Optimal Total Exchange in Anonymous Cayley Graphs

Sanghoun Oh[1], Adnan Mohamed[1], and Moongu Jeon[1*]
[1]*Department of Information and Communications,*
Gwangju Institute of Science and Technology,
1 Oryong-dong, Buk-gu, Gwangju 500-712, Rep. of Korea
{oosshoun,adnan,mgjeon}@gist.ac.kr

Abstract

Total exchange or all-to-all personalized communication problem is that each node in a network has a message to be sent to every other node. To solve this communication problem, we present a time-optima algorithm in anonymous Cayley graphs as assuming a single-port full duplex model in which every node is able to send and receive at most one message in each time unit.

Keywords: *Cayley Graphs, Total Exchange Algorithm, Optimality.*

1. Introduction

Total exchange wherein each node in a network sends a message to every other node has been studied in Cayley graphs. To effectively discover an optimal solution in its graphs, one algorithm has also been proposed [1]. This algorithm supposes that each unique identity of nodes in the network be the same as the elements of the group generated by the graph. In this paper, we preserve the same assumption of the exclusive identifier of each node.

Well-known applications of total exchange such as matrix transposition and related communication problems are encountered in scientific computing. The matrix transposition problem requires that each node should be defined as a unique identification. Because of two reasons, this requirement is important to solve the problem in anonymous systems. Firstly, it is clearly restrictive, not general enough. Secondly, the assumed globally unique identifiers may not be of a specific form stated above [1].

In view of these observations, we exploit the symmetry of Cayley graphs that are supposed to have allocated a sense of direction. It may be noted that existence of the optimal solution in the anonymous case is also of theoretical interest.

2. Model Description

Cayley graphs have been studied in the context of interconnection networks since they are highly symmetric and easy to reason about [5]. An example of Cayley graphs is illustrated in Figure. 1.

Given a group G and a set S of generators of G (not containing the identity element of G) such that $\forall \mathrm{s_i} \in \mathrm{S}, \forall {\mathrm{s_i^{-1}}}_{\mathrm{i}} \in \mathrm{S}$, the Cayley graph $\Gamma = \mathrm{Cay(G,S)}$ is defined as the graph whose vertex set is the set of elements of G and two vertices $\mathrm{u,v}$ in Γ are adjacent $\mathrm{v} = \mathrm{s_i u}$ for some $\mathrm{s_i} \in \mathrm{S}$. Because S is closed under taking inverses, the graph is essentially undirected, i.e., u is adjacent to v iff v is adjacent to u. For simplicity, we utilize the same symbols of vertices in Cayley graph and each element of the group from which it is generated. However, the object, to which the symbol refers, will be clear from the context.

In the area of distributed algorithms, graphs are widely used as abstractions of networks to express interconnected processes. There is usually no specific need to make any distinction between graphs (i.e., networks): the terms of *node*, *vertex* and *process* will be interchangeably transformed into *link* and *edge*.

It is assumed that the edge (u, v) is given the label $s_i = vu^{-1}$ by u and s_i^{-1} by v, respectively. The labels on links adjacent to a process can be used in computations.

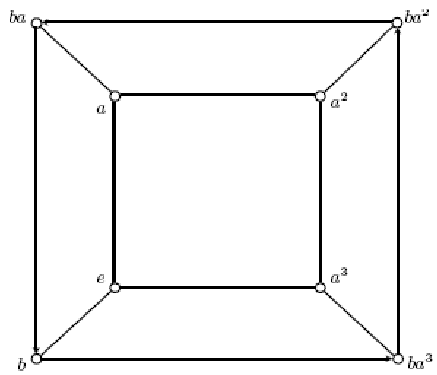

Figure. 1. Example of Cayley Graph

Now, the vertices of the graph can be labeled with the members of the group which generates it in many ways (as many as the number of automorphisms of the graph which preserve the edge labels).

In this paper, we do not presume any fixed labeling of the graph. Each node assumes itself to be the identity and labels the entire graph from this perspective. The labels themselves are derived from the edge labels such that the label $\lambda_u(v)$ given to v by u is equal to the product of the edge labels on any path from u to v in the order of traversal.

We design a distributed system whose communication graph is a Cayley graph. Each vertex and each edge of the graph represent a process and a link between two processes, respectively. Both processes are capable of directly communicating with them by sharing their only links. Time is divided into rounds (i.e., time units). During each round, the process can send and receive at most one message (i.e., *single-port full duplex*). Further, a message requires one time unit to be transferred between two adjacent processes.

