Computers Math. Applic. Vol. 27, No. 3, pp. 21–25, 1994
Printed in Great Britain.

0898-1221/94 $6.00 + 0.00

Optimal Data Exchange Algorithms on Star Graphs

K. COOLSAET
Department of Pure Mathematics, University of Gent
Galglaan 2, B–9000 Gent, Belgium

V. FACK
Department of Applied Mathematics and Computer Science, University of Gent
Krijgslaan 281–S9, B–9000 Gent, Belgium

(Received and accepted August 1993)

Abstract—Star graphs, as discussed in [1], are considered to be attractive alternatives for hypercubes. In this paper, we discuss optimal data exchange algorithms for star graphs of small dimension ($n \leq 6$). In particular we study odd-distance and total exchange algorithms, using the tabular method introduced in [2]. The algorithms use no intermediate buffering of messages.

Keywords—Total exchange, Odd-distance data exchange, Optimal algorithm, Star graph.

1. DATA EXCHANGE ALGORITHMS ON CAYLEY NETWORKS

A network of parallel processors can be treated as an abstract *graph* with processors as *nodes* (vertices) and duplex communication channels as *links* (edges). A sequence of nodes $i_0, i_1, \ldots, i_k$ where i_j and i_{j+1} are neighbours $(0 \leq j < k)$ is a *path* of length k between nodes i_0 and i_k. The *distance* between two nodes is the length of the shortest path connecting these nodes.

A *data exchange algorithm* is a set of programs (one for each processor in the network) which are executed in parallel and whose purpose it is to send data (in the form of messages) between several processors. A *total exchange algorithm* is a data exchange algorithm which allows every node i in the network to send a different message $m_{i,j}$ to every other processor j in the network [3,4]. We assume that messages are sent only at discrete time intervals, that the processing time of messages within a node is negligible, and that a node can send a message to all its neighbouring nodes simultaneously. An algorithm does not need *intermediate buffering* of messages if every message $m_{i,j}$ which arrives at a node k $(\neq j)$ at a given time T, leaves that node again at time $T + 1$.

We call a network a *Cayley* network if a regular automorphism group G exists for the underlying graph [5,6]. Every Cayley network is uniquely determined by its group G and the set $S = \{g_1, \ldots, g_d\}$ of neighbours of the identity $1 \in G$. Two nodes $g, h \in G$ are neighbours if and only if $g^{-1}h \in S$. For each path $h_0, \ldots, h_n$ in a Cayley network there is a corresponding *word* $w = g_1 \cdots g_n \in S^*$ (i.e., a word with elements in S), where $g_i \stackrel{\text{def}}{=} h_{i-1}^{-1} h_i \in S$. We have $h_0 w = h_n$. Conversely, every word $w \in S^*$ determines a path joining a node h_0 with the node $h_0 w$.

A data exchange algorithm on a Cayley network (G, S) is *locally defined* iff for every $g \in G$ and for every message $m_{h,k}$ $(h, k \in G)$ sent from node 1 to its neighbour $g_i \in S$ at a given time T, a message $m_{gh,gk}$ is sent at the same time T from node g to its neighbour gg_i. Note that the paths taken by the messages $m_{h,k}$ and $m_{gh,gk}$ correspond to the same word w. Hence, in order

to study locally defined algorithms, it is only necessary to investigate what happens at a *single node* (usually the node $1 \in G$).

In [7], we have proved the following theorem:

THEOREM 1. *Consider a locally defined algorithm on a Cayley network (G, S). Let m be a message with associated word $w = g_i w' \in S^*$ sent from a node $h \in G$ at a given time. Then in the same time interval, a message m' arrives at h from the node hg_i^{-1}. The path which m' still has to travel to reach its destination corresponds to the word w'. (If w' is empty, then h is the destination of the message m'.)*

As a consequence, we have the following theorem [2], which can be used to investigate the traffic of messages in a data exchange algorithm:

THEOREM 2. *Every locally defined data exchange algorithm without intermediate buffering on a Cayley network (G, S) corresponds to a rectangular table with $\#S$ rows whose entries are either blank or elements of S with the following properties:*

(1) *Every row consists of zero or more words in S^* separated by zero or more blanks.*
(2) *Every column contains each element of S at most once.*

The messages sent by this algorithm are exactly those that correspond to the words in the table. The total time taken by the algorithm is equal to the number of columns in the table.

