Parallel Construction of Conflict Graph for Phylogenetic Network Problem

D.S. Rao[1], G.N. Kumar[1], Dheeresh K. Mallick[2], and Prasanta K. Jana[1]

[1] Department of Computer Science and Engineering,
Indian School of Mines University, Dhanbad - 826 004, India
prasantajana@yahoo.com
[2] Department of Computer Science and Engineering,
Birla Institute of Technology, Mesra, Ranchi – 835 215, India
dkmallick@gmail.com

Abstract. Conflict graph is used as the major tool in various algorithms [14] - [18] for solving the phylogenetic network problem. The over all time complexity of these algorithms mainly depends on the construction of the conflict graph. In this paper, we present a parallel algorithm for building a conflict graph. Given a set of n binary sequences, each of size m, our algorithm is mapped on a triangular array in $O(n)$ time using $O(m^2)$ processors.

Keywords: Phylogenetic network, galled tree, conflict graph, parallel algorithm.

1 Introduction

A large number of databases are now available along with a massive volume of sequence data. In order to explore and analyze this massive data for many biological applications like phylogenetic inference, the use of high performance computing systems is inevitable. A phylogenetic tree that represents the evolutionary history of organisms has the drawback that it lacks the consideration of some important events such as genetic recombination, hybrid specifications, homoplasy and horizontal gene transfer. A perfect phylogeny problem, which states that each site mutates at most once can be solved in linear time for binary sequences under the infinite-sites assumption [1]. However, recombination is very important as it may provide some valuable clues such as locating the origin of the gene causing genetic diseases. The effects of avoiding recombination have been shown in [2] - [4]. The recombination event results from two individuals that lead a non-tree like structure. Therefore, a more biologically complete evolutionary model is a general network in which both the major evolutionary phenomena, i.e., mutation and recombination are taken care. This network is commonly known as phylogenetic network or ancestral recombination graph (ARG) in the population genetics literature. Hein [5], [6] introduced the problem of phylogenetic network with recombination. Wang et al. [11] showed that computing the minimum number of recombination is NP-hard. Other papers dealing with the lower bound computation on this number are due to Hudson and Kaplan [12], Myers and Griffiths

A. Ghosh, R.K. De, and S.K. Pal (Eds.): PReMI 2007, LNCS 4815, pp. 398 – 405, 2007.

[13], Song and Hein [7], [8]. Their algorithms are based on combinatorial methods and have exponential time complexity for the worst-case.

There have been many algorithms developed for reconstructing a phylogenetic network for a set of binary sequences. A comprehensive survey can be found in [9], [10]. Given a set of n binary sequences, each of size m, Wang et al. [11] gave an $O(nm + n^4)$ time algorithm to solve a special case of phylogenetic network problem called galled tree problem. In the recent years, several other algorithms have been developed for the same in which the structural properties of a conflict graph are used as the main tool. Gusfield et al. [14], [15] proposed an $O(nm + n^3)$ time algorithm with all-zero ancestral sequence. Later Gusfield [16] extended the results to the case when the ancestral sequence is not known in advance. They also reported a faster algorithm in [17] for site arrangement in gall that runs in $O(n^2)$ time. Bafna et al. [18] used the combinatorial properties of a conflict graph and developed an algorithm in $O(nm^2)$ time. However, the construction of the conflict graph is the principal computation that dominates the over all time complexity of the above algorithms. In this paper, we present a parallel algorithm for building the conflict graph with the motivation that the over all time complexity for solving the galled tree problem will be reduced. Our proposed algorithm requires $O(n)$ time on a triangular array using $O(m^2)$ processors. To the best of our knowledge no parallel algorithm has been developed for the same.

The rest of the paper is organized as follows. We describe the galled tree problem along with some basic terminologies in section 2. The proposed parallel algorithm is given in section 3 followed by the conclusions in section 4.

2 Basic Terminologies and Problem Definition

The following preliminaries will help in understanding our algorithm.

Definition 2.1 (*Phylogenetic network*). An (n, m) phylogenetic network (see Fig. 1) is a directed acyclic graph N that has exactly one root node, a set of internal nodes having indegree 1 or 2 and exactly n leaf nodes. A node with two incoming edges is called a recombination node. Each node is labeled with a binary sequence of length m starting with the root node labeled with all zero-sequence. Each edge except those entering into a recombination node is also assigned an integer (called site or column) within the range 1 to m. If e is an edge coming into a non-recombination node say, u, then the binary sequence (i.e., label) of u is obtained from u's parent by changing from 0 to 1 at position i where i is the integer assigned to the edge e. This corresponds to a mutation at site i on the edge e. Each recombination node w is associated with an integer r_w, $2 \le r_w \le m$. We call r_w as the recombination point for w. One of the two sequences labeling the parents of w is designated as P (to mean prefix) and the other as S (suffix). Then the sequence labeling w is formed by concatenating the first r_w -1 bits of P and the last $m - r_w + 1$ bits of S.

