International Journal of Grid and Distributed Computing
Vol. 2, No. 4, December, 2009

One-to-One Embedding between Hyper Petersen and Petersen-Torus Networks

JungHyun Seo[1], MoonSuk Jang[1], EungKon Kim[1], KyeongJin Ban[1], NamHoon Ryu[1], and HyeongOk Lee[2]

[1]*Dept. of Computer Eng.,* [2]*Dept of Computer Edu., {National Univ. of Sunchon, Maegok 315, Sunchon, Jeinnam, South Korea}*

{jhseo, jang, kek, multiwave}@scnu.ac.kr, kmpi9560@hanmail.net, oklee@scnu.ac.kr

Abstract

Once a designed network is embedded into a newly designed network, a developed algorithm in the designed network is reusable in a newly designed network. Petersen-Torus has been designed recently, and Hyper Petersen has already been designed as a well-known interconnection network. In this study, it was proven that Hyper Petersen network whose degree increases with the increased number of nodes can be embedded into a Petersen-Torus network having fixed degree. Hyper Petersen $HP_{log_2 n^2+3}$ was embedded into Petersen-Torus PT(n,n) with expansion 1, dilation 1.5n+2, and congestion 5n. The embedding algorithm was designed for expansion 1, and congestion and dilation are in proportion to O(n) owing to the aspect of Hyper Petersen that the degree increases.

Keywords: *Parallel Processing, Interconnection Network, Petersen-Torus, Hyper Petersen, Embedding*

1. Introduction

In parallel processing, a shared memory model is called a multiprocessor system whose processor uses common memory. The local memory model is called a multicomputer system whose each processor has local memory. If a communication link between processors for data exchange is fixed, it is called a static network, while if it is not fixed, it is called a dynamic network. In general, a network where a communication link between processors is fixed is called interconnection network. Interconnection network consists of a set of processors, local memory, and a communication link between processors for data transfer. An interconnection network for a multicomputer system is divided into a static network and a dynamic network. A static network is divided into the tree class, the mesh class, the hypercube class and the star graph class according to the composition of nodes and edges to compose the network. An interconnection network can be modeled in a graph $G=(V,E)$. Each processor P_i is an element of a node set V, and two processors P_i and P_j are connected by a communication links (P_i, P_j). If an interconnection network is modeled in a graph, a processor is mapped into a node and a communication link into an edge. The number of nodes adjacent to node P_i is defined as degree of the node.

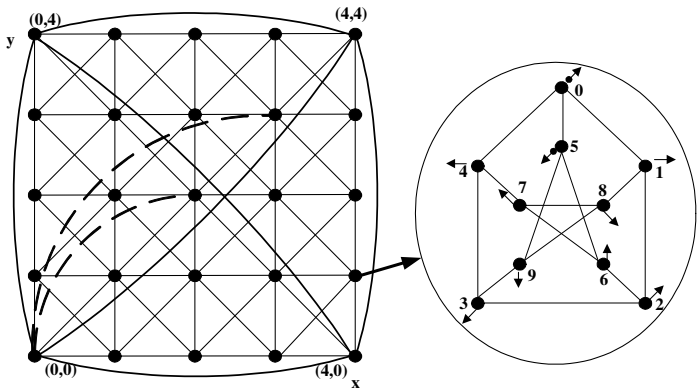
Given a guest graph G and a host graph H, embedding of G into H is described by an ordered pair (Φ, Ψ), where Φ maps each node of G to a node of H and Ψ maps each edge (u, v) of G to a path of H from nodes $\Phi(u)$ to $\Phi(v)$(hereafter referred to as Ψ-path). *Dilation* of the edge (u, v) is length of the Ψ-path in H, and *dilation* of the embedding (Φ, Ψ) is the largest value among dilations for all edges of G. *Congestion* of the edge of H is Ψ-paths traversing an edge of H, and *congestion* of the embedding (Φ, Ψ) is the largest

value among congestions for all edges of *H*. *Expansion* of the embedding (Φ, Ψ) is a ratio of the number of nodes of G to that of H. The measures to evaluate the embedding algorithm are *dilation*, *congestion* and *expansion*. The closer the values are to 1, the better the embedding algorithm is[1].

