Routing and Embeddings in Cyclic Petersen Networks: An Efficient Extension of the Petersen Graph

Chi-Hsiang Yeh and Behrooz Parhami
Department of Electrical and Computer Engineering
University of California,
Santa Barbara, CA 93106-9560, USA

Abstract

The Petersen graph is a Moore graph that has node degree 3, diameter 2, and optimal network size 10. In this paper, we present a class of interconnection networks, called cyclic Petersen networks (CPNs), which efficiently extend the Petersen graph to obtain larger networks with small diameter and node degree. We derive balanced routing algorithms and efficient embeddings for CPNs. In particular, we show that many normal mesh algorithms can be emulated on CPNs with a slowdown factor of about 1.1. We also show that complete CPNs can embed meshes, tori, meshes of trees, and folded Petersen networks with dilation 3, hypercubes and generalized hypercubes with dilation 4, and pyramids with dilation 5.

1 Introduction

The Petersen graph is a Moore graph that has 10 nodes of degree 3 and a diameter of 2, is symmetric, and is the most efficient small network in terms of node degree, diameter, and network size (see Fig. 1). Due to its unique and optimal properties, several network topologies based on the Petersen graph have been proposed and investigated in the literature [3, 4, 5, 7, 8, 9, 10, 11, 16].

In this paper, we present a class of interconnection networks called *cyclic Petersen networks (CPNs)*, which efficiently extend the Petersen graph to obtain larger networks with diameter and node degree smaller than those of a similar-size hypercube. The diameters of a 1-level CPN, a 2-level CPN, and l-level ring-CPNs, $l \geq 3$, are optimal within factors of 1, 1.25, and 1.8, respectively, given their node degrees; the "degree $\times$ diameter" costs [1] of a 1-level CPN, a 2-level CPN, and l-level CPNs, $l \geq 3$ are optimal within factors of 1, 1.25, and 1.8, respectively, for networks of any degree. We present efficient algorithms for balanced routing, efficient emulations, and constant-dilation embeddings in CPNs. In particular, we show that many normal mesh algorithms can be emulated on CPNs with a slowdown factor of about 1.1. Moreover, complete-CPNs can embed meshes, tori, meshes of trees, and folded Petersen networks with dilation 3, hypercubes, and generalized hypercubes with dilation 4, and pyramids with dilation 5. They can also emulate an l-D mesh under the all-port communication model with a factor of $\max(4, l+1)$ slowdown.

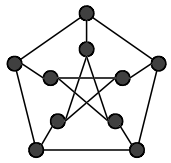

Figure 1. The Petersen graph.

The remainder of this paper is organized as follows. In Section 2, we present cyclic Petersen networks and derive several basic properties and algorithms. In Section 3, we present enhanced CPNs, and derive efficient embeddings and emulation algorithms for them. In Section 4, we present clustered CPNs, which have smaller step sizes. In Section 5, we conclude the paper.

2 Basic Cyclic Petersen Networks

In this section, we define basic CPNs (also called ring-CPNs), explore some of their topological properties, and introduce the needed notation.

2.1 Definition of Basic CPNs

For convenience, for any $j_1 \geq j_2$, we let $Z_{j_1:j_2}$ denote $Z_{j_1}Z_{j_1-1}\cdots Z_{j_2}$, where Z can be any symbol, such as U, V or X.

Definition 2.1 (Ring-CPN): Let $P = (V_P, E_P)$ be the Petersen graph. An l-level ring-cyclic Petersen network is defined as (V, E), where $V = \{V_{l:1} | V_i \in V_P, i = 1, \ldots, l\}$ is

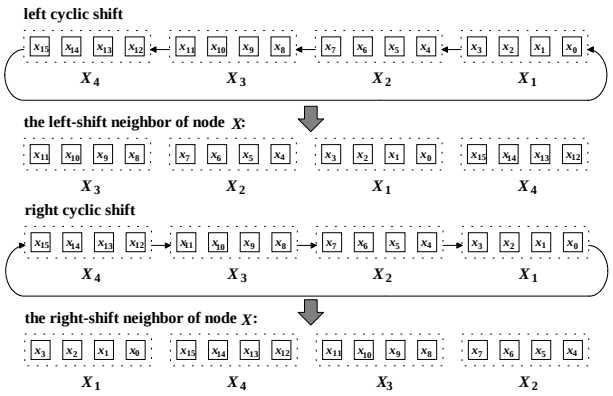

Figure 2. The derivation of shift links (neighbors) of a node $X = X_{4:1}$ in a 4-level ring-CPN. Symbols $X_i \in [0,9]$, $i = 1,2,3,4$, are represented by 4-bit binary numbers $x_{4i-1:4(i-1)}$.

the set of vertices, and $E = \{(U_{l:1}, V_{l:1}) |\ U_i, V_i \in V_P, i = 1,2,\ldots,l$, satisfying $U_{l:2} = V_{l:2}$ and $(U_1, V_1) \in E_P$, or $U_i = V_{(i \bmod l)+1}$, or $V_i = U_{(i \bmod l)+1}$, for $1 \le i \le l\}$ is the set of edges.

