An Efficient Parallel Algorithm for Merging in the Postal Model

Hae-Kyeong Park[a], Dong-Hae Chi, Dong-Kyoo Lee, and Kwan-Woo Ryu

Given two sorted lists $A = (a_0, a_1, \cdots, a_{l-1})$ and $B = (b_0, b_1, \cdots, b_{m-1})$, we are to merge these two lists into a sorted list $C = (c_0, c_1, \cdots, c_{n-1})$, where $n = l + m$. Since this is a fundamental problem useful to solve many problems such as sorting and graph problems, there have been many efficient parallel algorithms for this problem. But these algorithms cannot be performed efficiently in the postal model since the communication latency λ, which is of prime importance in this model, is not needed to be considered for those algorithms. Hence, in this paper we propose an efficient merge algorithm in this model that runs in $2\gamma \frac{\log n}{\log(\gamma+1)} + \gamma - 1$ time by using a new property of the bitonic sequence which is crucial to our algorithm. We also show that our algorithm is near-optimal by proving that the lower bound of this problem in the postal model is $f_\gamma(\frac{n}{2})$, where

$$\gamma \frac{\log n - \log 2}{\log(\lfloor \gamma \rfloor + 1)} \leq f_\gamma\left(\frac{n}{2}\right) \leq 2\gamma + 2\gamma \frac{\log n - \log 2}{\log(\lfloor \gamma \rfloor + 1)} .$$

Manuscript received March 31, 1988; revised March 23, 1999.
This work was supported by GRANT No. KOSEF 981-0925-131-2 from Korea Science and Engineering Foundation.

[a]Electronic mail: phk@etri.re.kr

I. INTRODUCTION

In many message-passing systems, such as distributed-memory parallel computers and high-speed communication networks, the exact structure of the underlying communication network may be ignored [1]−[4], [7]−[11], [13], [15], [17]. In such systems it is assumed that the network corresponds to a complete communication graph among processors, in which passing messages is associated with communication latency.

The postal model proposed by Bar-Noy and Kipnis describes well such characteristics of communications in those message-passing systems with the assumption that all the processors are completely connected [4], [5]. This model incorporates a latency parameter $\gamma \geq 1$ that measures the inverse of the ratio between the time when a processor prepares and sends a message to another processor and the time when this recipient processor receives and stores the message. More specifically, we assume that a processor p prepares a message M in a unit time to send it to its destination processor q. Message M later arrives at its destination q, and q spends a unit time to receive and handle it. Each processor p can simultaneously send one message to processor q and receive another message from processor γ. The communication latency γ is the total time during which p prepares message M, sends it to q, and finally q receives and stores M. After processor p sends M, i.e., after a unit time, it is free to perform other functions including preparing other messages.

Bar-Noy and Kipnis developed in this model an optimal algorithm for the single-message broadcasting problem that runs in $f_\gamma(n)$ time [4], and efficient algorithms for the multiple-message broadcasting problem [5]. Park and Ryu developed another optimal algorithm for the single-message broadcasting problem, and presented an optimal algorithm for the prefix-sums problem by utilizing this broadcast algorithm [16].

The merge problem is, given two sorted lists $A=(a_0, a_1,\cdots,a_{l-1})$ and $B=(b_0,b_1,\cdots,b_{m-1})$, to produce a sorted list $C=(c_0,c_1,\cdots c_{n-1})$ of A and B, where $n=l+m$. This problem is a very important problem since an efficient algorithm for this problem is useful to solve the sorting problem, graph problems, and many other problems. A lot of efficient parallel algorithms for this problem have been developed in several parallel computational models. Batcher developed an odd-even merge algorithm that can run in $O(\log n)$ time using $O(n\log n)$ operations on the EREW PRAM model, and can also be implemented on the bitonic merge network with $O(\log n)$ depth and $O(n\log n)$ comparators [6]. Tompson and Kung implemented the odd-even merge algorithm on $\sqrt{n}\times\sqrt{n}$ mesh in $O(\sqrt{n})$ time [19]. Valiant and Kruskal respectively presented an optimal merge algorithm that runs in $\Theta(\log\log n)$ time using $\Theta(n)$ operations on the CREW PRAM model [14], [20]. Snir proved that the merge problem requires at least $\Omega(\log n)$ time on the EREW PRAM model [18], and Hagerup and Rub proposed an optimal merge algorithm that runs in $\Theta(\log n)$ time using $\Theta(n)$ operations on the EREW PRAM model [12].

