Circular Shift operation on Mesh and Hypercube

Ms. Khyati Gohil[#1]

[#]*Computer Science and Engineering Department, Atmiya Institute of Tech. & Science*
Yogidham Gurukul, Kalawad Road, Rajkot
[1]hnskhyati@gmail.com

***Abstract*— this paper gives basic idea about how basic communication operation in parallel system containing more than one processing elements connected via some topology performed. In this paper circular shift operation on mesh and hypercube topology is discussed. An algorithm is developed to perform shift operation on these two topologies.**

***Keywords*— Mesh, Hypercube, cut-through routing, dual.**

I. Introduction

In parallel processing, multiple processes that are executing the task simultaneously need to exchange data with each other due to some dependencies in most cases.

These data exchange operations are called communication operations. Due to interaction delays in data exchange operations, an efficiency of programs may affect.

Many interactions in parallel programs may involve more than two processes. Some common basic patterns of inter-process communication are frequently used by variety of parallel algorithms.

By proper implementation of these basic communication operations in different parallel architectures (discussed in chapter 1) the performance and quality can be improved and development effort and cost can be reduced.

As the large scale parallel computers are based on linear array or ring topology, in this chapter, various algorithms are implemented for some commonly used communication operations on simple interconnection networks, such as the linear array, two-dimensional mesh, and the hypercube.

As in interconnection network transfer of m words of data between any two nodes can be done in cost of ***ts + m tw***, if the link between source and destination nodes is free.

It is assumed that network is bidirectional with single port in which communication is done with point-point. We also assume that all the communication networks use the ***cut-through routing*** to traverse the network.

Some of operations have ***duals***, which is the opposite operation of original operation, can be performed by reversing the direction and sequence of communication in original operation.

II. Circular shift

Circular Shift operation is the shift operation (means data of left node passes to the right node for each node) in circular manner (means after traversing each node data again passes to the first node after reaching last node in the network).

Each node passes its data simultaneously.

In circular ***q-shift*** operation data of node ***i*** reaches to ***"(i+q) mod p"*** node in the network after completion of q-shift, so it also called ***Rotation***.

Here Circular Shift on mesh and hypercube topology with algorithm is discussed.

III. Circular Shift on Mesh

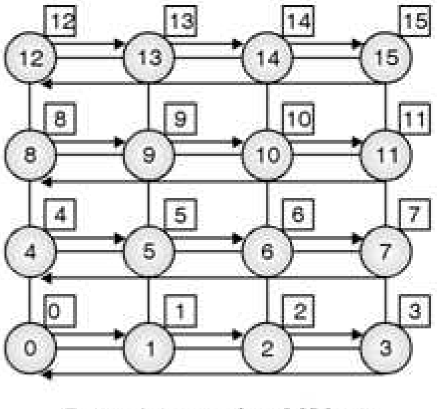
A circular ***q-shift*** can be performed on a ***p*** node square wraparound mesh in two stages. This is illustrated in figure for a circular 5-shift on a 4 x 4 mesh.

First, the each node shifts its data simultaneously in ***row-wise by total (q mod √p) shifts***. Then each node shifts its data in ***column-wise by total (q /√p) shifts***.

But during the row wise shift, data from highest label node of row moves to the lowest label node of that row because of the wraparound connection from last node of row to first node of that row. So rather than shifting data to forward it moves data in backward which requires an additional step (compensatory shift) along the columns to balance for the distance that they lost while traversing the backward.

During shifting of data if data moves backward by 2 columns then compensatory shift requires along 2 columns only and data on nodes of rest of columns are not affected.

For example, the ***5-shift*** requires one row shift, a compensatory column shift, and finally one column shift as shown in Fig. 1.