A 1-level CPN is the Petersen graph (see Fig. 1) and is called a *nucleus* of a CPN. Two nodes U and V are connected by an undirected link if and only if nodes U and V are neighbors within the same nucleus P, or the address of nodes U and V are cyclic shifts of the l-symbol addresses of one another (see Fig. 2). The former link is called a *nucleus link* and the latter is called a *left- (right-) shift link*. CPNs form a subclass of *cyclic-shift networks* (also called *cyclic networks*) [2, 17] that use the Petersen graph as the nucleus graph; cyclic-shift networks, in turn, form a subclass of *super-IP graphs* that use cyclic-shift operators as the *super-generators* [18, 20, 21].

Let $X^{(i)}$ be the address obtained from node X by performing i right cyclic shifts. That is, $X^{(0)} = X$ and $X^{(i)} = X_{i:1}X_{l:i+1}$ for $1 \le i < l$, where $X = X_{l:1}$. Note that $X^{(i)} = X^{(i \bmod l)}$. The addresses of *left- (right-) shift neighbors* of node X can be represented as $X^{(-1)}$ (or $X^{(1)}$, respectively). Let $X = X_{l:1}$ be a node in a ring-CPN, where $X_i \in V_P$ and $X \ne X^{(i)}$ for $i = 1,2,\ldots,l-1$. It can be seen that by the definition of ring-CPNs, nodes $X, X^{(1)}, X^{(2)}, \ldots, X^{(l-1)}$ form an l-node ring, connected through shift links. In general, the majority of rings formed by the shift links are of this type. However, when l is not a prime number, there will also be shorter rings with l_f nodes, where l_f divides l (see Fig. 3a). Since the addresses of shift links (neighbors) are obtained by performing cyclic shift on the address of a node, and these derived neighbors form a ring, we call such networks "ring-cyclic" Petersen networks. The rings with l nodes are called the *cyclic-shift (CS) graphs* of the ring-CPN; the rings with l_f nodes are called the *degenerate CS graphs* of the ring-CPN.

Note that a node with the same l symbols in its address has no shift links (or, alternatively, has shift links connecting to itself) and is called a *leader*. Leaders can be used as I/O ports or be connected to other leaders via their unused ports to provide better fault tolerance or to improve the performance and reduce the diameter of ring-CPNs without increasing the node degree of the network. Varying the connectivity between leaders results in other classes of ring-CPNs.

2.2 Routing and Topological Properties

The number of nodes in a ring-CPN is increased by a factor of 10 when the level is increased by 1, and the nucleus Petersen graph has 10 nodes. Thus, the number of nodes N of an l-level ring-CPN is

$$N = 10^l. \tag{1}$$

From Eq. 1, the level of an N-node ring-CPN is

$$l = \log_{10} N. \tag{2}$$

The node degree of an l-level ring-CPN is

$$d = \begin{cases} 3 \text{ when } l = 1, \\ 4 \text{ when } l = 2, \\ 5 \text{ when } l \ge 3. \end{cases} \tag{3}$$

Suppose that a routing algorithm for the nucleus P is known. Let the addresses of nodes X and Y within the l-level ring-CPN be $X_{l:1}$ and $Y_{l:1}$, respectively, where $X_i, Y_i \in V_G$. In what follows, we present a routing algorithm to route a packet from node X to node Y in an l-level ring-CPN using left- (right-) shift links and nucleus links.

Route(X,Y)

For $i = l$ downto 1 (or $i = 1$ to l)
Route the packet to node Y_i (or $Y_{(i \bmod l)+1}$)
within the nucleus in which the packet currently resides.
If $i \ne 1$ (or $i \ne l$), send the packet
through the left-shift link (or right-shift link).

If the routing algorithm on the nucleus P requires at most $T_R(P)$ time, the routing algorithm on an l-level ring-CPN requires time at most

$$T_R(l) = lT_R(P) + l - 1.$$

Since the diameter of the Petersen graph is 2, an optimal routing algorithm requires time at most $T_R(P) = 2$, leading to

$$T_R(l) = 3l - 1. \tag{4}$$

This leads to the diameter of an l-level ring-CPN, which is

$$3l-1=\frac{3}{\log_2 10}\log_2 N-1\approx 0.9\log_2 N-1.$$

The expected traffic on the network links of a ring-CPN is approximately balanced when the sources and destinations of the packets are uniformly distributed over network nodes. This can be intuitively justified by observing that the average time for an intra-nucleus routing phase is 1.5, the routing algorithm uses roughly the same number of intra-nucleus and shift routing phases, and each node has 1.5 times more nucleus links than shift links (3 versus 2). When $l=2$ and the external arrival rate is λ, the average traffic on a network link is 0.972λ, while the traffic on a shift link is λ and the traffic on any nucleus link is smaller than λ. Therefore, the expected traffic on any network link is no more than 2.9% above the average traffic on all network links. For any $l\geq 3$ the expected traffic on a network link in an l-level ring-CPN exceeds the average traffic on all network links by no more than 6%.