Given a set of binary sequence M (matrix of size $n \times m$), a phylogenetic network N derives M if and only if each sequence in M labels exactly one of the leaves of N. As

an example, a phylogenetic network deriving a binary matrix is shown in Fig. 1 for n = 7 and m = 5. The interpretation of binary sequences and its motivation is discussed in [15].

Definition 2.2 (*Galled tree*). In a phylogenetic network two paths out of a node meeting at a recombination node can form a cycle called recombination cycle and whenever a recombination cycle has no node common with any other recombination cycles it is called a gall. A phylogenetic network where every recombination cycle is a gall is called a galled tree.

Definition 2.3 (*Galled tree problem*). Given a set of binary sequences M, it is to determine whether there exists a galled tree that derives M and if it does, construct it.

Definition 2.4 (*Conflict*). We say that two columns in M conflict each other if and only if they have three rows with the combinations (0,1), (1,0) and (1,1) and a column is called conflicted if it has conflict with at least one other column; otherwise it called unconflicted.

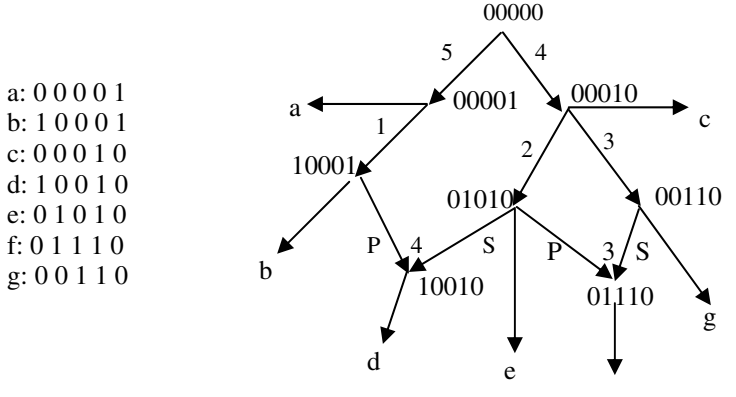


Fig. 1. A phylogenetic network deriving a set of binary sequences (shown in left)

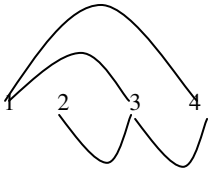
Definition 2.5 (*Conflict graph*). A conflict graph G is formed with all the sites in M where each node is labeled by a distinct site and there exist an undirected edge $<\alpha, \beta>$ if and only if the sites α and β conflict (see Fig. 2).

A connected component in G is the maximal subgraph of G such that for any pair of nodes in G there is at least one path between those nodes in G. A trivial connected component has only one node and has no edge and the site associated with that node is unconflicted. Note that the conflict graph shown in Fig. 2(b) has a single nontrivial connected component that consists of all the sites and there is no trivial connected component. In other words, there is no unconflicted site.

We now state some established theorems with the context of conflict graphs. A phylogenetic network is called perfect if it has no recombination node. The following theorem gives the necessary and sufficient conditions for the existence of a perfect phylogenetic network.

$$M = \begin{bmatrix} 1\,0\,1\,0 \\ 0\,1\,0\,0 \\ 1\,0\,1\,1 \\ 0\,1\,1\,0 \\ 0\,0\,1\,1 \\ 1\,0\,0\,1 \end{bmatrix}$$

(a) A set of binary sequences



(b) Conflict graph

Fig. 2. Conflict graph corresponding to the set of binary sequences M

Theorem 2.1. There exists a perfect phylogenetic network that derives M if and only if there is no conflicted site in M. Moreover, if there is a perfect phylogenetic network and all columns are distinct, then there is a unique phylogenetic network for M and each edge is also uniquely labeled. If there are identical columns then the perfect phylogeny is unique up to any ordering given to multiple sites that label the same edge [1], [2].

The following theorems show the one-one correspondences between the nontrivial connected components of a conflict graph and the galls.

Theorem 2.2. Each gall in a phylogenetic network with conflicted sites contains all the sites of one non-trivial connected component but no sites from a different non-trivial component [15].

