Gossiping Large Packets on Full-Port Tori

Ulrich Meyer and Jop F. Sibeyn

Max-Planck-Institut für Informatik
Im Stadtwald, 66123 Saarbrücken, Germany.
umeyer, jopsi@mpi-sb.mpg.de
http://www.mpi-sb.mpg.de/~umeyer, ~jopsi/

Abstract. Near-optimal gossiping algorithms are given for two- and higher dimensional tori. It is assumed that the amount of data each PU is contributing is so large, that start-up time may be neglected. For two-dimensional tori, an earlier algorithm achieved optimality in an intricate way, with a time-dependent routing pattern. In our algorithms, in all steps, the PUs forward the received packets in the same way.

1 Introduction

Meshes and Tori. One of the most thoroughly investigated interconnection schemes for parallel computation is the $n \times n$ *mesh*, in which n^2 processing units, *PUs*, are connected by a two-dimensional grid of communication links. Its immediate generalizations are d-dimensional $n \times \cdots \times n$ meshes. Despite their large diameter, meshes are of great importance due to their simple structure and efficient layout.

Tori are the variant of meshes in which the PUs on the outside are connected with "wrap-around links" to the corresponding PUs at the other end of the mesh, thus tori are node symmetric. Furthermore, for tori the bisection width is twice as large as for meshes, and the diameter is halved. Numerous parallel machines with two- and three-dimensional mesh and torus topologies have been built.

Gossiping. Gossiping is a fundamental communication problem used as a subroutine in parallel sorting with splitters or when solving ordinary differential equations: Initially each of the P PUs holds one packet, which must be routed such that finally all PUs have received all packets (this problem is also called *all-to-all broadcast*).

Earlier Work. Recently Šoch and Tvrdík [5] have analyzed the gossiping problem under the following conditions: Packets of unit size can be transferred in one *step* between adjacent PUs, *store-and-forward model.* In each step a PU can exchange packets with all its neighbors, *full-port model.*

The store-and-forward model gives a good approximation of the routing practice, if the packets are so large that the start-up times may be neglected. If an algorithm requires $k > 1$ packets per PU, then this effectively means nothing more than that the packets have to be divided in k chunks each.

In [5], it was shown that on a two-dimensional $n_1 \times n_2$ torus, the given problem can be solved in $\lceil (n_1 \cdot n_2 - 1)/4 \rceil$ steps if $n_1, n_2 \geq 3$. This is optimal. The algorithm is based on time-arc-disjoint broadcast trees: The action of each PU is time dependent, and thus, for every routing decision, a PU has to perform some non-trivial computation (alternatively these actions can be precomputed, but for a d-dimensional torus with P PUs this requires $\Theta(P)$ storage per processor).

New Results. In this paper, we analyze the same problem as Šoch and Tvrdík. Clearly, we cannot improve their optimality. Instead, we try to determine the minimal concessions towards simple time-independent algorithms: In our gossiping algorithms, after some $\mathcal{O}(d)$ precomputation on a d-dimensional torus, a PU knows once and for all, that packets coming from direction x_i have to be forwarded in direction x_j, $1 \leq i, j \leq d$. Time-independence ensures that the routing can be performed with minimal delay, for a fixed size network the pattern might even be built into the hardware. Also on a system on which the connections must be somehow switched, this is advantageous.

By routing the packets along d edge disjoint Hamiltonian paths, it is not hard to achieve the optimal number of steps if each PU initially stores $k = d$ packets. Here optimal means a routing time of $k \cdot P/(2 \cdot d)$ steps on a torus with P PUs. For $d = 2$, the schedule presented in Section 2 is particularly simple, and might be implemented immediately. Our scheme remains applicable for higher dimensions: the proof that edge-disjoint Hamiltonian cycles exist [3, 2, 1] is constructive, but these constructions are fairly complicted. Unfortunately, it is not possible to achieve the optimal number of steps using d fixed edge disjoint Hamiltonian paths and only one packet per PU which then circulates concurrently along all d paths. In [4] we show for the two-dimensional case that at least $P/40$ extra steps are needed due to multiple receives.

In Section 3, we investigate a second approach that allows an additional $o(P)$ routing steps. On a d-dimensional torus, it runs in $P/(2 \cdot d) + o(P)$ steps, with only one packet per PU. For $d = 2$, this is almost optimal and time independent. Therefore, this algorithm might be preferable over the one from [5]. More importantly, for higher dimensions, particularly for the practically relevant case $d = 3$, this gives the first simple and explicit construction that achieves close to optimally. In these algorithms, we construct *partial* Hamiltonian cycles: on a d-dimensional torus, we construct d cycles, each of which covers $P/d + o(P)$ PUs. These are such that for every cycle, every PU is adjacent to a PU through which this cycle runs.