The *d*-dimensional Hyper Petersen having $10\times2^{d-3}$ nodes in various algorithms for parallel processing has a simple solution that it has time complexity O(*d*). In terms of network scalability, since the mesh class adds and expands a few processors by linking the adjacent processor and the hypercube class adds the exactly same number of nodes and links each processor, it is readily scalable. Therefore it is meaningful that the hypercube class having a good parallel algorithm is embedded in the mesh class which can be commonly used by virtue of its high scalability[2,3]. As Torus has degree 4 and Hypercube has more degree with increased dimensions, when Hypercube(in excess of degree 4) above 5 dimensions is embedded into Torus, congestion and dilation gradually increase in excess of 1[3]. In study [4], 6-dimensional Hypercube was embedded into 8×8 mesh at dilation 11, congestion 2, and expansion 1. In study [2], 11-dimensional Hypercube was embedded into 32×64 mesh at dilation 32 and congestion 57, and smaller Hypercube and meshes were embedded. In these two studies, Hypercube of the limited size was embedded in mesh, and the embedding scale was constant. In study [3], the result was generalized without limitation to previous studies. The *d*-dimensional Hypercube was embedded into mesh having the same number of nodes at dilation 2^d-2 and expansion 1.

The paper is organized as follows. Section 2 introduces the Petersen-Torus network and the Hyper Petersen network. Section 3 proposes an embedding algorithm of Hyper Petersen into Petersen-Torus network, and finally conclusion is given.

2. Related Work



(a) *PT*(5,5) (b) Petersen Graph

Figure 1. Petersen-Torus *PT*(5,5)

The Petersen-Torus *PT(m,n)*(*m*,*n*≥2) sets a Petersen Graph(Figure 1(b)) as a basic module, arranges *m*(x axis)×*n*(y axis) modules on grid points, and connects them under

edge definition. The Petersen-Torus network $PT(m,n)=(V_{pt}, E_{pt})$. The node definition of Petersen-Torus $PT(m,n)$ is

$$V_{pt}=\{(x,y,p),\ 0\le x<m,\ 0\le y<n,\ 0\le p\le 9\}$$

The edge of $PT(m,n)$ is divided into internal edge and external edge. The edge connecting nodes belonging to the same module is called internal edge, and the edge of Petersen Graph is used as it is as internal edge. The edge connecting nodes in different modules is called external edge. Edges are defined in the following. The symbol '%' is remainder operator in the following equations. ① The longitudinal edge is $((x,y,6), (x,(y+1)\%n,9))$. ② The latitudinal edge is $((x,y,1), ((x+1)\ \%m,y,4))$. ③ The diagonal edge is $((x,y,2), ((x+1)\%m,(y+1)\%n,3))$. ④ The reverse diagonal edge is $((x,y,7), ((x-1+m)\%m,(y+1)\%n,8))$. ⑤ The diameter edge is $((x,y,0), ((x+\lfloor m/2\rfloor)\%m,(y+\lfloor n/2\rfloor)\%n,5))$. Edges are defined in[5].

The Hyper Petersen network is a Hypercube-like network and can be made by Cartesian Product of Petersen Graph and Hypercube. It is a regular graph where degree is same in all nodes. Also, it has high degree, high connectivity, and small diameter. j-dimension HP consists of $10\times2^{j-3}$ nodes and $5j\times2^{j-3}$ edges and is indicated in $HP_j=(V_{hp}, E_{hp})$. A set of peaks is composed of two tuples and defined as follows[6]:

$V_{hp}=\{[B(u),P(i)]|B(u)$ is j-3 binary bit string, $\{0\le P(i)\le 9\}$.

Edge is divided into two:

Petersen-edge : In $([B(u),P(i)], [B(u),P(j)])$, $P(i)$ and $P(j)$ are linked in a Petersen graph.

Hypercube-edge : In $([B(u), P(i)], [B(v), P(i)])$, $B(u)$ and $B(v)$ are 1 bit different exactly.

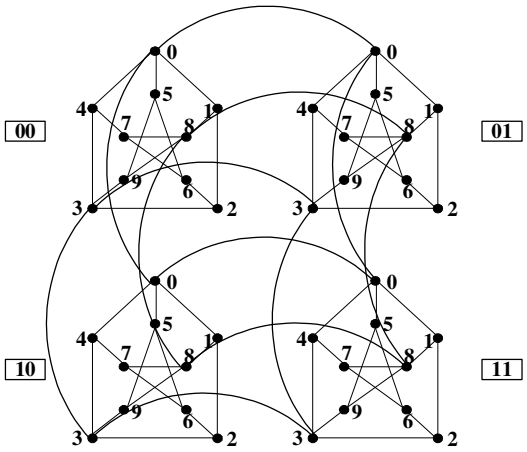


Figure 2. Hyper Petersen HP_5

j-dimension Hyper Petersen has 1.25×2^j nodes, degree j, and diameter j-1. The nature of Hyper Petersen is very similar to that of Hypercube, and the number of nodes for Hyper Petersen is more than 1.25 times of Hypercube on condition that it has similar diameter and degree[6].