Conversely, every such table corresponds to a locally defined data exchange algorithm without intermediate buffering in the following way. Remove the first $T - 1$ columns of the table. The words that begin in the first column of the table thus obtained correspond to the messages which are transferred from a given node at time T. Each message is sent to the neighbour that is associated with the first element of the corresponding word.

2. STAR GRAPHS

Consider the group S_n of permutations of the n elements $0, \ldots, n-1$ and consider the subset $S \subseteq S_n$ of transpositions $(0\ a)$, $a = 1, \ldots, n-1$. The Cayley graph (S_n, S) is called the *star graph* of dimension n [1,8].

With every group element $g \in S_n$ (i.e., with every node) we may associate a word w in the following way:

— Write g as a product of disjoint cycles (e.g., $g = (0\ 1\ 2)(3\ 4) \in S_5$).
— Substitute each cycle of the form $(0\ a \cdots z)$ by the word $a \cdots z$ (e.g., $(0\ 1\ 2)$ is translated to 12).
— Substitute each cycle $(a\ b \cdots z)$ which does not contain 0, by the word $ab \cdots za$ (e.g., $(3\ 4)$ becomes 343).
— Concatenate various words obtained in this way to a single word (hence, $g = (0\ 1\ 2)(3\ 4)$ translates to $w = 12343$).

It is easily seen that a word w thus obtained corresponds to a message path from the identity node to node g. The length of this path is given by

$$|w| = \begin{cases} c + m & \text{if the permutation fixes the element 0,} \\ c + m - 2 & \text{otherwise,} \end{cases}$$

where c denotes the number of nontrivial cycles in g, and m the total number of elements not fixed by g. In [8] it was proved that this is exactly the distance between g and the identity, hence, these message paths are of minimal length.

Note that the translation above is nondeterministic: a given node may correspond to different words of minimal length. For example, the permutation $(0\ 1\ 2)(3\ 4)$ translates to all of the following words:

12343, 12434, 34312, 43412.

2.1. Star Graphs with $n = 3$ and $n = 4$

The star graph with $n = 3$ corresponds to the hexagonal network. Data exchange algorithms for this network have already been given in [2].

The star graph with $n = 4$ has the nodes listed in the following table, where the horizontal line separates even permutations from odd ones:

cycle structure	massage path words
identity	*empty*
$(a\ b\ c)$	12 23 31
	13 21 32
	1231
	3213
$(a\ b)(c\ d)$	1232 2313 3121
$(a\ b)$	1 2 3
	121 232 313
$(a\ b\ c\ d)$	123 231 312
	132 213 321

For each node, we have given a message word of minimal length. For nodes to which correspond several words, only one is given. Note that we have arranged the words in triplets in such a way that each word in the triplet is obtained from the previous one by applying the permutation $\sigma : 1 \to 2 \to 3 \to 1$. The only exceptions are 1231 and 3213 which are 'fixed' by σ. Indeed, 1231, $\sigma(1231) = 2312$ and $\sigma^2(1231) = 3123$ denote the same node (and likewise for 3213). This arrangement into triples has its importance in the construction of the algorithms below.

Let us first consider a data exchange algorithm in which every node sends a message to each node at *odd distance*. The messages sent from a given node therefore correspond to all odd permutations. The partitioning into triplets induced by σ suggests the following algorithm table:

1	1	2	1	1	2	3	1	3	2
2	2	3	2	2	3	1	2	1	3
3	3	1	3	3	1	2	3	2	1

This algorithm takes time $T = 10$ (i.e., the number of columns, cf. Theorem 2). As there are no empty entries, this algorithm is optimal.