2.3 Emulation of Normal Mesh Algorithms

If a mesh algorithm executes t operations (or routing steps) on the average along a dimension before executing an operation along the next consecutive dimension (cyclicly), we call it a *normal mesh algorithm with (an average of) t row operations*. Note that a node can send data to both its west and east neighbors along the same dimension at the same time. Many mesh algorithms naturally fall into this category. Many other algorithms can be easily transformed to a normal mesh algorithm without affecting the leading constant of the running time.

In what follows, we show that normal mesh algorithms can be emulated on CPNs efficiently.

Theorem 2.1 *A normal mesh algorithm for l-dimensional* $10\times 10\times\cdots\times 10$ *meshes with an average of t row operations can be emulated on an l-level ring-CPN with a slowdown factor of* $1+\frac{1}{t}$.

Proof: The Petersen graph contains a 10-node linear array so transmissions along dimension-1 mesh links can be performed directly. To emulate operations along dimension a, a node X in the guest mesh is mapped onto node $X^{(a-1)}$ of the host CPN. Since operations are performed along consecutive dimensions when the dimension is changed, only one transmission along cyclic-shift links is required for the change of mapping. Note that we assume that the data held by a node for future computation can be transmitted in one unit of time, which is true for some algorithms such as sorting. If the required transmission time is increased, the overhead for emulation is simply increased accordingly. Since

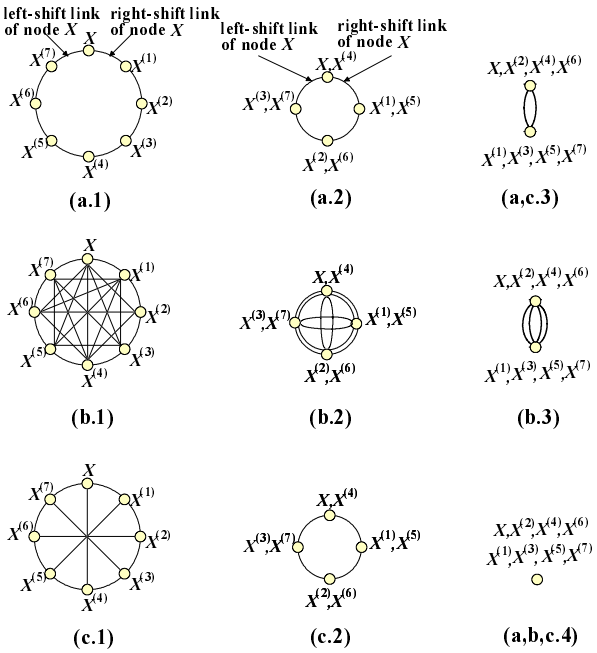

Figure 3. Various CS graphs and degenerate CS graphs containing a node X within 8-level CPNs. (a) (degenerate) CS graphs that are simple rings. (b) (degenerate) CS graphs that are complete graphs. (c) (degenerate) CS graphs that are chordal rings. (x.1) for node X with $X\neq X^{(4)}$. (x.2) for node $X=X_{8:1}$ with $X_{8:5}=X_{4:1}$ and $X\neq X^{(2)}$. (x.3) for node X with $X_{8:7}=X_{6:5}=X_{4:3}=X_{2:1}$ and $X\neq X^{(1)}$. (x.4) for node X with $X=X^{(1)}$ (i.e., $X_i=X_j$, $i,j=1,2,\ldots,8$).

$\frac{T}{t}-1$ additional steps are required for routing along cyclic-shift links on the average during T steps of the normal mesh algorithm the slowdown factor is $1+\frac{1}{t}$. □

In many normal mesh algorithms of interest t is close to 9 so the slowdown factor is close to 1.1.

3 Enhanced Cyclic Petersen Networks

In this section we generalize the definition of basic CPNs to enhanced CPNs. We derive a variety of efficient embeddings and emulation algorithms for them and propose a scheme for routing in enhanced CPNs with balanced utilization over network links.

3.1 Definition of Enhanced CPNs

An enhanced CPN is usually obtained by adding more links to the original CS graphs (i.e., the l-node rings) of a ring-CPN. More precisely, the original CS graph formed by nodes X, $X^{(1)}$, $X^{(2)}$,..., $X^{(l-1)}$ and their shift links

are replaced by another graph (or hypergraph), such as a complete graph or a chordal ring, that connects nodes $X, X^{(1)}, X^{(2)}, \ldots, X^{(l-1)}$; a degenerate CS graph is replaced by the degenerate version of the new CS graph. The nucleus links remain unchanged.

In what follows we formally define enhanced CPNs that use complete graphs as the CS graphs.

Definition 3.1 (Complete-CPN): Let the nucleus graph be a Petersen graph $P = (V_P, E_P)$. An l-level complete-cyclic Petersen network is defined as the graph complete-CPN $= (V, E)$, where $V = \{V_{l:1} | V_i \in V_P, i = 1, \ldots, l, \}$ is the set of vertices, and $E = \{(U_{l:1}, V_{l:1}) | U_i, V_i \in V_P, i = 1, 2, \ldots, l$, satisfying $U_{l:2} = V_{l:2}$ and $(U_1, V_1) \in E_P$, or $V = U^{(i)}$, for some integer $i, 1 \leq i < l\}$ is the set of edges.

