Leader Election in Hyper-Butterfly Graphs

Wei Shi and Pradip K. Srimani

Department of Computer Science
Clemson University
Clemson, SC 29634 USA

Abstract. Leader election in a network is one of the most important problems in the area of distributed algorithm design. Consider any network of N nodes; a *leader* node is defined to be any node of the network unambiguously identified by some characteristics (unique from all other nodes). A leader election process is defined to be a uniform algorithm (code) executed at each node of the network; at the end of the algorithm execution, exactly one node is elected the *leader* and all other nodes are in the non-leader state [GHS83, LMW86, Tel93, Tel95a, SBTS01] In this paper, our purpose is to propose an election algorithm for the oriented hyper butterfly networks with $\mathcal{O}(N \log N)$ messages.

1 Hyper Butterfly Graphs

1.1 Hypercube

A hypercube H_n, of order n, is defined to be regular symmetric graph $G = (V, E)$ where V is the set of 2^n vertices, each representing a distinct n-bit binary number and E is the set of symmetric edges such that two nodes are connected by an edge iff the Hamming distance between the two nodes is 1 i.e., the number of positions where the bits differ in the binary labels of the two nodes is 1. For example, in H_3, the node 010 is connected to three nodes 110, 000 and 011. It is known that the number of edges in H_n is $n \times 2^{n-1}$ and the diameter of H_n is given by $\mathcal{D}(H_n) = n$.

1.2 Butterfly Graph

A wrapped butterfly network, denoted by B_n, is defined [Lei92] as follows: a vertex is represented as $(z_{n-1} \cdots z_0, \ell)$, where $z_{n-1} \cdots z_0$ is a n-bit binary number and ℓ is an integer, $0 \leq \ell \leq n-1$.

The edges of B_n are defined by a set of four generators. Consider an arbitrary node $(z_{n-1} \cdots z_0, \ell)$ in B_n. We define $\alpha(\ell) = \ell + 1 (\mathrm{mod}\, n)$ and $\beta(\ell) = \ell - 1 (\mathrm{mod}\, n)$. The four edges node $(z_{n-1} \cdots z_0, \ell)$ has can be derived by the following four generators:

$$g(z_{n-1} \cdots z_0, \ell) = z_{n-1} \cdots z_0, \alpha(\ell)$$
$$g^{-1}(z_{n-1} \cdots z_0, \ell) = z_{n-1} \cdots z_0, \beta(\ell)$$
$$f(z_{n-1} \cdots z_0, \ell) = z_{n-1} \cdots z_{\ell+1} \bar{z}_\ell z_{\ell-1} \cdots z_0, \alpha(\ell)$$
$$f^{-1}(z_{n-1} \cdots z_0, \ell) = z_{n-1} \cdots z_{\beta(\ell)+1} \bar{z}_{\beta(\ell)} z_{\beta(\ell)-1} \cdots z_0, \beta(\ell)$$

Remark 1. We refer to the frist part in the butterfly label, i.e. $z_{m-1} \cdots z_0$ as **complementation index**; and the second part, i.e. ℓ, as **permutation index**.

H. Jin et al. (Eds.): NPC 2004, LNCS 3222, pp. 292–299, 2004.
© IFIP International Federation for Information Processing 2004

1.3 Hyper-Butterfly Graph $HB_{(m,n)}$

Consider two undirected graphs $G = (V_G, E_G)$ and $H = (V_H, E_H)$; the product graph $G \times H$ has node set $V_G \times V_H$. Let u and v be any two nodes in G, and let x and y be any two nodes in H; then, $(\langle u, x\rangle, \langle v, y\rangle)$ is an edge of $G \times H$ iff either (1) (u, v) is an edge of G and $x = y$, or (2) (x, y) is an edge of H and $u = v$.