Theorem 2.3. If there is a galled tree for M, then every non-trivial connected components of the conflict graph must be bipartite, and the bipartition is unique, i.e., the sites on one side of the bipartite graph must be strictly smaller than the sites on the other side [15].

Thus the above theorems imply that the construction of the conflict graph is the major computation towards the solution of the phylogenetic netowrk/ galled tree problem.

3 Proposed Algorithms

Let us represent the binary matrix M as follows

$$M = \begin{bmatrix} a_{11} & a_{12} \cdots & a_{1m} \\ a_{21} & a_{22} \cdots & a_{2m} \\ \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} \cdots & a_{nm} \end{bmatrix}$$

In the construction of the conflict graph, we need to compare each column with all its subsequent columns. As the size of the matrix is $n \times m$, it requires $n\left[\frac{m(m-1)}{2}\right]$, i.e., $O(nm^2)$ time. It can be noted that conflict checking between any pair of columns say, i and j is same as that of between j and i. Further, we do not require conflict checking of i with itself. Therefore, a triangular array of $\frac{m(m-1)}{2}$ processors will suffice the conflict calculation. Such a triangular array for $n = 6$ and $m = 4$ is shown in Fig. 3 in which a single '*' indicates one unit delay. The columns of the matrix are fed accordingly. Note that we start inputting from column 1 row wise while we start from column 2 column wise in the triangular array. We label the processor with a pair of indices (i, j) according to the columns i and j of the matrix M that are fed to the processor. Assume that each processor has some local registers. We use three flag registers $C1$, $C2$ and $C3$ to indicate the pattern 01, 10 and 11 respectively. At the end of the algorithm, the result of conflict is stored in the status register S, which is basically the adjacency matrix representation of the conflict graph.

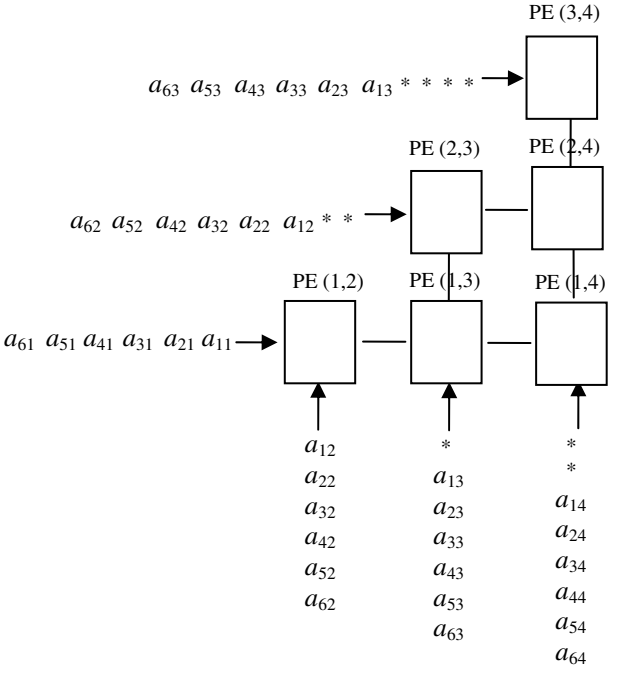


Fig. 3. Triangular array for computing the conflict graph of 6×4 matrix

The basic idea of our algorithm is as follows. Initially all the flag registers are set to 0. The columns of the matrix M are fed through the boundary processors. On receiving two inputs, the content of $C1$ / $C2$ / $C3$ is OR'ed (logical OR) with '1'

depending on the input patterns 01, 10 or 11 respectively. It then sends the inputs to the corresponding next row / column processor. This method is continued until the processing is reached to the processor PE(m -1, m). Finally we obtain the conflict result between the columns i and j by AND operation on the contents of $C1$, $C2$ and $C3$ registers. The algorithm is given stepwise as follows.