HP_j is $Q_{j-3} \times P(j\ge3)$ and made by Cartesian product of Petersen Graph and j-3 dimensional Hypercube. That is, each node of Q_{j-3} Hypercube is replaced with a Petersen Graph. Figure 2 shows 5-dimensional Hyper Petersen. The node address is a combination of $B(u)$ consisting of binary bit strings and $P(i)$ consisting of one-digit numbers. Petersen-edge is all indicated and

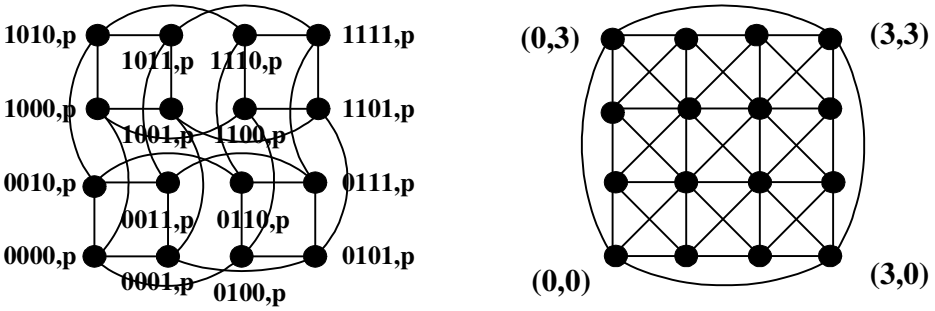
to avoid complexity of the figure, Hypercube-edge is indicated only if $P(i)$ is 0, 3, and 8. Even if $P(i)$ is a number except 0, 3, and 8, edge exists between nodes where $B(u)$ is one bit different and $P(i)$ is same, as in the case of 0, 3, and 8. Hyper Petersen dimension is the number of bit strings for $B(u)$ + 3.

3. Embedding Hyper Petersen into Petersen-Torus

The Basic embedding strategy is to map a Petersen Graph of HP into that of a PT basic module. Mapping the node address $B(u)$ of HP into the address of PT basic module (x,y). Ten nodes(Petersen Graph) having the same node address $B(u)$ among HP nodes are called a HP basic module.

Theorem 1 Hyper Petersen $HP_{log_2 n^2+3}$ is embedded into Petersen-Torus $PT(n,n)$ at expansion 1, dilation $1.5n+2$, and congestion $5n$ (n is a power of 2).

Proof The number of nodes for Petersen-Torus $PT(n,n)$ and that for Hyper Petersen $HP_{log_2 n^2+3}$ is $10n^2$. Therefore expansion is 1. In $HP_{log_2 n^2+3}$ node $[B(u),P(k)]$, the bit in the place of i for $B(u)$ is set as $B(u)_i$. In $B(u)$, a combination of bits in the place of odd numbers by order from LSB is $xb(u)=B(u)_{\log_2 n^2-1} \ldots B(u)_{i+2}B(u)_iB(u)_{i-2} \ldots B(u)_3B(u)_1$, and a combination of bits in the place of even numbers by order is $yb(u)=B(u)_{\log_2 n^2} \ldots B(u)_{i+3}B(u)_{i+1}B(u)_{i-1} \ldots B(u)_4B(u)_2$($i$ is an odd number). The node $[B(u),P(k)]$ of $HP_{log_2 n^2+3}$ is mapped into the node $(xb(u)_{(10)},yb(u)_{(10)},P(k))$ of $PT(n,n)$.



(a) HP_7 (b) $PT(4,4)$

Figure 3. HP_7 embedding into $PT(4,4)$

Figure 3 shows HP_7 and $PT(4,4)$. In the two graphs, one point is a basic module, and in mapping into a PT basic module from an HP basic module, the address of a Petersen Graph is mapped into the exactly same node. For example, xb of HP node [1001,5] is 01 and yb is 10, therefore it is mapped into the PT node (1,2,5).

HP edge is divided into Petersen-edge and Hypercube-edge. The edge ($[B(u),P(k)]$, $[B(v),P(k)]$) is Hypercube-edge having the same Petersen Graph address, while the edge ($[B(u),P(k)]$, $[B(u),P(o)]$) is Petersen-edge having the same Hypercube address. As Petersen-

edge ([$B(u)$,$P(k)$],[$B(u)$,$P(o)$]) is mapped into (($xb(u)_{(10)}$,$yb(u)_{(10)}$,$P(k)$) and ($xb(u)_{(10)}$,$yb(u)_{(10)}$,$P(o)$), dilation is 1 and congestion is 1.