The directed link connecting node U to node $V = U^{(i-1)}$ is called link C^i, which corresponds to i right cyclic shifts. Link C^{-i} represent a shift link corresponding to i left cyclic shifts and $C = C^1$. The rule to construct an enhanced CPN using other CS graphs is similar (see Fig. 3 for examples using loop-based topologies [13, 14]). Note that there may be multiple links between two nodes in a degenerate CS graph. For example, node $V = 0101$ in a 4-level complete-CPN is connected to node $U = 1010$ with two links since $V = U^{(1)}$ and $V = U^{(3)}$. Several examples illustrating multiple links in degenerate CS graphs of 8-level CPNs are given in Fig. 3.

A complete-CPN has the strongest embedding and emulation capacity among all CPNs and also the highest node degree, which is equal to $\log_{10} N + 2 \approx 0.3 \log_2 N + 2$. But the degree of a complete-CPN is still a small number (e.g., $d \leq 7$) for networks of practical size (e.g., $N \leq 100$K).

3.2 Embeddings and Algorithms Emulation

Embeddings and emulation between hypercubic networks and for new network topologies have been an important and intensively studied research area [6, 12, 15]. In this subsection, we present efficient emulation algorithms for complete-CPNs under the all-port communication model. We also derive constant dilation embeddings and packings of trees, meshes, tori, hypercubes, meshes of trees, pyramids, generalized hypercubes, folded Petersen networks, as well as any product network in complete-CPNs.

Theorem 3.1 *Any algorithm in a* $10^{l_1} \times 10^{l_2} \times \cdots \times 10^{l_m}$ *mesh under the all-port communication model can be emulated on an* l*-level complete-CPN with a slowdown factor of at most* $\max(4, l+1)$*, where* $\sum_{i=1}^{m} l_i = l$.

Proof: The emulation algorithm under the all-port communication model simply performs single-dimension emulation for all dimensions at the same time with proper scheduling to avoid congestion. Let W (or E) represent the directed nucleus link that is connected to a node's west neighbor (or east neighbor, respectively), in the emulated l-D mesh. A packet for a dimension-j west (or east) neighbor, $2 \leq j \leq l$, in the emulated mesh will be sent through links C^{j-1}, W, C^{1-j} (or C^{j-1}, E, C^{1-j}, respectively). A possible schedule for emulating an l-D $10 \times 10 \times \cdots \times 10$ mesh can be obtained as follows.

We first consider the case when l is odd.

- At time 1, each node sends the packets for its dimension-1 neighbors (in the emulated mesh) through links W and E.
- At time 1, each node also sends the packets for its west neighbors of even dimension and east neighbors of odd dimension i, $i = 2, 3, 4, \ldots, l$, through links C^{i-1}.
- At time 2, each node sends the packets for its east neighbors of even dimension and west neighbors of odd dimension i, $i = 2, 3, 4, \ldots, l$, through links C^{i-1}.
- At even time $t = 2, 4, \ldots, l-1$, each node forwards the packets for its west neighbors of even dimension t through links W and then at the next time step, each node forwards the packets through links C^{1-t}.

Links of a complete-CPN	Dimension j of the mesh being emulated									
	1		2		3		4		5	
	W	E	W	E	W	E	W	E	W	E
Step 1	W	E	C^1	—	—	C^2	C^3	—	—	C^4
Step 2			W	C^1	C^2	E	—	C^3	C^4	—
Step 3			C^4	E	W	C^3	—	—	—	—
Step 4				C^4	C^3		W	—	—	E
Step 5								C^2	E	W	C^1
Step 6									C^2	C^1

(a)

Links of a complete-CPN	Dimension j of the mesh being emulated							
	1		2		3		4	
	W	E	W	E	W	E	W	E
Step 1	W	E	C^1	—	—	C^2	C^3	—
Step 2			W	C^1	C^2	E	—	C^3
Step 3			C^3	—	W	C^2	—	E
Step 4				E	C^2		W	C^1
Step 5					C^3			C^1

(b)

Figure 4. Schedules for emulating meshes on complete-CPNs under the all-port communication model. Note that a certain link appears at most once in a row, and each column for dimension $j \geq 2$ consists of links C^{j-1}, W, C^{1-j} or C^{j-1}, E, C^{1-j}. (a) Emulating a 5-dimensional mesh on a 5-level complete-CPN. (b) Emulating a 4-dimensional mesh on a 4-level complete-CPN.

- At even time $t = 2,4,\ldots,l-1$, each node also forwards the packets for its east neighbors of odd dimension $t+1$ through links E and then at the next time step, each node forwards the packets through links C^{-t}.
- At odd time $t = 3,5,\ldots,l$, each node forwards the packets for its east neighbors of even dimension $t-1$ through links E and then at the next time step, each node forwards the packets through links C^{2-t}
- At odd time $t = 3,5,\ldots,l$, each node forwards the packets for its west neighbors of odd dimension t through links W and then at the next time step, each node forwards the packets through links C^{1-t}.

Figure 4a shows such a schedule for emulating a 5-dimensional mesh on a 5-level complete-CPN.