Definition 1. *A* **Hyper-Butterfly** *graph $HB_{(m,n)}$, of order (dimension) $(m + n)$ is defined as the product graph of a hypercube H_m of dimension m and a butterfly B_n of dimension n.*

In $HB_{(m,n)}$, each node is assigned a label $(x_{m-1} \dots x_0, z_{n-1} \cdots z_0, \ell)$ where each x_i and z_j are binary bit, $0 \le i \le m-1$ and $0 \le j \le n-1$. ℓ is an integer, $0 \le \ell \le n-1$. $x_{m-1} \cdots x_0$ is the *hypercube-part-label* and $(z_{n-1} \cdots z_0, \ell)$ is *butterfly-part-label*. The edges of the $HB_{(m,n)}$ graph are defined by the following $m + 4$ generators:

$$h_i(x_{m-1} \cdots x_0, z_{n-1} \cdots z_0, \ell) = (x_{m-1} \cdots x_{i+1} \bar{x}_i x_{i-1} \cdots x_0, z_{n-1} \cdots z_0, \ell) \quad \forall i,\ 0 \le i \le m-1$$

$$g(x_{m-1} \cdots x_0, z_{n-1} \cdots z_0, \ell) = (x_{m-1} \cdots x_0, z_{n-1} \cdots z_0, \alpha(\ell)) \quad \alpha(\ell) = \ell + 1 (\mathrm{mod} n)$$

$$g^{-1}(x_{m-1} \cdots x_0, z_{n-1} \cdots z_0, \ell) = (x_{m-1} \cdots x_0, z_{n-1} \cdots z_0, \beta(\ell)) \quad \beta(\ell) = \ell - 1 (\mathrm{mod} n)$$

$$f(x_{m-1} \cdots x_0, z_{n-1} \cdots z_0, \ell) = (x_{m-1} \cdots x_0, z_{n-1} \cdots z_{\alpha(\ell)+1} \bar{z}_{\alpha(\ell)} z_{\alpha(\ell)-1} \cdots z_0, \alpha(\ell))$$

$$f^{-1}(x_{m-1} \cdots x_0, z_{n-1} \cdots z_0, \ell) = (x_{m-1} \cdots x_0, z_{n-1} \cdots z_{\ell+1} \bar{z}_\ell z_{\ell-1} \cdots z_0, \beta(\ell))$$

Remark 2.

- The set of $m+4$ generators of the graph $HB_{(m,n)}$, $\Omega = \{h_i, 0 \le i < m, f, g, f^{-1}, g^{-1}\}$ is closed under inverse; in particular h_i for all i is its own inverse, g is inverse of g^{-1} and f is inverse of f^{-1}; thus the edges in $HB_{(m,n)}$ are bidirectional.
- For an arbitrary n, $n > 2$, for any arbitrary node v of the graph $HB_{(m,n)}$, $\delta(v) \ne v$ where $\delta \in \Omega = \{h_i, 0 \le i < m, f, g, f^{-1}, g^{-1}\}$; also, for any two $\delta_1, \delta_2 \in \Omega$, $\delta_1(v) \ne \delta_2(v)$.
- Hyper butterfly graph $HB_{(m,n)}$ is a Cayley graph of degree $m + 4$.
- For any m and n, $n \ge 3$, the graph $HB_{(m,n)}$ (1) is a symmetric (undirected) regular graph of degree $m+4$; (2) has $n \times 2^{m+n}$ vertices; and (3) has $(m+4) \times n \times 2^{m+n-1}$ edges.

Definition 2.

- *The m edges generated by the generators h_i are called* **hypercube edges** *and the 4 edges generated by either of the generators g, f, g^{-1}, f^{-1} are called* **butterfly edges**.
- *Any arbitrary node $v = (h, b) \in HB_{(m,n)}$ has m* **hypercube neighbors** $\{(h^{(i)}, b),\ 1 \le i \le m\}$ *(reached from v by the m hypercube edges) and has 4* **butterfly neighbors** $\{(h, b^{(j)}),\ 1 \le j \le 4\}$ *(reached from v by the 4 butterfly edges).*

Remark 3. Along any hypercube edge, only the hypercube-part-label of a node changes, and along any butterfly edges, only the butterfly-part-label changes.

Remark 4. The labeling of a hyper-butterfly graph is not unique. There exist many possible different label assignments with the same graph using traditional labeling scheme. We arbitrarily choose one such traditional labeling and refer to it as canonical labeling and will refer to the nodes using its canonical label.

Definition 3.