2 Optimal-Time Algorithms

In this section we describe a simple optimal gossiping algorithm for two-dimensional $n_1 \times n_2$ tori assuming that initially each PU holds $k = 2$ packets.

Basic Case. The PU with index (i, j) lies in row i and column j, $0 \leq i < n_1$, $0 \leq j < n_2$. PU $(0, 0)$ is located in the upper-left corner. First PU (i, j) determines whether j is odd or even and sets its routing rules as follows:

$$j < n_2 - 1, j \text{ even } : T \leftrightarrow R; B \leftrightarrow L.$$
$$j < n_2 - 1, j \text{ odd } : T \leftrightarrow L; B \leftrightarrow R.$$

Here T, B, L, R designate the directions "top", "bottom", "left" and "right", respectively. By $T \leftrightarrow R$, we mean that the packets coming from above should be routed on to the right, and vice-versa. The other $\leftrightarrow$ symbols are to be interpreted analogously. Only in the special case $j = n_2 - 1$ we apply the rule $T \leftrightarrow R$; $B \leftrightarrow L$. The resulting routing scheme is illustrated in the left part of Figure 1.

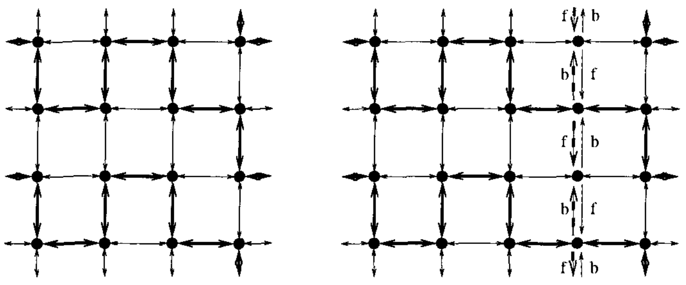

Fig. 1. Left: A Hamiltonian cycle on a 4×4 torus, whose complement (drawn with thin lines) also gives a Hamiltonian cycle. Right: Partial Hamiltonian cycles on a 4×5 torus. The PUs in column 3 lie on only one cycle. Such a PU passes the packets on this cycle that are running forwards to the PU below it, and those that are running backwards to the PU above it.

Odd n_i. Now we consider the case n_1 even and n_2 odd, the remaining cases can be treated similarly. Here we do not construct complete Hamiltonian cycles, but cycles that visit most PUs, and pass within distance one from the remaining PUs. Except for Column $n_2 - 2$, the rules how to pass on the packets are the same as in the basic case. In Column $n_2 - 2$ we perform $L \leftrightarrow R$.

PUs which do not lie on a given cycle, out-of-cycle-PUs, abbreviated *OOC-PUs*, are provided with the packets transferred along this cycle by their neighboring on-cycle-PUs, *OC-PUs*. With respect to different cycles, a PU can be both out-of-cycle and on-cycle. The packets received by an OOC-PU are not forwarded. This can be achieved in such a way, that a connection has to transfer only one packet in every step. The resulting routing scheme is illustrated in the right part of Figure 1.

Changing Direction. Each OOC-PU C receives packets from two different OC-PUs, A and B. Let A transfer the packets that are walking forwards, let B provide the packets in backward direction. If m denotes the cycle length and A and B are not adjacent on the cycle, then $m/2$ circulation steps are not enough: some packets are received from both directions, while others pass by without notice. We modify the algorithm as follows. Let l be the number of OC-PUs between A and B, then during the first l steps, A transfers to C the packets that are running forward, and during the last $m/2 - l$ steps those that run backward. B operates oppositely. In our case $l = 2 \cdot n_2 - 1$ for all PUs in Column $n_2 - 2$. Thus, each PU can still compute its routing decisions in constant time.

Theorem 1 *If every PU of an $n_1 \times n_2$ torus holds 2 packets, then gossiping can be performed in $\lceil n_1 \cdot n_2/2 \rceil$ steps.*

The considered schemes have the advantage that their precomputation can be performed in constant time. Alternatively, one might use the scheme in [3] for the construction of two edge-disjoint Hamiltonian cycles on any two-dimensional torus.

3 One-Packet Algorithms

In this section we give another practical alternative to the approach of [5]: we present algorithms which require only one packet per PU and are optimal to within $o(P)$ steps. The idea is simple: we construct d edge-disjoint cycles of length $P/d + o(P)$. Each of the cycles must have the special property, that if a PU does not lie on such a cycle, this PU must be adjacent to two other PUs belonging to this cycle. Each of these two PUs will transmit the packets from one direction of the cycle to the out-of-cycle-PU.