In Hypercube-edge ([$B(u)$,$P(k)$], [$B(v)$,$P(k)$]) of HP, the HP node [$B(u)$,$P(k)$] is mapped into the PT node (($xb(u)_{(10)}$,$yb(u)_{(10)}$,$P(k)$) and the HP node [$B(v)$,$P(k)$] into the PT node($xb(v)_{(10)}$,$yb(v)_{(10)}$,$P(k)$)). $B(u)$ and $B(v)$ are exactly 1 bit different, and if the different bit is the bit in the place of an odd number from LSB, $xb(u)\neq xb(v)$ and $yb(u)=yb(v)$, while if the different bit is the bit in the place of an even number, $xb(u)=xb(v)$ and $yb(u)\neq yb(v)$. The bit in the place of t for $xb(u)$ is assumed as $xb(u)_t$, and the bit in the place of t for $yb(u)$ is assumed as $yb(u)_t(1\leq t\leq \log_2 n^2/2)$. If $xb(u)_t\neq xb(v)_t$, there is difference of 2^{t-1} between $xb(u)_{(10)}$ and $xb(v)_{(10)}$. When t is the largest value $\log_2(n/2)$, it is $n/2$, thus length of the external path for (($xb(u)_{(10)}$,$yb(u)_{(10)}$,$P(k)$), ($xb(v)_{(10)}$',$yb(v)_{(10)}$',$P(k)$)) is up to $n/2$ and that of the internal path is $(n/2+1)\times 2$. The sum of two path lengths is $1.5n+2$, and the value is dilation. The same is also applied to $yb(u)_t\neq yb(v)_t$.

In two PT nodes ($xb(u)_{(10)}$,$yb(u)_{(10)}$,$P(k)$) and ($xb(v)_{(10)}$,$yb(v)_{(10)}$,$P(k)$), two nodes u and v of the HP edge (u, v) are mapped. If $xb(u)_t\neq xb(v)_t$, two nodes are at a distance of n/2 in the x axis when y value is in the same basic module and at the farthest distance. Both the start basic module of the path and the ($n/2$)-1 middle passing basic module use the same external edge ((x,y,1), (x+1,y,4)). If $yb(u)_t\neq yb(v)_t$, an external edge ((x,y,6), (x,y+1,9)) is used. At this edge, congestion is $n/2\times 10$. Therefore embedding into dilation $1.5n+2$ and congestion $5n+2$ is possible. □

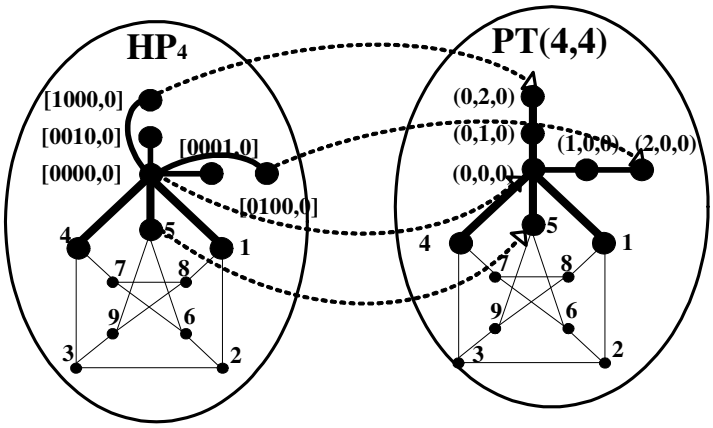


Figure 4. HP_7 embedded into $PT(4,4)$

For example, at the edges [0000,4] and [0100,4] of Figure 4, the node [0000,4] is mapped into (0,0,4) and the node [0100,4] into node (2,0,4). The path between node (0,0,4) and node(2,0,4) is as follows: (0,0,4), (0,0,0), (0,0,1), (1,0,4), (1,0,0), (1,0,1). (2,0,1), (2,0,0), (2,0,4). The path has two external edges and 3×2 internal edges and the length is 8.

Theorem 2 Hyper Petersen $HP_{log_2 2n^2+3}$ is embedded into Petersen-Torus $PT(2n,n)$ at expansion 1, dilation $3n+2$, and congestion $10n+2$ (n is a power of 2).

