O\left(\frac{n^2}{p}\right)$ time steps on $p$ processors. The one for OSPP in Tchendji and Myoupo (2012) runs in $O\left(\frac{n^2}{p}\right)$ time steps with $\sqrt{2p}$ communication rounds. The prior CGM algorithm for OBST (Kechid and Myoupo, 2008) requires $O\left(\frac{n^2}{p}\right)$ time steps with $p$ communication rounds. A serious drawback of these algorithms is that the loads of processors are unbalanced, and for two of them, a processor can detain up to $O\left(\frac{p}{2}\right)$ blocks in the worst case.

In this paper, we propose a general methodology to derive CGM parallel algorithms for the cost of the OBST problem. An algorithm is derived based on the parameter to be optimized. Therefore, load-balanced, efficient and minimum-communication-rounds algorithms are obtained. The best one with regard to load balancing, already obtained in Myoupo and Tchendji (2014), requires only $\sqrt{p}$ communication rounds and $O\left(\frac{n^2}{\sqrt{p}}\right)$ time steps. We also present a better one with regard to time steps, running in $O\left(\frac{n^2}{p}\right)$ with $p$ communication rounds and unbalanced loads. We highlight the
A Scalable Approach to Real-Time System Timing Applications
www.igi-global.com/chapter/scalable-approach-real-time-system/36426?camid=4v1a

Adaptive Grid Services
www.igi-global.com/chapter/adaptive-grid-services/58751?camid=4v1a