Two-Dimensional Tori. The construction can be viewed as an extension of the approach described in Section 2. There we interrupted the regular zigzag pattern for one column in order to cope with an odd value for n_2. Now we discard the zigzag in $n_2 - 2$ consecutive columns, the only two remaining zigzag columns connect the long stretched row parts to form two partial cycles:

$$\begin{aligned} j = 0 &: T \leftrightarrow R; B \leftrightarrow L. \\ j = 1 &: T \leftrightarrow L; B \leftrightarrow R. \\ 2 \leq j < n_2 &: L \leftrightarrow R. \end{aligned}$$

Binding the out-of-cycle-PUs is done exactly as in the algorithm of Section 2; the case of odd n_1 and n_2 can be treated by inserting one special row.

Each of the partial cycles consists of $n_1 \cdot n_2/2 + n_1$ PUs, so using both directions in parallel one needs $n_1 \cdot n_2/4 + n_1/2$ steps to spread all the packets within the cycle. For every OOC-PU the two supplying OC-PUs are separated by $n_2 + 1$ other PUs on their cycle, thus we obtain a fully time-independent $n_1 \cdot n_2/4 + \mathcal{O}(n_1 + n_2)$ step algorithm. Applying the forward/backward switching as presented in Section 2 we get

Theorem 2 *If every PU of an $n_1 \times n_2$ torus holds 1 packet, then gossiping can be performed in $n_1 \cdot n_2/4 + n_1/2 + 1$ steps.*

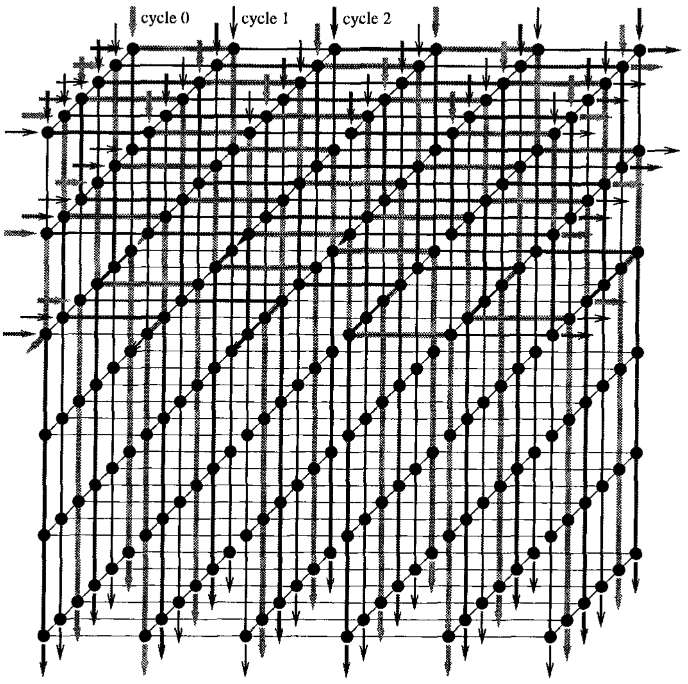

Fig. 2. Three edge-disjoint cycles used in the one-packet algorithm on a three-dimensional torus.

Three-Dimensional Tori. For three-dimensional tori, we generalize the previous routing strategy, showing more abstractly the underlying approach. We assume that n_2 and n_3 are divisible by three. The constructed patterns are similar to the bundle of rods in a nuclear power-plant.

We construct three cycles. These are identical, except that they start in different positions: Cycle j, $0 \leq j \leq 2$, starts in PU $(0, j, 0)$. The cycles for two-dimensional tori were composed of $n_1/2$ *laps*: sections of a cycle starting with a zigzag and ending with the traversal of a wrap-around connection. The zigzags were needed to bring us two positions further in order to connect to the next lap. Here, we have three zigzags, bringing us three positions further. The sequence of directions in the zigzag pattern is given by $(2, 1, 2, 1, 2, 1)$. After these zigzags, moves in direction 1 are repeated until a wrap-around connection is traversed, the next lap can begin.

In this way we can fill up one plane (using $n_2/3$ laps), but in order to get to the next plane, there must be a second zigzag type consisting of the following direction pattern: $(2, 1, 2, 1, 3, 1)$. This type is applied every $n_2/3$ laps. The total schedule is illustrated in Figure 2.