A *total exchange* algorithm cannot as readily be constructed, because of the 'exceptional' words 1231 and 3213. However, these two words can be combined with the triplets $[1, 2, 3]$, $[12, 23, 31]$, and $[13, 21, 32]$ to yield the following table:

1	2	3	1	3	2	1	3
2	3	1	2	1	3	2	1
3	1	2	3		1	3	2

This table can now easily be extended to an algorithm table for a total exchange algorithm by adding the remaining triplets in the same way as before. This results in a total exchange algorithm taking time $T = 21$. Again this is an optimal algorithm, as there is only one empty entry, and therefore, less than 21 columns do not suffice.

2.2. Star Graphs with $n \geq 5$

Lists of message words for star graphs with $n \geq 5$ can easily be constructed (like we did above for $n = 4$), but quickly grow very large. However, as can be seen from the previous example, the design of a data exchange algorithm table hinges on the existence of a certain 'rotational'

symmetry induced by σ, which in the general case can be written as

$$\sigma : 1 \to 2 \to \cdots \to n-1 \to 1.$$

The important key to the construction of those tables is the knowledge of the 'exceptional' words, i.e., the words for which repeated application of σ does not yield a full $(n-1)$-tuple.

The following lemmas may help us determine these 'exceptional' words.

LEMMA 1. *The 'exceptional' words as defined above are those corresponding to permutations that commute with the permutation* $\sigma = (1\ 2\ 3\ \cdots\ n-1)$ *or with one of its powers.*

PROOF. It is easily seen that the action of σ on a word w corresponds to the conjugation of the corresponding permutation p by σ. Hence, applying σ i times to a word w will return a word corresponding to the same node, iff $p^{\sigma^i} = p$, i.e., iff p commutes with σ^i. ∎

LEMMA 2. *Consider a permutation* $\tau \in S_n$, *which fixes* 0 *and which is a product of* d *disjoint cycles of length* e, *with* $de = n-1$, *say,*

$$\tau = (a_0\ a_d\ a_{2d}\ \cdots)(a_1\ a_{d+1}\ \cdots)\cdots(a_{d-1}\ a_{2d-1}\ \cdots).$$

Then the permutations of S_n *commuting with* τ *form a group* $C(\tau)$ *generated by the cycles of the form*

$$(a_0\ a_d\ \cdots),\ (a_1\ a_{d+1}\ \cdots),\ldots$$

together with all permutations ρ *with the property* $\rho(a_i) = a_j$ *iff* $\rho(a_{i+d}) = a_{j+d}$.

PROOF. It is easily verified that the given permutations commute with τ. Indeed, permutations of the first kind simply rotate the elements of a cycle in τ, while permutations of the second kind simply permute the cycles of τ. The order of the group H generated by these elements is equal to $d!e^d$.

Now, the order of the centralizer $C(\tau)$ is equal to $n!$ divided by the number of conjugates of τ. The latter is equal to $n!/(d!e^d)$. Hence, the order of $C(\tau)$ is equal to the order of H, and hence, $H = C(\tau)$. ∎

As an example, we consider the star graph with $n = 5$, for which we have $\sigma = (1\ 2\ 3\ 4)$. By Lemma 2 the permutations commuting with σ are

$$(1\ 2\ 3\ 4),\ (1\ 3)(2\ 4)\ \text{ and }\ (1\ 4\ 3\ 2),$$

while the permutations commuting with $\sigma^2 = (1\ 3)(2\ 4)$ are

$$(1\ 3),\ (2\ 4),\ (1\ 3)(2\ 4),\ (1\ 2)(3\ 4),\ (1\ 4)(2\ 3),\ (1\ 2\ 3\ 4)\ \text{ and }\ (1\ 4\ 3\ 2),$$

and σ^3 yields the same permutations as σ.