In what follows we extend the previous schedule to the case when l is even and $l \geq 4$. We initially start with the schedule for an $(l+1)$-level complete-CPN. Clearly, the transmissions corresponding to the emulation of dimension $l+1$ in the initial schedule are not used by the l-level complete-CPN. Therefore, we can now reschedule the link E of dimension l (from time l) to time $l-1$. We then swap the time for the rescheduled link E with that of a link E of smaller dimension. Due to the previous modifications for dimensions j, we also have to modify the schedule for some links C^{1-j} and maybe links C^{j-1}. In particular, we will move link C^{1-j} to the step after the use of link E for the emulation of dimension-l link E in the mesh. As a result, the time required for emulation under the all-port communication model is equal to $l+1$ when $l \geq 4$, and is equal to 4 when $l = 2$ or 3. Figure 4b shows such a schedule for emulating a 4-dimensional mesh on a 4-level complete-CPN.

Since an m-D $10^{l_1} \times 10^{l_2} \times \cdots \times 10^{l_m}$ mesh is a subgraph of an l-D $10 \times 10 \times \cdots \times 10$ mesh, the results follow. □

An l-level folded Petersen network [11], a recently proposed competitor for the hypercube, is defined as $\underbrace{P \times P \times \cdots P}_{l}$, which is the iterative Cartesian product on the Petersen graph P. An l-level folded Petersen network is symmetric and has node degree $3l \approx 0.9\log_2 N$, diameter $2l \approx 0.6\log_2 N$, and average distance about $1.5l \approx 0.45\log_2 N$, all of which are smaller than those of a similar-size hypercube. Some efficient algorithms have been developed for folded Petersen networks [5, 10, 11].

Theorem 3.2 *Any algorithm in an l-level folded Petersen network under the all-port communication model can be emulated on an l-level complete-CPN with a slowdown factor of* $\max(6, l+1)$.

Proof: The proof is similar to the proof of Theorem 3.1 and the corresponding proofs in [19, 21]. □

In what follows, we present constant dilation embeddings and packings of a variety of popular topologies in complete-CPNs.

Theorem 3.3 *An l-level complete-CPN can embed a $10^{l_1} \times 10^{l_2} \times \cdots \times 10^{l_m}$ mesh or an l-level folded Petersen network with load 1, expansion 1, and dilation 3, where $\sum_{i=1}^{m} l_i = l$.*

Proof: Any link of an l-level folded Petersen network can be mapped to a cyclic shift link, a nucleus link, followed by another cyclic shift link (similar to the emulation of Theorem 3.2). Since an l-level folded Petersen network contains a $10^{l_1} \times 10^{l_2} \times \cdots \times 10^{l_m}$ mesh the results follow. □

Theorem 3.4 *An l-level complete-CPN can pack* $\binom{l}{i}$ *copies of an*

$$\underbrace{m \times m \times \cdots m}_{i} \times \underbrace{(10-m) \times (10-m) \times \cdots (10-m)}_{l-i}$$

torus for each $i = 0,1,2,\ldots,l$ with load 1, expansion 1, and dilation 3, where $m = 8$ or 9. An l-level complete-CPN can pack 2^l copies of a 5-ary l-cube with load 1, expansion 1, and dilation 3.

Proof: Since a Petersen graph can pack two 5-node rings, or an 8-node and a 2-node ring, or a 9-node and a 1-node ring as subgraphs, $\binom{l}{i}$ copies of an $\underbrace{m \times m \times \cdots m}_{i} \times \underbrace{(10-m) \times (10-m) \times \cdots (10-m)}_{l-i}$ torus, $i = 0,1,2,\ldots,l$, form node and edge disjoint subgraphs of an l-level folded Petersen network, where $m = 5,8$, or 9. Therefore, the embedding results follow from Theorem 3.3 for embedding a folded Petersen network in a complete-CPN. Since these tori collectively have 10^l nodes, the expansion is equal to 1. □

Theorem 3.5 *An l-level complete-CPN can embed an l-dimensional radix-10 generalized hypercube, or any l-dimensional Cartesian product network with 10-node factor graphs with load 1, expansion 1, and dilation at most equal to 4.*

Proof: Since the diameter of the Petersen graph is equal to 2, any link of the above graphs can be mapped to a cyclic shift link, two links of a nucleus Petersen graph, followed by another cyclic shift link. □

Theorem 3.6 *An l-level complete-CPN can pack* $\binom{l}{i}$ *copies of a $(3l-2i)$-dimensional hypercube for each $i = 0,1,2,\ldots,l$ with load 1, expansion 1, and dilation 4.*

Proof: Since a 10-node complete graph contains a 3-dimensional hypercube and a 1-dimensional hypercube as subgraphs, $\binom{l}{i}$ copies of a $(3l-2i)$-dimensional hypercube for $i = 0, 1, 2, \ldots, l$, form node and edge disjoint subgraphs of a radix-10 l-dimensional generalized hypercube. The packing result follows from Theorem 3.5 for embedding a generalized hypercube in a complete-CPN. Since these hypercubes collectively have 10^l nodes, the expansion is equal to 1. □

Theorem 3.7 *An l-level complete-CPN can pack a complete binary tree of height $3l-1$ and two complete binary trees of height $3l-4$ with load 1 and dilation 3.*