Proof The number of nodes for Petersen-Torus $PT(2n,n)$ and that for Hyper Petersen $HP_{log_2 2n^2+3}$ is $20n^2$. Therefore expansion is 1. Node mapping is as demonstrated in

Theorem 1. This embedding just indicates that HP_j is made by Cartesian product of a Q_{j-3} graph and a Petersen Graph on condition that j-3 is an odd number, and in a PT graph, the x axis is exactly twice greater than the y axis. This embedding is same as that of $PT(2n,2n)$ into $HP_{log_2 4n^2+3}$. Length of the external path for $((xb(u)_{(10)},yb(u)_{(10)},p)$ and $(xb(v)_{(10)}',yb(v)_{(10)}',p))$ is up to n. Length of the internal path is $(n+1)\times 2$ and maximum length of the path is $3n+2$, thus it is embedded at dilation $3n+2$. As 10 nodes of a Petersen Graph having the edge ($[B(u),P(k)]$, $[B(v),P(k)]$) use the same external path, congestion at external path is $10n$. Congestion at the last node of the path is 2, therefore it is embedded at congestion $10n+2$. Thus embedding at dilation $3n+2$ and congestion $10n+2$ is possible. □

4. Conclusion

Embedding between interconnection networks is meaningful in that the designed parallel algorithm is reusable. In this study, a Hyper Petersen network designed based on a Petersen Graph was embedded into a PT network. Hyper Petersen $HP_{log_2 n^2+3}$ was embedded into $PT(n,n)$ at expansion 1, dilation $1.5n+2$, and congestion $5n$. This result indicates that the algorithm designed in Hyper Petersen is effectively reusable in PT. Reversely, it is worth studying embedding of a PT network into other networks in order to use the algorithm developed in a PT network in another network.

References

[1] S. Bettayeb and B. Cong and M. Girou and I. H. Sudborough, Embedding Star Networks into Hypercubes, IEEE trans. comput., VOL. 45, No. 2, pp. 186-194, Feb. 1996.

[2] T. M Stricker, Supporting the hypercube programming model on mesh architectures, Proc. of the fourth annual ACM symposium on Parallel algorithms and architectures, pp. 148-157, 1992.

[3] A. Gonzalez, V. Garcia and L. D. Cerio, Excuting Algorithm with Hypercube Topology on Torus Multicomputers, IEEE Transactions on Parallel and Distributed Systems, Vol. 6, No. 8, pp. 803-814, Feb 1995.

[4] S. Matic, Emulation of Hypercube Architecture on Nearest-Neighbor Mesh-Connected Prcessing Elements, IEEE Transaction on computer, Vol. 39, pp. 698-700, May 1990.

[5] J. H. Seo, H. O. Lee and M. S. Jang, Petersen-Torus Networks for Multicomputer Systems, Proc. int'l Conf. of NCM2008, Vol. 1, pp. 567-571, Sep, 2008.

[6] S. K. Das and A. K. Banerjee, Hyper Petersen network: Yet another hypercube-like topology, In Proc. of the 4th Symp., on the Frontiers of Massively Parallel Computation, pp 270-277, McLean, Virginia, USA, Oct. 1992.

Authors

JungHyun Seo, e-mail: jhseo@scnu.ac.kr. He received the Ph.D. in department of computer science from Sunchon National University, Korea in 2008. He is a chief researcher who is

working for department of strategy industry rearing at Jeonnam TechnoPark, Korea. His research interests include parallel processing, interconnection network, algorithm.

Moonsuk Jang, E-mail : jang@scnu.ac.kr , He received the Ph.D. in department of computer science from Kwangwoon University, Korea in 1995. He is a professor who is working for department of computer science at Sunchon National University, Korea. His research interests include artificial intelligence, GIS, parallel computer.

EungKon Kim, E-mail : kek@scnu.ac.kr, He received the Ph.D. in department of computer engineering from Chosun University, korea in 1994. He is a professor who is working for department of computer science at Sunchon National University, Korea. His research interests include computer graphics, multimedia, HCI.

Kyeong-Jin Ban, e-mail: *multiwave@scnu.ac.kr*, He received the M.S. degrees in Computer Science in 2005 from Sunchon National University, Korea. He is now a Ph.D. Candidate in Sunchon National University. His research interests include computer graphics, RFID, USN.

NamHoon Ryu, e-mail: *kmpi9560@hanmail.net,* He received the M.S. degrees in Computer Science in 2009 from Sunchon National University, Korea. He is now a Ph.D. Candidate in Sunchon National University. His research interests include computer graphics, algorithm.

Hyeongok Lee, e-mail: oklee@scnu.ac.kr , He received the Ph.D. in department of computer science from Jeonnam National University, Korea in 1999. He is a professor who is working for department of computer education at Sunchon National University, Korea. His research interests include algorithm, interconnection network, parallel processing.