- *We use* $H_m^{(*,z,\ell)}$ *to denote an* m*-dimensional hypercube subgraph of* $HB_{(m,n)}$ *where each node has the same butterfly-part-label* (z, ℓ).
- *We use* $B_n^{(h,*,*)}$ *to denote an* n*-dimensional butterfly subgraph of* $HB_{(m,n)}$ *where each node has the same hypercube-part-label* h.
- *We use* $R_n^{(h,z,*)}$ *to denote a ring of* n *nodes where each node has the same hypercube-part-label* h *and same complementation index* $z = z_0 \cdots z_n$.
- *We use* $HR_{m,n}^{(*,z,*)}$ *to denote the set of nodes that have the same complementation index* z*. This set of node is actually the product of a* $H_m^{(*,z,\ell)}$ *and* $R_n^{(h,z,*)}$*, with the same* z *value.*

2 Leader Election Algorithm in Hyper-Butterfly Graph

Consider a hyper-butterfly graph $HB_{(m,n)}$; a *leader* node is defined to be any node of the graph unambiguously identified by some characteristics (unique from all other nodes). A leader election process is defined to be an uniform algorithm executed at each node of the network; at the end of the algorithm execution, exactly one node is elected the *leader* and all other nodes are in the non-leader state.

Remark 5. If each node knows its canonical label, this election process is trivial. Consider the node having the smallest label, i.e. $(0 \cdots 0, 0 \cdots 0, 0)$ in $HB_{(m,n)}$; we can say that the node with this label will automatically become the leader and all other nodes are non-leaders.

In this paper, as in [Tel95b], we assume that the nodes in the graph do not know their canonical labels. We will still refer to the nodes by some canonical labels for convenience, but these labels or names have no topological significance [Tel95b]. We also assume in this paper that the network is oriented in the sense that each node can differentiate the links incident to it by different generators. (in contrast, a node in an un-oriented star graph distinguishes its adjacent links by different but uninterpreted names). We define the *direction* of a link as the index of the generator that generates the link. So, the link that is associated with generator h_i has direction i, $0 \leq i \leq n-1$. And the link that is associated with generator g, g^{-1}, f, f^{-1} has direction g, g^{-1}, f, f^{-1} respectively.

The whole elction algorithm in hyper-butterfly graph consists of three major steps. At different step, the graph is divided into different regions, and the leader for each

region is elected. At first step, the hyper-butterfly graph $H_{(m,n)}$ is divided into $n \times 2^n$ hypercubes. Within each hypercube, the nodes run a formerly proposed election algorithm for hypercubes. After this step, each hypercube will have a leader node. In the second step, the nodes with the same complementation index are considered in one region. For a certain complementation index z, the region is actually a ring of hypercube, i.e. $HR_{(m,n)}^{(*,z,*)}$. The hypercube leaders elected in the first step will compete with each other and elect one leader in $HR_{(m,n)}^{(*,z,*)}$ for each different z value. The third step is the final step, where the leaders elected in the second step compete with each other and elect one final leader for $HB_{(m,n)}$. In the following sections, we discuss the detail of election algorithm at each step.

Remark 6. It should be noted that in an oriented hyper-butterfly graph, each node can identify the region with the knowledge of direction of edges, i.e. one node can send a message to some other node in the same region by using a sequence of certain directions. the node does not have to know its canonical label in order to identify the region.

2.1 Election Algorithm in $H_m^{(*,z,\ell)}$

In $HB_{(m,n)}$, there are $n \times 2^n$ different butterfly labels, which we denote as (z, ℓ), $0 \leq z < 2^n$, $0 \leq \ell < n$; and the nodes with the same butterfly part label form a hypercube of dimension m, which we denote as $H_m^{(*,z,\ell)}$.

In this step, each nodes first runs a leader elction algorithm for hypercube []. The algorithm uses only *hypercube edges*, i.e. edges with direction 0 to $n-1$. At the end of this procedure, there will be one leader elected in $H_m^{(*,z,\ell)}$ for each different (z, ℓ) value pair. The details of the algorithm is listed as follows:

After the leader is set at $H_m^{(*,z,\ell)}$. The leader will broadcast its id to all nodes in $H_m^{(*,z,\ell)}$. Each node receives this broadcast message will save the leader's id into a variable HL (abbreviation for hypercube leader).