Proof: It follows from Theorem 3.3 and the fact that these three trees form node and edge disjoint subgraphs of an l-level folded Petersen network [11]. □

Theorem 3.8 *An l-level complete-CPN can pack a $2^{3m-1} \times 2^{3(l-m)-1}$ mesh of trees, two $2^{3m-1} \times 2^{3(l-m)-4}$ meshes of trees, two $2^{3m-4} \times 2^{3(l-m)-1}$ meshes of trees, and four $2^{3m-4} \times 2^{3(l-m)-4}$ meshes of trees with load 1 and dilation 3.*

Proof: It follows from Theorem 3.3 and the fact that these meshes of trees form node and edge disjoint subgraphs of an l-level folded Petersen network [11]. □

Lemma 3.9 *Let t_1, t_2, t_3 be the load, expansion, and dilation for embedding graph G in an l-level folded Petersen network. Then an l-level complete-CPN can embed graph G with load t_1, expansion t_2, and dilation $2t_3+1$.*

Proof: From Theorem 3.3, we know that a link in a folded Petersen network can be mapped to a shift link, a nucleus link, and finally another shift link in a complete-CPN. It can be seen that two connected links can be mapped to a shift link, a nucleus link, a shift link, a nucleus link, and finally another shift link, since two shift links in a CS graph of a complete-CPN (which is an l-node complete graph) can be replaced by a shift link. By induction, a path consisting of t links in a folded Petersen network can be mapped to a path consisting of $2t_3+1$ links in a complete-CPN. □

Theorem 3.10 *An l-level complete-CPN can pack $\binom{l}{i} \cdot 2^l + \binom{l}{i} \cdot 2^{l-1}$ copies of a $2^i \times 2^i$ pyramid for all $i = 0, 1, 2, \ldots, l$ with load 1, expansion smaller than 1.25, and dilation 5.*

Proof: It follows from Lemma 3.9 and the fact that these pyramids can be packed in an l-level folded Petersen network with load 1 and dilation 2 [11]. These pyramids have $\frac{4}{5}10^l + 2^{2l-1}$ nodes collectively so the expansion is smaller than 1.25. □

The embeddings presented in this subsection can be easily extended to other classes of CPNs. For example, when $l = 2$ or 3, ring-CPNs are the same as complete-CPNs, so these embedding and emulation results can be directly applied to them. When $l = 4$ or 5, the dilations for embedding and packing the previous networks (from Theorems 3.3 to 3.10) in ring-CPNs are only increased by 2 (additively), except for the embedding of pyramids, whose dilation is increased by 3 (additively). Since networks of size smaller than or equal to 100K seem to be sufficient for interconnection networks in the near future, the dilations for embedding these popular topologies in any type of CPNs are all small numbers in practice. Since many efficient algorithms have been designed for the guest graphs considered in this subsection [6, 12] and the proposed embeddings and emulation algorithms are quite efficient, we can obtain a vast variety of efficient algorithms for CPNs through embeddings and emulation.

Although enhanced CPNs using complete graphs have good performance for algorithm emulation, their node degrees will vary with the number of levels l. Therefore, it may be desirable to use loop-based networks [14] or other small networks as the CS graph (see Fig. 3bd) to obtain networks whose cost and performance fall between those of ring-CPN and complete-CPN.

3.3 Routing with Balanced Traffic

The routing algorithm presented in Subsection 2.2 can be applied to enhanced CPNs without modification as long as the new (degenerate) CS graph contains a Hamiltonian cycle. However, the traffic over network links is not balanced since the additional shift links in the CS graphs are underutilized. In this subsection, we introduce a routing scheme for enhanced cyclic Petersen networks that can uniformly utilize network links.

Assume that a nucleus Petersen graph or several nucleus Petersen graphs are placed within the same module (e.g., a chip, board, or multi-chip module (MCM)), then nucleus links become on-module links and all or most shift links become off-module links. If we can balance the traffic on shift links, then we can always find an appropriate bandwidth for on-module links so that no network link is congested. It is reasonable to make on-module links faster than this required bandwidth to further improve the performance since it is relatively cheaper to implement on-module links with higher bandwidth and the number of transmissions over nucleus links is larger than that over shift links.

Recall that the routing algorithm Route(X,Y) given in Subsection 2.2 is composed of repeated routing within a nucleus and transmission over a shift link C (or C^{-1}) for $l-1$ iterations, followed by routing within a nucleus. This algorithm works because the shift links C or C^{-1} bring each digit of the address of the source node X to the rightmost position exactly once. That is, shift links $\underbrace{CC\cdots C}_{l-1}$ bring the 2nd, 3rd, ... , l^{th} digits to the rightmost position (in that order); shift links $\underbrace{C^{-1}C^{-1}\cdots C^{-1}}_{l-1}$ bring the l^{th}, ... , 3rd, 2nd, digits to the rightmost position. Similarly, we can find a routing algorithm for the enhanced CPN if and only if we can find a sequence of shift links that can bring each digit of the address of the source node X to the rightmost position at least once.