Step : 1 (row-wise shifting)

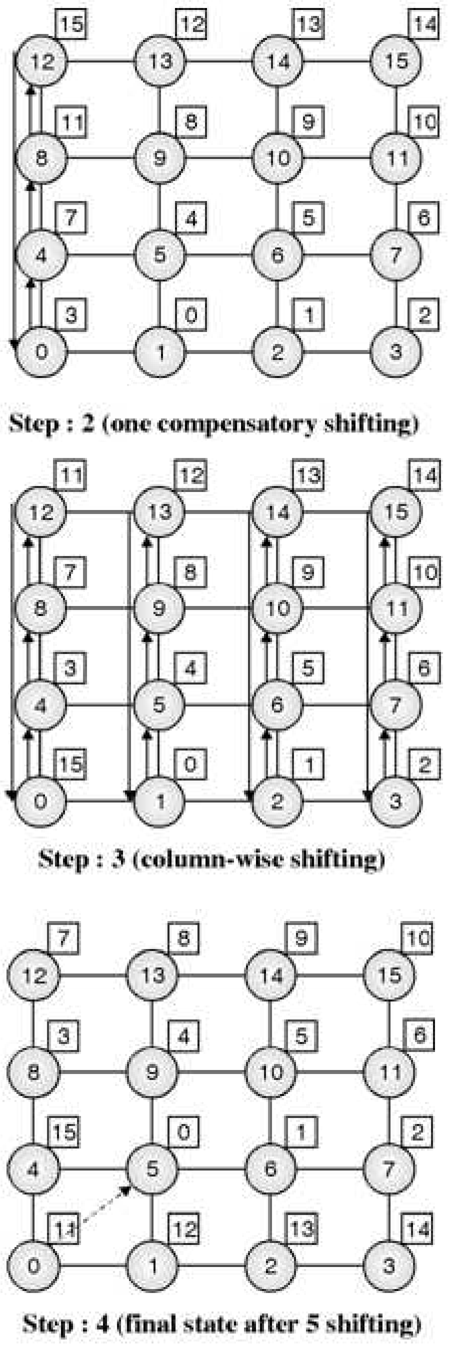

Step : 2 (one compensatory shifting)
Step : 3 (column-wise shifting)
Step : 4 (final state after 5 shifting)

Fig. 1 5-Shift operation on 4×4 Mesh

In step 1 shown in above figure data is shifting row-wise simultaneously in circular manner. Data of first row is shifted such that node 0 sends to node1, node 1 sends to node 2, node 2 sends to node 3 and node 3 sends to node 0. This type of communication is done at each row of mesh.

After completion of step 1, data at higher node such that node 3, 7, 11, and 15 will shifted to lower ranked node of its row rather than shifted to higher ranked node.

In step 2 compensation of this is done by shifting those data to column wise. And then step 3 will perform column-wise shifting of data stored at each node of mesh.

Algorithm for circular shift operation on mesh topology is shown here:

Algorithm for Circular shift on Mesh :

```
procedure CIRCULAR_SHIFT_MESH(my_id, my_msg,
                                        p, q)
begin
  left := my_id – (my_id mod √p) + (my_id – 1) mod √p;
  right := my_id – (my_id mod √p) + (my_id + 1) mod √p;
  up:= (my_id – √p) mod p;
  down:= (my_id + √p) mod p;
  count:=0;
  msg := my_msg;

/*row-wise communication*/
for i:=1 to (q mod √p) do
        send msg to right;
        receive msg from left;
        count:=count+1;
end for;

/* compensatory shift*/
if(my_id mod √p) < count then
        send msg to up;
        receive msg from down;
end if;

/*column-wise communication*/
for j:=1 to (q/√p) do
        send msg to up;
        receive msg from down;
end for;

end CIRCULAR_SHIFT_MESH;
```

An algorithm uses four arguments : my_id, my_msg, p and q. “my_id” is the rank or index or id of processes/nodes of mesh network. “my_msg” is the message stored at each node to be shifted during operation. “p” indicates total number of processing elements in mesh network, and “q” indicates how many shifts to be performed on mesh.

Algorithm is executed by all the processing elements presented in mesh network. Each node first computes the address / id of all of its neighbours : left, right, up and down. During row-wise communication algorithm sends data stored at each node of mesh network to its left neighbour and receives data from its right neighbour.

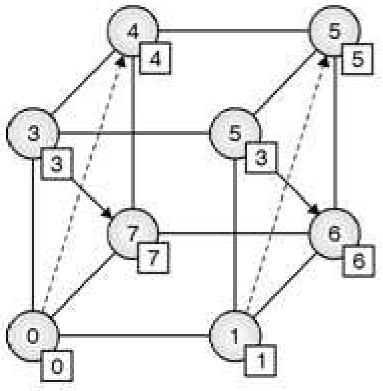
During compensatory shift (if performed) node sends data to up neighbour and receives from down neighbour. In last column-wise shifting process sends data to up and receives data from down node connected in mesh network.