But these merge algorithms can not be performed efficiently in the postal model since they do not consider the communication latency γ which is of crucial importance in this model. For example, Batcher's odd-even merge algorithm would take $O(\gamma\log n)$ time if it is implemented in the postal model.

Hence, in this paper, we present a new property of the bitonic sequence, and develop by using this property a near-optimal merge algorithm running in $2\gamma\frac{\log n}{\log(\gamma+1)}+\gamma-1$ time. We also prove that this algorithm is almost optimal by showing that the merge problem requires at least $f_\gamma(\frac{n}{2})$ time in the model, where

$$\gamma\frac{\log n-\log 2}{\log(\lfloor\gamma\rfloor+1)}\le f_\gamma(\frac{n}{2})\le 2\gamma+2\gamma\frac{\log n-\log 2}{\log(\lfloor\gamma\rfloor+1)}$$

as shown in [4].

The rest of this paper is organized as follows. In Section II, a new property of the bitonic sequence is presented. In Section III, the lower bound of the merge problem in the postal model and our almost optimal algorithm for solving the problem are described.

II. BITONIC SEQUENCE

Let $X=(x_0,x_1,\cdots,x_{n-1})$ be a sequence of elements drawn from a linearly ordered set. The sequence X is called bitonic if, for some non-negative integers $j, l < n$, we have $x_{j \bmod n}\le x_{(l+2) \bmod n}\le\cdots\le x_{l \bmod n}$ and $x_{(l+1) \bmod n}\ge x_{(l+2) \bmod n}\ge\cdots\ge x_{(l+n-1) \bmod n}$. When $Low(X)$ and $High(X)$ of a bitonic sequence X are defined as

$$Low(X)=(\min\{x_0, x_{\frac{n}{2}}\},\ \min\{x_1, x_{\frac{n}{2}+1}\},\cdots,\ \min\{x_{\frac{n}{2}-1}, x_{n-1}\})\text{ and}$$
$$High(X)=(\max\{x_0, x_{\frac{n}{2}}\},\ \max\{x_1, x_{\frac{n}{2}+1}\},\cdots,\ \max\{x_{\frac{n}{2}-1}, x_{n-1}\}),$$

$Low(X)$ and $High(X)$ are bitonic, and each element of $Low(X)$ is no greater than each element of $High(X)$ [6].

Batcher developed an odd-even merge algorithm that can run in $O(\log n)$ time using $O(n\log n)$ operations on the EREW PRAM model by using this property of the bitonic sequence [6]. However, if the Batcher's algorithm is implemented in the postal model, it requires $\gamma\log n$ time since it does not consider the communication latency γ. Hence, in this section, we propose a new property of the bitonic sequence which will provide a more efficient solution for the merge problem in this model.

In fact, our new property of the bitonic sequence is a generalization of the above property and can be described as follows: Let the i-th subsequence, i-th(X), of a bitonic sequence $X=(x_0,x_1,\cdots,x_{n-1})$ be defined as

$$i\text{-th}(X) = (i\text{-th smallest }\{x_0, x_{\frac{n}{k}},\cdots,x_{(k-1)\frac{n}{k}}\},\ i\text{-th smallest }\{x_1, x_{1+\frac{n}{k}},\cdots,x_{1+(k-1)\frac{n}{k}}\},\cdots,\ i\text{-th smallest }\{x_{\frac{n}{k}-1}, x_{2\frac{n}{k}-1},\cdots,x_{n-1}\}),\quad (1)$$

where k divides n, $0\le i\le k-1$, $2\le k<n$ and i-th smallest $\{a_0,a_1,\cdots,a_{k-1}\}$ is the i-th smallest element in $\{a_0,a_1,\cdots,a_{k-1}\}$. Then, the subsequence i-th(X) is bitonic, and each element of j-th(X) is no less than each element of ($j-1$)-th(X), for $1<j<k$. The above property used in

Batcher's bitonic merge algorithm is just a special case, where $k=2$, of this new property.