When l is a prime number, $l-1$ shift links C^i for any $i=1,2,\ldots,l-1$ can be used for routing. For example, when $l=5$, four shift links C^2 bring the 3th, 5th, 2nd, and finally the 4th digits to the rightmost position; four shift links C^3 bring the 4th, 2nd, 5th, and finally the 3rd digits to the rightmost position. Therefore, as long as we use shift links C^i for routing with probability $\frac{1}{l-1}$, the traffic among all the shift links of the l-level complete-CPN is exactly balanced, assuming uniformly distributed destinations. Note that the last digit brought to the rightmost position should be "corrected" to be equal to the 1st digit of the destination node Y (by routing within the nucleus to which the destination node Y belongs). If it is initially the i^{th} digit of the source node X, then the j^{th} digit of the source node X should be "corrected" to be equal to the $(j-i+1)^{th}$ digit of the destination node Y.

When l is not a prime number, routing with balanced utilizations is somewhat more complicated. We can see that when $l=4$, applying shift links C^2 alone cannot bring the 2nd or the 4th digits to the rightmost position, so we need a different routing algorithm. Fortunately, we can always find a combination of different classes of shift links that accomplish the job. For example, when $l=4$, we can use shift links $C^2C^1C^2$ or $C^2C^3C^2$ for routing. We can also use $l-1$ shift links C (or $C^{-1}=C^3$). If we assign probability $1/4$ to each of the above sequences for routing, it can be seen that the utilizations for these three shift links are the same; more precisely, the average number of shift links C^i that will be used for routing a packet is slightly smaller than 1 for each $i=1,2,3$. When $l=6$, we can, again, use $l-1$ shift links C^1 (or $C^{-1}=C^5$) for routing. We can also use one of the following 4 sequences

$$C^2C^2C^1C^2C^2, C^4C^4C^5C^4C^4, C^3C^1C^3C^1C^3, \text{ or } C^3C^5C^3C^5C^3$$

for routing. If we assign probability $\frac{1}{12},\frac{1}{12},\frac{1}{4},\frac{1}{4},\frac{1}{6},\frac{1}{6}$ for the above sequences to be used, the utilizations for these five shift links will be the same; more precisely, the average number of shift links C^i that will be used for a routing task is equal to 1 for each $i=1,2,\ldots,5$. Routing in complete-CPNs of higher level can be done is a similar manner. In fact, we may omit the first j iterations in algorithm Route(X,Y) when the least significant digits of the source $X_jX_{j-1}\cdots X_1$ happen to be the same as the most significant digits of the destination $Y_lY_{l-1}\cdots Y_{l-j+1}$. If we take advantage of this property, the expected traffic on the cyclic-shift links becomes slightly different. However, a set of probabilities that are slightly different from the previous ones can always be found to exactly balance the expected traffic, leading to the following theorem.

Theorem 3.11 *There exist a set of sequences of shift links and a corresponding set of probabilities for routing in a complete-CPN such that the traffic among all shift links of the complete-CPN is exactly balanced.*

When the destinations are not uniformly distributed over all network nodes, we may need to adjust the probabilities for the sequences to be used. For example, when a task in the enhanced CPN is emulating a normal mesh algorithm, the task will generate a considerable amount of traffic over shift links C and C^{-1}. Therefore, we have to use sequences that involve fewer or no shift links C or C^{-1} more frequently when performing other routing tasks, in order to balance the utilizations of shift links. This can be done in enhanced CPNs since only a subset of shift-link classes are required when routing a packet and we have several choices for the combination of shift links to be used.

The routing strategies proposed in this subsection can be easily generalized to other classes of enhanced CPNs. For example, consider $l=6$ with the CS graph of the CPN being a degree-3 chordal ring; that is, it has three shift links C^1, C^3, and C^5. We can use one of the following 4 sequences $C^1C^1C^1C^1C^1$, $C^5C^5C^5C^5C^5$, $C^3C^1C^3C^1C^3$, or $C^3C^5C^3C^5C^3$ for routing. If we assign probability $1/4$ for each of the above sequences, it can be seen that the utilizations for these three shift links are approximately the same. These techniques can also be applied to general cyclic-shift networks [17, 18].

4 Clustered CPNs – A Scalable Variant

Although CPNs have good performance and embedding capabilities, their size increases by a factor of 10 with each added level, making it difficult to closely match the network size to the need for computational power. In this subsection, we present a method for obtaining variants of CPNs with smaller step size.

A 2-level CPN is built from 10 nuclei, each of which is a Petersen graph. To obtain a smaller network, we can simply remove some of its nuclei; the resultant network is called a *2-level clustered CPN*.

Theorem 4.1 *The diameter of a 2-level clustered CPN is at most 5.*

Proof: Let $X = X_2X_1$ and $Y = Y_2Y_1$ be the addresses of the source and destination nodes. The routing algorithm Route(X,Y) for a 2-level CPN will send the packet out of nucleus X_2 to nucleus Y_2. Since nucleus Y_2 cannot be one of the removed nuclei, the algorithm Route(X,Y) is directly applicable to a clustered CPN. Therefore, the diameter of a 2-level CPN is upper bounded by 5. □

Note that the removed links of the network nodes can be reconnected to further reduce the average distance and/or diameter and to improve the fault tolerance properties. To obtain higher level CPNs with small step size, we refer the reader to [17, 18] for several possible strategies. Other variants of CPNs can be found in [16].

