Parallel Range Searching in Large Databases Based on General Parallel Prefix Computation *

Chun-Hsi Huang †, ***Xin He*** ‡

General Prefix Computation (GPC) problem has been shown to be the kernel routine that captures the hardest part of parallel algorithms in fields such as computer graphics, medical imaging, databases, and computational geometry. We present the first computation and communication optimal algorithm on Coarse Grained Multicomputers (CGM) for solving GPC and, specifically, present its application on parallel range searching in large databases in the same computation and communication efficient manners.

General Prefix Computation (GPC, [3]) is a generic problem component that captures the most common, difficult kernel of many types of problems. The definition is as follows : Let $\{f(1), f(2), \cdots, f(n)\}$ and $\{y(1), y(2), \cdots, y(n)\}$ be two sequences of elements with a binary associative operator $*$ defined on the f elements and a linear order $\prec$ defined on the y elements. It is required to compute the sequence $\{D(1), D(2), \cdots, D(n)\}$ whose elements $D(m)$,$m = 1, 2, \cdots, n$ are defined as $D(m) = f(j_1) * f(j_2) * \cdots * f(j_k)$, where $j_1 < j_2 < \cdots < j_k$ and $\{j_1, j_2, \cdots, j_k\}$ is the sequence of indices such that $j_i < m$ and $y(j_i) \prec y(m)$ for $i = 1, 2, \cdots, k$.

*Computational resources and technical support provided by Center for Computational Research at SUNY-Buffalo (UB CCR).

†Department of Computer Science, State University of New York at Buffalo, Buffalo, NY 14260. Email: ch25@cse.buffalo.edu

‡Research supported in part by NSF grant CCR-9912418.

Department of Computer Science, State University of New York at Buffalo, Buffalo, NY 14260. Email: xinhe@cse.buffalo.edu

It has been shown that GPC can be applied to a wide variety of geometric (point set and tree) problems [3], including triangulation of point sets, two-set dominance counting, ECDF searching, finding two- and three-dimensional maximal points, and the (classical) reconstruction of trees from their traversals. Therefore, providing a portable parallel algorithm with predictable communication efficiency for GPC becomes an essential step for solving these applications on practical parallel computing platforms.

In this paper, we first present a computation and communication optimal parallel GPC algorithm on a general purpose parallel programming model, CGM (Coarse Grained Multicomputers, [2]). A CGM computer consists of p processors $P_0, \cdots P_{p-1}$, where each processor has $O(\frac{n}{p})$ local memory. The processors can be connected through any communication medium, i.e. any interconnection network or shared memory. Typically, the local memory is considerably larger than $O(1)$. A CGM algorithm consists of alternating local computation and global communication rounds. In a communication round, a single h-relation (with $h = O(\frac{n}{p})$) is routed, i.e., each processor sends $O(\frac{n}{p})$ and receives $O(\frac{n}{p})$ data. Therefore, the cost of each communication round is considered the same. The communication cost can be specified in terms of a single parameter, λ, the number of communication rounds. If the best possible sequential algorithm for a given problem takes $T_s(n)$ time, then ideally we would like to design a CGM algorithm using $O(1)$ communication rounds and $O(\frac{T_s(n)}{p})$ total local computation time.

Next we present the main algorithm, which uses the CGM GPC algorithm as a subroutine to solve the *range searching* problem in large databases [1]. The *range searching* problem is defined as follows : Let $Q = \{q_1, q_2, \cdots, q_n\}$ be a finite sequence of ordered pairs of real numbers. Each entry $q_i = (x_i, y_i)$ of Q can be viewed as Cartesian coordinates of a point in the plane. A rectangle G is given whose sides are parallel to the axes of the coordinates; thus, G is defined by the two intervals $[a, b]$ and $[c, d]$ on the x and y axes, respectively. For some functions f and a binary operator $*$, it is required to compute the quantity $f(i) * f(j) * \cdots * f(k)$, where $q_i, q_j, \cdots, q_k$ are those elements of Q which fall inside G. More generally, the elements of Q are ordered tuples of the form $(x, y, \cdots, z)$, and it is required to find those elements whose components $x, y, \cdots, z$, fall within given ranges specified in the query $G = \{[a, b], [c, d], \cdots, [e, f]\}$; that is, $a \leq x \leq b, c \leq y \leq d, e \leq z \leq f$.

Long communication latency and the overhead of synchronization tend to be the two critical factors that greatly affect the speedup of a parallel algorithm on present multiprocessor architectures. Our algorithm is intended to minimize both. Since CGM programming model conforms to most of the commercially available parallel machines, this algorithm is deemed highly portable. We conclude that scalable parallel algorithms, in both computation and communication efficient manners, for problems mentioned in this paper are indeed achievable.

Bibliography

[1] Selim G. Akl. *Parallel Computation, Models and Methods.* Prentice Hall, 1997.

[2] F. Dehne, A. Fabri, and A. Rau-Chaplin. Scalable Parallel Geometric Algorithms for Coarse Grained Multicomputers. In *Proc. 9th ACM Annual Computational Geometry*, 1993, 298-307.

[3] F. Springsteel and I. Stojmenovic. Parallel General Prefix Computations with Geometric, Algebraic, and Other Applications. *International Journal of Parallel Programming*, 18(6):485–503, 1989.