Example 1: Given a bitonic sequence X=(7, 9, 12, 15, 18, 17, 13, 8, 5, 1, 3, 6) of $n=12$ elements, where $k=3$ and $0 \leq i \leq 2$, the subsequence i-th(X) is (i-th smallest{7, 18, 5}, i-th smallest{9, 7, 1}, i-th samllest{12, 13, 3}, i-th samllest({15, 8, 6}).

Clearly, each of 0-th(X)=(5, 1, 3, 6), 1-th(X)=(7, 9, 12, 8), and 2-th(X)=(18, 17, 13, 15) is bitonic, and each element of 1-th(X) is no less than each element of 0-th(X), and is no greater than each element of 2-th(X). □

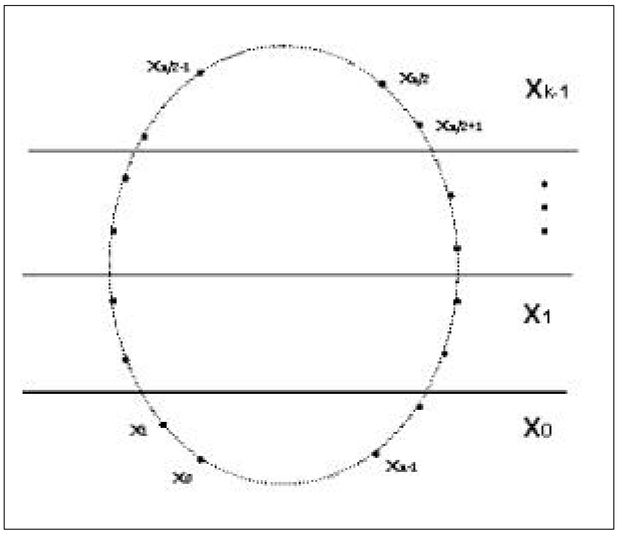


Fig. 1. The bitonic sequence on a circle.

We now prove that our new property of the bitonic sequence is correct. Assume throughout this paper that the bitonic sequence $X=(x_0,x_1,\cdots,x_{n-1})$ consists of distinct elements, and satisfies $n=k^c$, where k and c are positive integers, and $x_0 \leq x_1 \leq \cdots \leq x_l$, $x_{l+1} \geq x_{l+2} \geq \cdots \geq x_{n-1}$, $0 \leq l \leq n-1$. All the results based upon this assumption can be modified to be applied to more general bitonic sequences with no difficulty.

When all the elements of X are placed on a circle, this circle can be partitioned into k subsequences, $X_0, X_1, \cdots, X_{k-1}$, by $k-1$ horizontal lines as in Fig. 1; each subsequence has $\frac{n}{k}$ elements, and each element of X_j is greater than each element of X_{j-1} and is less than each element of X_{j+1}, for $0 < j < k-1$. Each subsequence X_i consists of two monotonic sequences, the increasing sequence $L_i=(x_p,x_{p+1},\cdots,x_{p+s-1})$ and the decreasing sequence $R_i=(x_q,x_{q+1},\cdots,x_{q+t-1})$, where $0 < i < k-1$, $|L_i|=s$, $|R_i|=t$, $0 \leq s,t \leq \frac{n}{k}$, and $s+t=\frac{n}{k}$ (see Fig. 2). It is clear that $X_i=(x_p,x_{p+1},\cdots,x_{p+s-1},x_q,x_{q+1},\cdots,x_{q+t-1})$ is bitonic and consists of the same elements as the i-th sublist $\{x'_{i\frac{n}{k}},x'_{i\frac{n}{k}+1},\cdots,x'_{(i+1)\frac{n}{k}-1}\}$ of the sorted list $X'=(x'_0,x'_1,\cdots,x'_{n-1})$ of X.