Lemma 1. *This step requires less than* $7.24 \times n \times 2^{m+n}$ *messages [Tel95b].*

2.2 Election in $HR_{(m,n)}^{(*,z,*)}$

After the first step, there will be a leader in each hypercube $H_m^{(*,z,\ell)}$, $(0 \leq z < 2^n,\ 0 \leq \ell < n)$. We use $(h(z,\ell), z, \ell)$ to denote the label of the leader in $H_m^{(*,z,\ell)}$, where $h(z,\ell)$ specifies the hypercube part label of the leader.

Remark 7. $h(z,\ell)$ does not denote a particular function to derive the hypercube part label from z and ℓ. $h(z,\ell)$ only indicates that the hypercube part label of the leader varies for different hypercubes. Since the leader in $H_m^{(*,z,\ell)}$ is determinate for any (z, ℓ) value pair, the hypercube part label of leader is also determinative and solely depends on the value of z and ℓ.

In the first step, the leader of each hypercube also broadcasts its id within the hypercube when it becomes the leader. After the broadcast procedure, every node in $H_m^{(*,z,\ell)}$ will be informed with the id of leader, i.e. $(h(z,\ell), z, \ell)$, and has variable HL set to it.

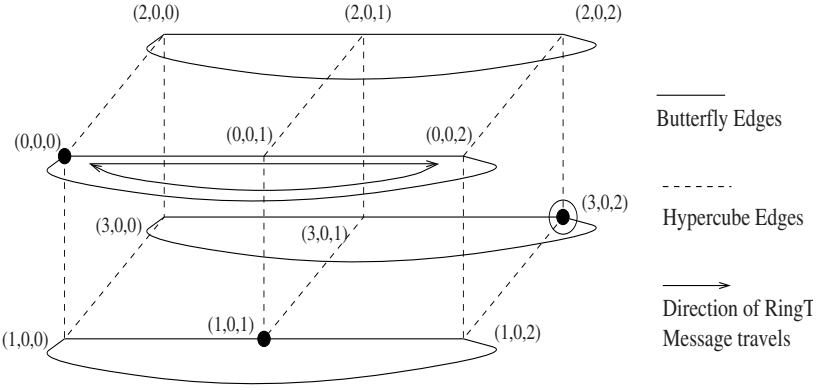

Fig. 1. Example of Leader Election in $HR^{(*,0,*)}_{(2,3)}$ (A subgraph of $HB_{(2,3)}$)

The objective in the second step is to elect leaders in larger regions. In this step, the nodes in hyper-butterfly graph $HB_{(m,n)}$ is considered to be grouped into 2^n new regions: $HR^{(*,z,*)}_{(m,n)}$, $0 \leq z < 2^n$, each consists of n hypercubes with the same complementation index, $H^{(*,z,\ell)}_m$, $0 \leq \ell < n$. Each hypercube has one leader elected from the first step, and there are totally $n \times 2^n$ such hypercube leader, with n in each new region $HR^{(*,z,*)}_{(m,n)}$. In the second step, each hypercube leader $(h(z,\ell),z,\ell)$ invokes procedure $HRElect$ to compete with other $n-1$ hypercube leaders in the same region $HR^{(*,z,*)}_{(m,n)}$. Only one of them becomes the new leader. After every hypercube leader finishes procedure $HRElect$, there will be only 2^n leaders left, with one in each $HR^{(*,z,*)}_{(m,n)}$, $0 \leq z < 2^n$. The *** code of procedure $HRElect$ is listed below.

Procedure $HRElect(h(z,\ell),z,\ell)$
Initial Conditions:

1. Node $(h(z,\ell),z,\ell)$ is the leader of $H^{(*,z,\ell)}_m$.
2. All nodes in $HB_{(m,n)}$ have viarable HL set to the label of the hypercube leader.

Invocation of the Procedure:

Node $(h(z,\ell),z,\ell)$ sends message $RingTest((h(z,\ell),z,\ell),0,True)$ along direction g.