These permutations correspond to the 'exceptional' words

$$12341,\ 131242,\ 14321,\ 131,\ 242,\ 121343,\ 141232.$$

Together with the 4-tuples $[1,2,3,4]$ and $[13,24,31,42]$, these can be arranged into an algorithm table in the following way:

1	2	1	4	3	4	3	1	3	2	4	2
4	3	2	1	4	1	2	3	4	1	1	3
3	1	3	3	2	3	1	4	1	4	2	4
2	4	4	2	1		4	2		3	3	1

This table can be extended to a full total exchange algorithm table by adding the remaining 4-tuples as usual. This yields an algorithm taking minimal time $T = 111$.

If we only consider messages of odd length, i.e., the odd-distance data exchange algorithm, then the 'exceptional' words are 12341, 14321, 131 and 242. Together with the 4-tuples $[1,2,3,4]$, $[124,231,342,413]$ and $[142,213,324,431]$, they result in the following algorithm table:

1	2	3	4	1	4	1	3	1	2	4
2	1	4	3	2	3	2	4	2	1	3
3	4	2	2	3	1	3	1	3	4	1
4	3	1	1	4	2	4	2	4	3	2

Together with the other 4-tuples, this leads to an optimal algorithm with $T = 56$.

A similar approach can be used to construct data exchange tables for still larger values of n. Because the constructions are rather *ad hoc*, it is not immediately clear whether an optimal algorithm exists for all values of n. However, in view of the large number of nodes ($n!$), we feel that data exchange algorithms for star graphs with $n > 6$ have little practical value.

An optimal total exchange table for $n = 6$ (and then $T = 689$) can be constructed from the following table, which is built around the exceptional words 123451, 253142, 352413 and 432154 together with all words of length ≤ 2:

2	4	1	2	3	4	5	1	4	3	2	1	5	4
3	5	2	4	1	3	2	5	3	1	4	2	1	5
4	1	3	1	2	5	1	2	5	4	3	5	2	1
5	2		3	4	2	3	4	2	2	5	3	3	2
1	3	4	5	5	1	4	3	1	5	1	4	4	3

Because all exceptional words are of even length, an odd-distance exchange algorithm is readily constructed. In this case, we have $T = 250$. In fact, for odd-distance exchange algorithms we have the following general theorem:

THEOREM 3. *On the n-star with $n-1$ an odd prime, there always exists an optimal data exchange algorithm without buffering in which every node sends a (possibly different) message to every node at odd distance.*

PROOF. $n-1$ is prime, and therefore, all powers of $\sigma = (1\ 2\ \cdots\ n-1)$ are cycles of length $n-1$. Applying Lemma 2 with $d = n-1$, $e = 1$ yields $C(\sigma) = \langle\sigma\rangle$, which contains only even permutations. Therefore, the 'exceptional' words are all of even length and an odd-distance exchange algorithm may simply be built by partitioning the odd length words according to σ. ∎

REFERENCES

1. S. Sur and P.K. Srimani, Topological properties of star graphs, *Computers Math. Applic.* **25** (12), 87–98 (1993).
2. K. Coolsaet, V. Fack and H. De Meyer, A tabular method for verification of data exchange algorithms on networks of parallel processors, (submitted).
3. D. Bertsekas and J. Tsitsiklis, *Parallel and Distributed Computation, Numerical Methods*, Prentice Hall International Editions, (1989).
4. Y. Saad and M. Schultz, Data communication in parallel architectures, *Parallel Computing* **11**, 131–150 (1990).
5. N. Biggs, *Algebraic Graph Theory*, Cambridge Tracts in Mathematics No. 67, Cambridge University Press, (1974).
6. F. Harary, *Graph Theory*, Addison-Wesley, (1969).
7. K. Coolsaet, H. De Meyer and V. Fack, Optimal algorithms for total exchange without buffering on the hypercube, *BIT* **32**, 559–569 (1992).
8. S.B. Akers, D. Harel and B. Krishnamurthy, The star graph: An attractive alternative to n-cube, *Proceedings of Intl. Conf. on Parallel Processing*, pp. 393–400, (1987).