Step 1. /* Initialization of the flag registers */
for all PE(i, j),$1 \le i < j \le m$ *do in parallel*
$C1(i, j) := 0$;
$C2(i, j) := 0$;
$C3(i, j) := 0$
end forall
Step 2. /* Conflict calculations */
for all PE(i, j), $1 \le i < j \le m$ *do in parallel*
while PE(i, j) receives two inputs a (from a row)and a'(from a column) *do*
(i) *if* (a = '0' *AND* a' = '1') *then*
$C1(i, j) := C1(i, j)$ *OR* '1';
else if (a = '1' *AND* a' = '0') *then*
$C2(i, j) := C2(i, j)$ *OR* '1';
else if (a = '1' *AND* a' = '1') *then*
$C3(i, j) := C3(i, j)$ *OR* '1';
(ii) *if* ($i < m$) *then*
send a' to PE($i + 1, j$)
if ($j < m$) *then*
send a to PE($i, j + 1$)
end while
end forall
Step 3. /* Output */
for all PE(i, j),$1 \le i < j \le m$ *do in parallel*
$S(i, j) := C1(i, j)$ AND $C2(i, j)$ AND $C3(i, j)$
if $S(i, j) := 1$ *then* write ("There is a conflict between i and j ")
end forall
Step 4. Stop

The above algorithm is illustrated in Fig. 4 for the matrix given in Fig. 2. The contents of the flag registers after 1st, 2nd and final clock are shown in Fig. 4(a), 4(b) and Fig. (c) respectively. At the end of final clock (shown in Fig. 4 d), '1' is stored in the status registers $S(1, 3)$, $S(1, 4)$, $S(2,3)$ and $S(3,4)$ which indicates the conflict between the pair of columns (1, 3), (1,4), (2,3) and (3,4) respectively.

Complexity: Step 1 and Step 3 require constant time. In step 2, the elements $a_{n\ m\ -1}$ and $a_{n\ m}$ take $2m + n - 4$ communication time from the beginning of the computation to its termination reaching the last processor PE(m -1, m). Since m can be at most $2n$ if there exists a galled tree [11], the time complexity of the above algorithm is $O(n)$.

It is important to note that the above algorithm can work independent of the value of *n*, i.e., the number of rows (binary sequences)