Upon receiving message $RingTest(id,i,b)$ **from direction** g^{-1}**:**

//id is the label of the node which invokes procedure $HRElect$.
//i is an integer from 0 to $n-1$ and b is a boolean
//with the value of either $True$ or $False$.
if $(i < n-1)$
{
if $(b == False || HL > id)$
send message $RingTest(id,i+1,False)$ through direction g.

```
        else
            send message RingTest(id, i + 1, True) through direction g.
        }
    else
        {
        // This means the message gets back to the leader node (h(z, ℓ), z, ℓ)
        if (b == True)
            // This node passed all tests in the ring, becomes the new leader.
            Current node (h(z, ℓ), z, ℓ) becomes the leader of HR_(m,n)^(*,z,*).
        else
            // Failed the test, becomes non-leader.
            Current node becomes non-leader in HR_(m,n)^(*,z,*).
        }
```

As we can see, the procedure can be invoked from any hypercube leader $(h(z,\ell),z,\ell)$. It consists of sending message $RingTest(id,i,b)$ carrying three parameters. The first parameter is the id of the hypercube leader that invokes the procedure. The second parameter i is an integer that counts the number of nodes message $RingTest(id,i,b)$ has passed except the origin node. b is a boolean to indicate if id is large enough to be the leader of the part of $HR^{(*,z,*)}_{(m,n)}$ that the message has passed so far.

Since every node increments i and relays the message through direction g, message $RingTest(id,i,b)$ will go through a ring of n nodes and get back to the origin node $(h(z,\ell),z,\ell)$ after that. At each intermediate node, the variable HL is compared to id that comes from the message. If HL is larger than id, then b is set to $False$ to indicate that the node invoke the procedure is not large enough to be the leader of $HR^{(*,z,*)}_{(m,n)}$. When message $RingTest(id,i,b)$ gets back to the origin node that invokes the procedure, the node checks the value of b and becomes the leader of $HR^{(*,z,*)}_{(m,n)}$ or a non-leader accordingly.

Lemma 2. *For any arbitrary value of z, $0 \le z < 2^n$, after all n hypercube leaders in $HR^{(*,z,*)}_{(m,n)}$ execute procedure $HBElect$ and get back the message $RingTest$, only one of them will become the leader of $HR^{(*,z,*)}_{(m,n)}$ and all others will become non-leader.*

Proof. For any arbitrary value z, there are n hypercube leaders in $HR^{(*,z,*)}_{(m,n)}$. They are $(h(z,\ell),z,\ell)$, $0 \le \ell < n$. Each of them invokes procedure $HRElect$ and will determine to becomes a leader or non-leader depending on the value of b returned from message $RingTest$.

Consider an arbitrary leader $(h(z,\ell),z,\ell)$ among them. This node starts procedure $HRElect$ by sending message *RingTest ((h(z,ℓ),z,ℓ),0,True)* through direction g. Because every node gets the message also relays it through direction g, the message will traversal every node in $R_n^{(h(z,\ell),z,*)}$ which include n nodes: $(h(z,\ell),z,j)$, $0 \le j < n$. And because node $(h(z,\ell),z,j)$ has variable HL set to the id of leader of hypercube

$H_m^{(*,z,j)}$, i.e. $(h(z,j),z,j)$. The id $(h(z,\ell),z,\ell)$ is compared with the id of leaders of other hypercubes $H_m^{(*,z,j)}$, i.e. $(h(z,j),z,j)$, $0 \leq j < n$. If some node becomes the leader after executing procedure $HRElect$, it is assured that its id is larger than all other leaders in $H_m^{(*,z,j)}$, $0 \leq j < n$. Therefore, from n hypercube leaders $(h(z,\ell),z,\ell)$, $0 \leq \ell < n$, only one of them can claim as the leader of $HR_{(m,n)}^{(*,z,*)}$ after executing procedure $HRElect$; all other $n-1$ nodes will become non-leaders.

Remark 8.

- After every hypercube leaders (elected from first step) complete procedure $HRElect$, there will be 2^n nodes remain as leader, with each from $HR_{(m,n)}^{(*,z,*)}$, $0 \leq z < 2^n$.
- After a node becomes the leader in $HR_{(m,n)}^{(*,z,*)}$, it broadcast its id to all nodes in $HR_{(m,n)}^{(*,z,*)}$. And the node receives the broadcast message will save the leader's id into variable HRL.

Lemma 3. *There will be $n \times 2^{m+n} + n^2 \times 2^n$ number of messages generated in the second step, including procedure $HRElect$ and the broadcast process afterwards.*

Proof. There are $n \times 2^n$ hypercube leaders elected from the first step. Each leader executing procedure $HRElect$ generates n messages. And there will be 2^n leaders elected afterwards. Each leader in $HR_{(m,n)}^{(*,z,*)}$, $0 \leq z < 2^n$ will broadcast it id, which takes $n \times 2^m$ message. So the total number of message needed in this step is $n \times 2^{m+n} + n^2 \times 2^n$.