3. Related Work

Some existing models of optimal total exchange algorithms have been proposed by hypercubes [2], star graphs [3] and general Cartesian product networks [4]. Another optimal algorithm for general Cayley graphs has also been reported [1]. Its proposed algorithm assumes that the nodes of the graph have unique identities as specially labeling nodes with the elements of the group created by the graph in computations. In this paper, we give an optimal algorithm for total exchange in general Cayley graphs without uniquely identified nodes. We only assume that the edges of the graph are labeled with the group elements that generate those edges. Removing the assumption of unique identifiers leads to a more simplified algorithm and analysis besides allowing total anonymity of nodes in a distributed system.

4. Total Exchange Algorithm

4.1. Notations

f is a function, which is given a message queue, returns the message at the head of the queue. $m.scr$ and $m.dest$ refer to the source and destination labels of message m, respectively. $receive$ returns the message received. e denotes the identity element of the group that is generated from the graph.

For purposes of analysis, we define two messages of m_1 and m_2 to be identical iff $m_1.src = m_2.src$ and $m_1.dest = m_2.dest$. Two message queues are said to be identical iff they have equal messages in the same sequence. Let Q be a message queue, m an arbitrary message and m_h the message at the head of Q. Then, $Q \cup m$ is the new queue result when m is inserted at the head of Q. On the other hand, $Q - m_h$ is the new outcome when m_h is removed from Q. Finally, $Q_u(t)$ is the message queue at node u at the beginning of round t.

4.2. Lower Bound

It has already been shown that total exchange in Cayley graphs takes at least $s(u)$ time units, where $s(u)$ is the status or the total distance of any node u of the graph [1, 6]. Note that Cayley graphs are vertex transitive and thus all nodes have the same status. It implies that $s(u)$ is a constant independent of node u. Further, it is known [1] that in order to achieve the above lower bound, it is sufficient that at every time unit, every node sends a message and receives a message and that every transmitted message gets closer to its destination.

4.3. Proposed Algorithm

We assume that a function $N(\cdot)$ is available that maps destination labels to an outgoing edge on the shortest path to that destination. This function is the same at whole nodes.

Let u be an arbitrary node. Initially, u has a message $m_u(v)$ to be sent to v, for all v except u itself. The message $m_u(v)$ is tagged with the destination label $\lambda_u(v)$ and the source label which is equal to identity ($\lambda_u(u) = e$). All messages in queue are arranged according to the order of the destination labels. This ordering is the same at all the nodes, but otherwise arbitrary. For instance, the local labels could be sorted lexicographically following a total order on sequences of edges (and hence paths) induced by an ordering of the generators.

During every time unit, u selects the message at the head of the queue to be sent, and then uses function N to find an outgoing edge of g on the shortest path to its destination. It then updates the source and destination labels of the message and then sends the message on the edge labeled g. These steps area repeated as long as there are messages to be sent. When a message is received on the edge labeled h, u checks the destination label of the message to find out whether it is the identity or not. If the destination is indeed the identity, then the message is meant for u. Otherwise, the message is placed at the head of the queue.

Algorithm 1. Total Exchange algorithm at node u.

```
while Q ≠ φ do
   m = f(Q)
   Q = Q − m
   g = N(m.dest)
   m.dest = m.dest * g^-1
   m.src = m.src * g^-1
   send m on edge g
   m = receive
   if m.dest ≠ e then
      Q = Q ∪ m
   else
      save m for local consumption
   end if
end while
```

4.4. Optimality

In order to prove that the algorithm terminates in $s(u)$ time steps, we need to show two things; firstly, at every time step, each node sends a message and receives a message and each message sent gets closer to its destination. The second property is easily seen to be true since each message is always sent via an edge on the shortest path to its destination.

To show that the first property holds, we need to show that at each time step, each node is selected by exactly one neighbor to forward a message. This means that each node receives a message. Note that a node always sends a message as long as its message queue is not empty. Obviously, if each node receives a message, and if each node sends at most one message, then each node sends exactly one message. Further, we will show that at each time step, the message queues of all nodes is identical and hence, when the message queue of a node is empty, there are no further messages to be sent or received by any node.

Lemma 1. If the message queues of all nodes are identical and non-empty at the beginning of round t, then in round t, each node sends exactly one message and receives exactly one message.

Proof. Let $Q_u(t) \neq \phi$ be independent of u at the beginning of round t. Then so are $m = f(Q_u(t))$ and $g = N(m.dest)$. Thus, each node selects the edge labeled by g, and if a node w is selected by one of its neighbors u, i.e., u selects the edge (u, w), then edge (w, u) is labeled g^{-1} by w. Since each edge incident on a node is given a unique label by this node, it follows that at most one neighbor selects w. Further, there exists an edge labeled g^{-1} incident on any given node. The other end of this edge is labeled g. Thus, each node receives exactly one message and sends exactly one message in round t.