If PU $P = (x, y, z)$, $x \geq 3$ belongs to cycle i, then PUs $P_r = (x, (y + 1) \bmod 3, z)$ and $P_d = (x, y, (z + 1) \bmod 3)$ belong to cycle $(i + 1) \bmod 3$, $P_l =$

$(x, (y-1) \bmod 3, z)$ and $P_u = (x, y, (z-1) \bmod 3)$ belong to cycle $(i-1) \bmod 3$. In this way, P on cycle i can be supplied with packets which are not lying on its own cycle: it receives packets running forward on cycle $(i+1) \bmod 3$ from P_r, backward-packets from P_d. P_l provides packets running backward on cycle $(i-1) \bmod 3$, forward-packets are sent by P_u. Each pair of supporting on-cycle-PUs is separated by $(n_1/3+1) \cdot n_2 - 1$ other PUs.

In the upper part of our reactor, exactly two cycles pass through each PU P, and the shifts are so, that the connections that are not used by cycle traffic lead to PUs that lie on the other cycle. At most $2 \cdot (n_1/3+1) \cdot n_2 - 1$ PUs lie between two supporting on-cycle-PUs. Multiple receives in the OOC-PUs can again be eliminated by the forward/backward switching of Section 2. Otherwise $\mathcal{O}(n_1 \cdot n_2)$ extra steps are necessary.

Theorem 3 *If every PU of an $n_1 \times n_2 \times n_3$ torus (n_2, n_3 integer multiples of three) holds 1 packet, then gossiping can be performed in $n_1 \cdot n_2 \cdot n_3/6 + n_1 \cdot n_2/2 + 1$ steps.*

Higher Dimensional Tori. The idea from the previous section can be generalized without problem for d-dimensional tori. Now, we construct d edge-disjoint cycles, each of them covering $P/d + o(P)$ PUs.

The cycles are numbered 0 through $d-1$. Cycle j starts in position $(0, j, 0, \ldots, 0)$. As before a cycle is composed of laps, starting with a zigzag and ending with moves in direction 1. Now there are $d-1$ types of zigzags, which are used in increasingly exceptional cases. The general pattern telling along which axis a (positive) move is made is as follows:

$$\begin{aligned}
\mathit{zigzag}(2,1) &= (2,1,2,1). \\[6pt]
\mathit{zigzag}(3,1) &= (2,1,2,1,2,1), \\
\mathit{zigzag}(3,2) &= (2,1,2,1,3,1). \\[6pt]
\mathit{zigzag}(i,1) &= (\mathit{zigzag}(i-1,1),2,1), \\
&\vdots \\
\mathit{zigzag}(i,i-2) &= (\mathit{zigzag}(i-1,i-2),2,1), \\
\mathit{zigzag}(i,i-1) &= (2,1,2,1,3,1,4,1,5,1,\ldots,i-1,1,i,1).
\end{aligned}$$

Here $\mathit{zigzag}(d,j)$ indicates the j-th zigzag pattern for d-dimensional tori. During lap i, $1 \le i \le n_2/d \cdot n_3 \cdot \cdots \cdot n_d$, the highest numbered zigzag is applied for which the condition $i \bmod (n_2/d \cdot n_3 \cdot \cdots \cdot n_j) = 0$ is true. Namely, after precisely so many laps, the cycle has filled up the hyperspace spanned by the first j coordinate vectors.

Using induction over the number of dimension we can prove

Theorem 4 *If every PU of a d-dimensional $n_1 \times \cdots \times n_d$ torus ($n_2, \ldots, n_d$ integer multiples of d) with P PUs holds 1 packet, then gossiping can be performed in $P/(2 \cdot d) + P/(2 \cdot n_1) + 1$ steps.*

4 Conclusion

We have completed the analysis of the gossiping problem on full-port store-and-forward tori. In [5] only one interesting aspect of this problem was considered. We have shown that almost equally good performance can be achieved by simpler time-independent algorithms, and given explicit schemes for higher-dimensional tori as well.

References

1. Alspach, B., J-C. Bermond, D. Sotteau, 'Decomposition into Cycles I: Hamilton Decompositions,' *Proc. Workshop Cycles and Rays*, Montreal, 1990.
2. Aubert, J., B. Schneider, 'Decomposition de la Somme Cartesienne d'un Cycle et de l'Union de Deux Cycles Hamiltoniens en Cycles Hamiltonien,' *Discrete Mathematics*, 38, pp. 7–16, 1982.
3. Foregger, M.F., 'Hamiltonian Decomposition of Products of Cycles,' *Discrete Mathematics*, 24, pp. 251–260, 1978.
4. Meyer, U., J. F. Sibeyn, 'Time-Independent Gossiping on Full-Port Tori,' *Techn. Rep. MPI-98-1-014*, Max-Planck Inst. für Informatik, Saarbrücken, Germany, 1998.
5. Šoch, M., P. Tvrdík, 'Optimal Gossip in Store-and-Forward Noncombining 2-D Tori,' *Proc. 3rd International Euro-Par Conference*, LNCS 1300, pp. 234–241, Springer-Verlag, 1997.