Compensatory shift is performed or not is depends on how many steps node with higher ranked process sends its data to lower nodes of its row. It is used to compensate backward traversing of data performed by higher ranked node.

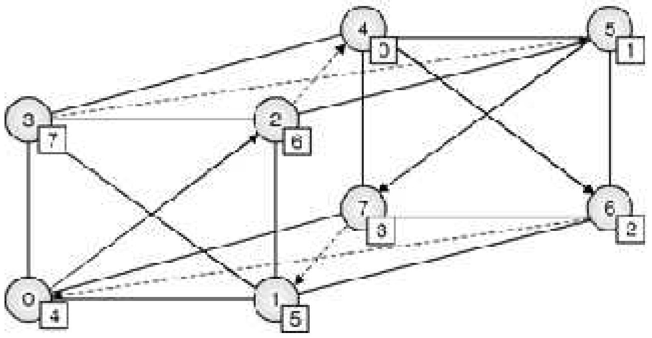
This algorithm reduces total steps need to performed.

IV. Circular shift on hypercube

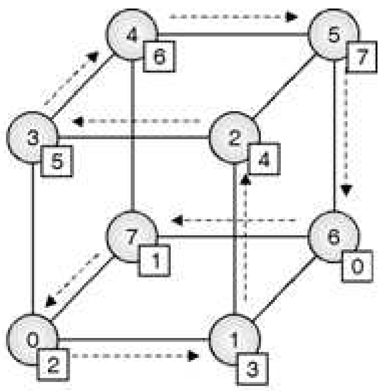
To perform circular shift operation on hypercube, map a linear array with 2^d nodes onto a d-dimensional hypercube by using d-bit binary reflected gray code (RGC).



2^2 shift



2^1 shift



2^0 shift

Fig. 2 7-Shift operation on 3-D Hypercube

To perform q-shift, expand q as sum of distinct powers of 2. For example, if q=7, then $\mathbf{q = 2^2 + 2^1 + 2^0}$.

These three terms correspond to bit positions at 0, 1 and 2 for value ‘1’ in the binary representation of 7, which is 111.

If q is the sum of ***s*** distinct powers of 2, then circular q shift on hypercube is performed in s phases. In our case for shift = q = 7, there are 3-phases are performed on hypercube starting from highest power: one for ***2^2=4 shift***, second for ***2^1=2 shift*** and third for ***2^0=1 shift***.

In each phase, all data packets move closer to their respective destination by leaps of powers of 2. For example, if q=7, then in first phase the data shifts by 2^2 i.e. 4 positions in linear array and so on.

An algorithm for circular shift operation on hypercube is shown below:

Algorithm for Circular shift on Hypercube :

```
procedure CIRCULAR_SHIFT_HCube(my_id, my_msg,
                                                d, q)
begin
  msg := my_msg;

  for i:=d-1 down to 0 do
        if (2^i AND q = 2^i) then
            partner := (my_id + 2^i) mod 2^d;
            send msg to partner;
        end if;
   end for;

end CIRCULAR_SHIFT_HCube;
```

In above algorithm my_id, my_msg, d and q are the arguments taken by it. “my_id” indicates the rank/index of processing elements of hypercube. “my_msg” is the message that each process holds. “d” is the dimension of hypercube and “q” is the number of shift to be performed by algorithm.

Loop executes 3 times for 3 dimension hypercube. During each iteration of loop each process of hypercube founds the partner node to send and receive data from.

V. Conclusion

The algorithm presented here assumes that each node of communication network has bidirectional channels on which data can be sent and received. Both algorithm improves throughput by splitting and routing message to be sent in to smaller parts rather than sending whole message to one intermediate node in the path. Apart from this rather than using single port for communication at each node, if node contains multiple communication ports with multiple links connecting nodes in the network speed of communication can be improved.

References

- “Parallel Processing” by Khyati Gohil, Techmax Publication (ISBN : 978-93-5077-189-1)