Lemma 2. At any given round, message queues of all nodes are identical.

Proof. We proceed by induction on the number of rounds t. By definition, $Q_u(0)$ is independent of u. Assume that $Q_u(t)$ is independent of u. Then, $Q_u(t+1) = Q_u(t) - f(Q_u(t)) \cup m$, where $m = \phi$ if $f\big(Q_u(t)\big).dest \times N(f\big(Q_u(t)\big))^{-1} = e$. Otherwise, m is a message such that $\mathrm{m.dest} = f(Q_u(t)).\mathrm{dest} \times N(f(Q_u(t)))^{-1}$, and $\mathrm{m.src} = f(Q_u(t)).\mathrm{src} \times N(f(Q_u(t)))^{-1}$. In either case, $Q_u(t+1)$ is independent of u.

Theorem 1. Algorithm 1 solves total exchange in s(u) rounds.

Proof. The total distance that must be traversed by all the messages originating at node u is given by $s(u) = \sum_{d=1}^{e(u)} d.n_{d,}$ where n_d is the number of nodes at distance d from u and e(u) is the eccentricity of u.

Since Cayley graphs are vertex transitive and thus s(u) is independent of u, the total distance that must be traversed by all the messages is N.s(u) [1], where N is the number of nodes in the graph. By Lemma 1, at any round, if the message queues are not empty, N messages are transferred each of which gets one link closer to its destinations. This means after each round, N is subtracted from the total distance that must be traversed by messages. Therefore, after s(u) rounds, all the messages reach their destinations. By Lemma 2, when the message queue of any node is empty, so are the message queues of all the other nodes. Therefore, when the message queue of a node is empty at the beginning of some round, it has received all the messages meant for it, and so have all the other nodes. From the discussion above, this happens precisely after s(u) rounds.

5. Conclusion

In this paper, we propose the optimal time total exchange algorithm for anonymous Cayley graphs. Removing the assumption that processes have unique identifiers simplifies the algorithm and its analysis. Anonymity is more inclusive and desirable. Further, even when nodes do have unique identifiers, they may not be of a particular form making it difficult to exploit these identifiers in total exchange.

Acknowledgement

This research was supported in part by the Center for Distributed Sensor Network, GIST

References

[1] V. V. Dimakopoulos, and N. J. Dimopoulos, “Optimal Total Exchange in Cayley Graphs”, IEEE Trans. Parallel and Distributed Systems. 12(11), 2001, pp. 1162-1168.

[2] D. P. Bertsekas, and J. N. Tsitsiklis, “Parallel and Distributed Computation: Numerical Methods”, Englewoods Cliffs, N. J, Prentice Hall, 1989.

[3] J. Missic, and Z. Jovanovic, “Communication Aspects of the Star Graph Interconnection Network”, IEEE Trans. Parallel and Distributed Systems. 5(7), 1994, pp. 678-687.

[4] V. V. Dimakopoulos, and N. J. Dimopoulos, “A Theory for Total Exchange in Multidimensional Interconnection Networks”, IEEE Trans. Parallel and Distributed Systems. 9(7), 1998, pp. 639-649.

[5] S. B. Akers, and B. Krishnamurthy, “A Group Theoretic Model for Symmetric Interconnection Networks”, IEEE Trans. Computers. 38(4), 1989, pp. 555-566.

[6] F. Buckley, and F. Harary, “Distance in Graphs, Reading, Mass”, Addison-Wesley, 1990.

Authors

Sanghoun Oh received the B.S. degree in computer engineeering from Dong-A University, Korea in 2005, and then received the M.S. degree from the Department of Information and Communications, Gwangju Institute of Science and Technology (GIST) in 2007. Currently he is a Ph.D. candidate in School of Information and Communications, Gwangju Institute of Science and Technology. His main research areas are evolutionary algorithms and machine learning. In particular, he has been working on genetic programming and estimation of distribution algorithms.

Adnan Mohamed received the M.S. degree from the Department of Information and Communications, Gwangju Institute of Science and Technology (GIST) in 2004. His main research areas are the graph theory and parallel programming.

Moon-Gu Jeon received the B.S. degree in architectural engineering from the Korea University, Korea in 1988. He then received both the M.S. and Ph.D. degrees in computer science and scientific computation from the University of Minnesota in 1999 and 2001, respectively. Currently, he is an Associate Professor at the School of Information and Communications, Gwangju Institute of Science and Technology (GIST). His current research interests are in algorithms, machine learning and pattern recognition and evolutionary computation.

* Corresponding author