5 Conclusion

In this paper, we have presented cyclic Petersen networks as efficient extension of the Petersen graph for small- to large-scale parallel processing. We derived efficient embeddings and packings of meshes, tori, meshes of trees, folded Petersen networks, hypercubes, generalized hypercubes and pyramids for CPNs. We also developed algorithms for balanced routing and efficient emulations in them.

References

[1] Bhuyan, L.N. and D.P. Agrawal, "Generalized hypercube and hyperbus structures for a computer network," *IEEE Trans. Comput.*, vol. 33, no. 4, Apr. 1984, pp. 323-333.

[2] Cypher, R. and J.L.C. Sanz, "Hierarchical shuffle-exchange and de Bruijn networks," *Proc. IEEE Symp. Parallel and Distributed Processing*, 1992, pp. 491-496.

[3] Das, S.K. and A.K. Banerjee, "Hyper Petersen networks: yet another hypercube-like topology," *Proc. Symp. Frontiers of Massively Parallel Computation* 1992, pp. 270-277.

[4] Das, S.K., S.R. Öhring, and A.K. Banerjee, "Embeddings into hyper Petersen networks: yet another hypercube–like interconnection topology," *Journal of VLSI Design*, Vol. 2, no. 4, pp. 335-351, 1995.

[5] Das, S.K., D.H. Hohndel, M. Ibel, and S.R. Öhring, "Efficient communication in folded Petersen networks," *Int'l J. Foundations of Computer Science*, vol. 8, no. 2, Jun. 1997, pp. 163-185.

[6] Leighton, F.T., *Introduction to Parallel Algorithms and Architectures: Arrays, Trees, Hypercubes*, Morgan-Kaufman, San Mateo, CA, 1992.

[7] Öhring, S. and S.K. Das, "The folded Petersen network: a new communication–efficient multiprocessor topology," *Proc. Int'l Conf. Parallel Processing* 1993, Vol. I, pp. 311-314.

[8] Öhring, S., S.K. Das and D.H. Hohndel, "Scalable interconnection networks based on the Petersen graph," *Proc. Int'l Conf. Parallel and Distributed Computing Systems* 1994, pp. 581-586.

[9] Öhring, S. and S.K. Das, "Efficient communication on the folded Petersen interconnection networks," *Proc. Parallel Architectures and Languages Europe*, 1994, pp. 689-700.

[10] Öhring, S., D.H. Hohndel, and S.K. Das, "Fault tolerant communication algorithms on the folded Petersen networks based on arc-disjoint spanning trees," *Proc. Conpar 94/VAPP VI*, Vol. 854, 1994, pp. 749-760.

[11] Öhring, S. and S.K. Das, "Folded Petersen cube networks: new competitors for the hypercubes," *IEEE Trans. Parallel Distrib. Sys.*, vol. 7, no. 2, Feb. 1996, pp. 151-168.

[12] Parhami, B., *Introduction to Parallel Processing: Algorithms and Architectures*, Plenum Press, 1999.

[13] Parhami, B. and D.-M. Kwai, "Periodically regular chordal rings," *IEEE Trans. Parallel Distrib. Sys.*, vol. 10, Jun. 1999, pp. 658-672.

[14] Raghavendra, C.S., M. Gerla, and A. Avizienis, "Reliable loop topologies for large local computer networks," *IEEE Trans. Comput.*, vol. C-34, Jan. 1985, pp. 46-55.

[15] Schwabe, E.J., "Efficient embeddings and simulations for hypercubic networks," Ph.D. dissertation, Dept. Electrical Engineering and Computer Science, Massachusetts Institute of Technology, 1991.

[16] Yeh, C.-H. and B. Parhami, "Cyclic Petersen networks: efficient fixed-degree interconnection networks for large-scale multicomputer systems," *Proc. Int'l Conf. Parallel and Distributed Processing: Techniques and Applications*, 1996, pp. 549-560.

[17] Yeh, C.-H. and B. Parhami, "Cyclic networks – a family of versatile fixed-degree interconnection architectures," *Proc. Int'l Parallel Processing Symp.*, Apr. 1997, 739-743.

[18] Yeh, C.-H., "Efficient low-degree interconnection networks for parallel processing: topologies, algorithms, VLSI layouts, and fault tolerance," Ph.D. dissertation, Dept. Electrical & Computer Engineering, Univ. of California, Santa Barbara, Mar. 1998.

[19] Yeh, C.-H. and E.A. Varvarigos, "Macro-star networks: efficient low-degree alternatives to star graphs," *IEEE Trans. Parallel Distrib. Sys.*, vol. 9, no. 10, Oct. 1998, pp. 987-1003.

[20] Yeh, C.-H. and B. Parhami, "The index-permutation graph model for hierarchical interconnection networks," *Proc. Int'l Conf. Parallel Processing*, Sep. 1999, to appear.

[21] Yeh, C.-H. and B. Parhami, "A unified model for hierarchical networks based on an extension of Cayley graphs," *IEEE Trans. Parallel Distrib. Sys.*, to appear.