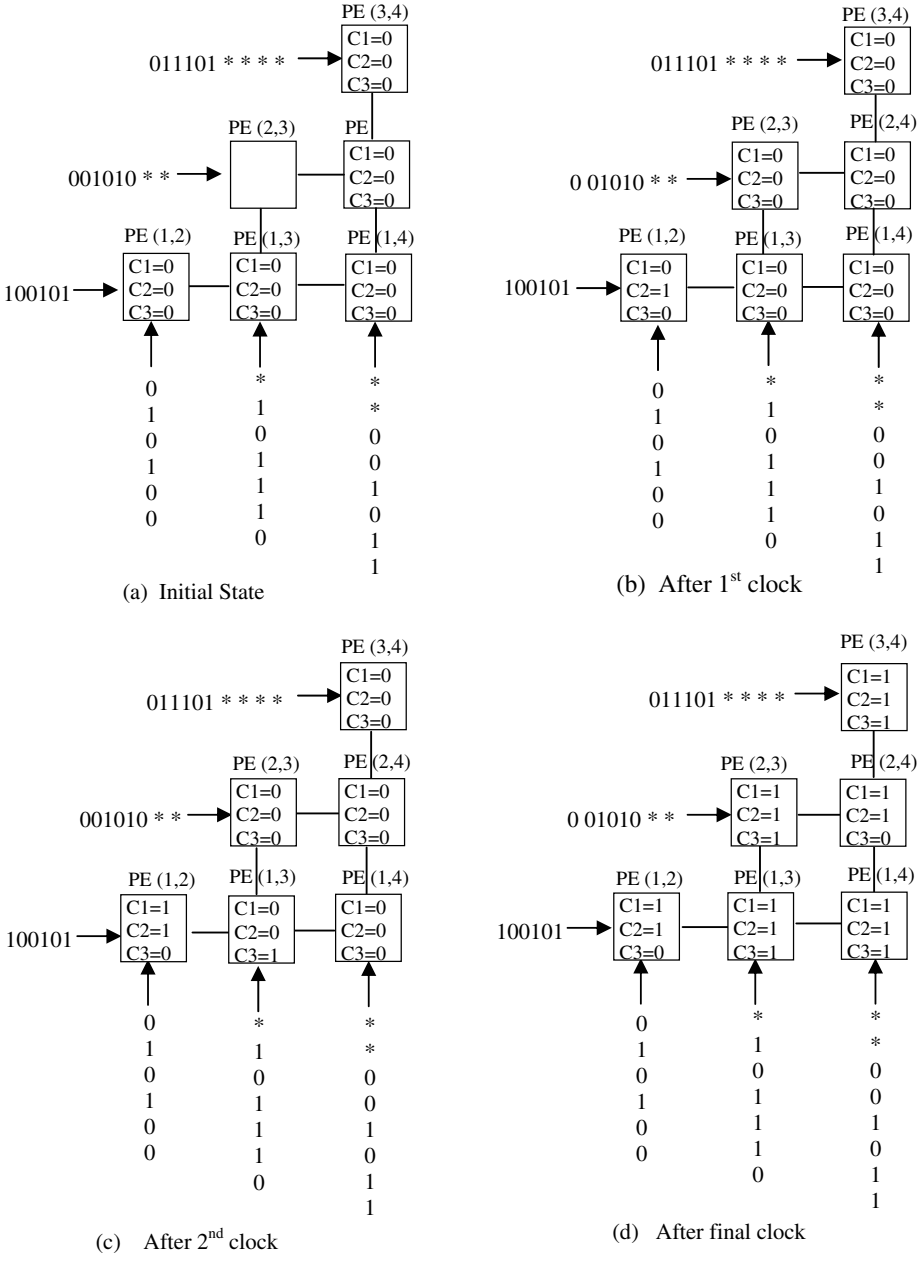


Fig. 4. An example of conflict computation

4 Conclusions

We have presented a parallel algorithm for the construction of a conflict graph, which is used as a major tool to solve a galled tree problem. Given set of n binary sequences, each of size m, our algorithm is mapped on a triangular array in $O(n)$ time using $O(m^2)$ processors and shown to be scalable with respect to n.

References

[1] Gusfield, D.: Efficient algorithms for inferring evolutionary trees. Networks 21, 19–28 (1991)
[2] Schierup, M.H., Hein, J.: Consequences of recombination on traditional phylogenetic analysis. Genetics 156, 879–891 (2000)
[3] Schierup, M.H., Hein, J.: Recombination and the molecular clock. Mol. Biol. Evol. 17, 1578–1579 (2000)
[4] Posada, D., Crandall, K.: The effect of recombination on the accuracy of phylogeny estimation. Journal of Molecular Evolution 54, 396–402 (2002)
[5] Hein, J.: Reconstructing evolution of sequences subject to recombination using parsimony. Math. Biosci. 98, 185–200 (1990)
[6] Hein, J.: A heuristic method to reconstruct the history of sequences subject to recombination. J. Mol. Evol. 36, 396–405 (1993)
[7] Song, Y., Hein, J.: On the minimum number of recombination events in the evolutionary history of DNA sequences. Journal of Mathematical Biology 48, 160–186 (2003)
[8] Song, Y., Hein, J.: Parsimonious reconstruction of sequence evolution and haplotype blocks: Finding the minimum number of recombination events. In: Proc. of 2003 Workshop on Algorithms in Bioinformatics, Berlin, Germany (2003)
[9] Linder, C.R., Moret, B.M.E., Nakhleh, L., Warnow, T.: Reconstructing networks part II: computational aspects. In: the ninth pacific symposium on Biocomputing (2004)
[10] Zahid, M.A.H., Mittal, A., Joshi, R.C.: Use of phylogenetic networks and its reconstruction algorithms. Journal of Bioinformatics India 4, 47–58 (2005)
[11] Wang, L., Zhang, K., Zhang, L.: Perfect phylogenetic networks with recombination. Journal of Computational Biology 8, 69–78 (2001)
[12] Hudson, R.R., Kaplan, N.L.: Statistical properties of the number of recombination events in the History of a sample of DNA sequences. Genetics 111, 147–164 (1985)
[13] Myers, S.R., Griffths, R.C.: Bounds on the minimum number of recombination events in a sample history. Genetics 163, 375–394 (2003)
[14] Gusfield, D., Satish, E., Langley, C.: Efficient reconstruction of phylogenetic networks (of SNPs) with constrained recombination. In: Proceedings of 2nd CSB Bioinformatics Conference, Los Alamitos, CA (2003)
[15] Gusfield, D., Satish, E., Langley, C.: Optimal efficient reconstruction of phylogenetic networks with constrained recombination. Journal of Bioinformatics and Computational Biology 2, 173–213 (2004)
[16] Gusfield, D.: Optimal, efficient reconstruction of root-unknown phylogenetic networks with constrained recombination, Technical Report, Department of Computer Sc., University of California, Davis, CA
[17] Gusfield, D., Satish, E., Langley, C.: The fine structure of galls in phylogenetic networks. Informs Journal on Computing 16, 459–469 (2004)
[18] Bafna, V., Bansal, V.: The number of recombination events in a sample history conflict graph and lower bounds. IEEE Ttrans. on Computational Biology and Bioinformatics 1, 78–90 (2004)