2.3 Leader Election in $HB_{(m,n)}$

After the second step, there will be one leader left in each $HR_{(m,n)}^{(*,z,*)}$, $0 \leq z < 2^n$. We use $(h(z),z,\ell(z))$ to denote the label of the leader in $HR_{(m,n)}^{(*,z,*)}$.

Remark 9. Similar to the second step, $h(z)$ or $\ell(z)$ does not specify the function to derive the hypercube part label or permutation index of the leader node. They only indicates the dependency relationship with between those labels with z.

In the third step, which is the final step, the objective is to elect one leader for entire hyper-butterfly graph $HB_{(m,n)}$. Since we already have 2^n leaders in each $HR_{(m,n)}^{(*,z,*)}$, we use similar approach as in the second step to elect one node from the those leaders to become the final leader.

In this step, each of the 2^n leaders from the second step sends $TreeTest$ message through a tree structure in butterfly graph. We ensure that the message will get to the nodes with different complementation index, so that the id of each leader will be tested to see if it is larger than the ids of all other leaders. As in procedure $HRElect$, only the node that passes all tests will become the leader which is the leader of entire

hyper-butterfly graph $HB_{(m,n)}$. The pseudocode listing of the details of the procedure $HBElect$ to be invoked by every leader of $HR_{(m,n)}^{(*,z,*)}$ is omitted for lack of space.

There are two types of messages used in the procedure. The first type of message is $TreeTest$ which travels down a binary tree because each node distributes the message through direction g and f. The parameter id and b have the same meaning as in the second step, while i indicates the current level of the tree. The other type of message is $TreeReply$ which go through the reversal path of $TreeTest$. Only the leaf nodes will compare the value of HRL with id that comes from the message. The intermediate nodes only act as transmit message $TreeTest$ to both children and collect $TreeReply$ from them. We state the following lemmas (the proofs are omitted for lack of space).

Lemma 4. *Consider the execution of procedure $HBElect$ from any node $(h(z), z, \ell(z))$, there are 2^i nodes that receive both message $TreeTest(id, i, b)$ and $TreeReply(id, i, b)$. These 2^i nodes have the same permutation index and hypercube label, but different complementation index.*

Lemma 5. *After every leader from the second step completely execute procedure $HBElect$, only one node will become leader of $HB_{(m,n)}$. All other leaders will become non-leader.*

Lemma 6. *There are 2^{2n+2} number of messages generated in the third step.*

Theorem 1. *The total number of message needed for a leader election algorithm in $HB_{(m,n)}$ is $8.24 \times n \times 2^{m+n} + (n^2 + 2^{n+2}) \times 2^n$.*

3 Acknowledgement

The work of Pradip Srimani was partially supported by a National Science Foundation award # ANI-0218495.

References

[GHS83] R. G. Gallagar, P. A. Humblet, and P. M. Spira. A distributed algorithm for minimum weight spanning trees. *ACM Transactions on Programming Languages and Systems*, 5:67–77, 1983.

[Lei92] F. T. Leighton. *Introductions to Parallel Algorithms and Architectures: Arrays, Trees and Hypercubes*. Morgan Kaufman, 1992.

[LMW86] M. C. Loui, T. A. Matsuhita, and D. B. West. Election in a complete network with a sense of direction. *Information Processing Letters*, 22:185–187, 1986.

[SBTS01] W. Shi, A. Bouabdallah, D. Talia, and P. K. Srimani. Leader election in wrapped butterfly networks. In *Proceedings of Parallel Computing 2001*, pages 382–389, Naples, Italy, September 2001.

[Tel93] G. Tel. Linear election in oriented hypercubes. Technical Report RUU-CS-93-39, Computer Science, Utrecht University, 1993.

[Tel95a] G. Tel. Linear election in hypercubes. *Parallel Processing Letters*, 5:357–366, 1995.

[Tel95b] G. Tel. Linear election in hypercubes. *Information Processing Letters*, 5:357–366, 1995.