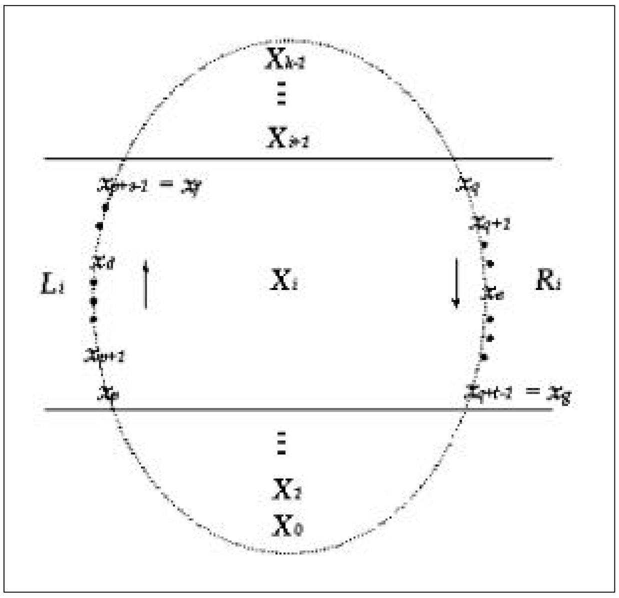


Fig. 2. The subsequence L_i and R_i of X_i.

Example 2: Given a bitonic sequence X=(1, 2, 6, 9, 12, 13, 15, 16, 14, 11, 10, 8, 7, 5, 4, 3), where $n=16$ and $k=4$, the elements of X can be placed on a circle as in Fig. 3, and this circle can be partitioned into 4 subsequences, $X0$=(1, 2, 4, 3), $X1$=(6, 8, 7, 5), $X2$=(9, 12, 11, 10), $X3$=(13, 15, 16, 14), each with $\frac{n}{k}=4$ elements. We can see that each element of X_j is less than each element of X_{j+1}, and is greater than each element of X_{j-1}, for $0 < j < 3$, and that each subsequence is bitonic with two monotonic subsequences. For example, X_0 consists of the increasing subsequence $L_0=(1,2)$ and the decreasing subsequence $R_0=(4,3)$. □

Lemma 1. *Each subsequence* X_i, $0 \leq i \leq k-1$, *is a bitonic sequence.*

Proof: X_i is a concatenated list of L_i and R_i, and L_i is an increasing sequence and R_i is a decreasing sequence. Thus X_i is a bitonic sequence. □

The following lemma shows that, when $mod\frac{n}{k}$ operation is applied to the indices of all the elements of X_i, all the resulting values are distinct. For example, consider $X_1 = (6,\ 8,\ 7,\ 5) = (x_2,\ x_{11},\ x_{12},\ x_{13})$ in Example 1, where $n = 16$ and $k = 4$. When $mod\, 4$ operation is applied to its index list $(2,\ 11,\ 12,\ 13)$, the result is $(2,\ 3,\ 0,\ 1)$, which consist of distinct values. Note also that this is a circular shift of $(0,\ 1,\ 2,\ 3)$.

Lemma 2. *If two elements x_d and x_e of X_i satisfy $d \fallingdotseq e$, then $d\ mod\frac{n}{k} \fallingdotseq e\ mod\frac{n}{k}$, for $0 \le i \le k-1$.*

Proof: Assume, without loss of generality, that $d \langle e$. There are two cases for the proof. The first case is that $x_d \in L_i$ and $x_e \in R_i$. In this case, the elements between L_i and R_i in clock-wise order of the circle in Fig. 1 are all the elements of $X_{i+1}, X_{i+2}, \cdots, X_{k-1}$ and the number of elements is $(k-i-1)\frac{n}{k}$, and $|L_i| + |R_i| = \frac{n}{k}$. Hence it is clear that $d\ mod\frac{n}{k} \fallingdotseq e\ mod\frac{n}{k}$. The second case is that both x_d and x_e are in L_i or R_i. Since $|L_i|, |R_i| \le \frac{n}{k}$, $d\ mod\frac{n}{k} \fallingdotseq e\ mod\frac{n}{k}$. □

Given $X_i = (x_p, x_{p+1}, \cdots, x_{p+s-1}, x_q, x_{q+1}, \cdots, x_{q+t-1})$, let M_i be the list $\{p\ mod\frac{n}{k}, (p+1)\ mod\frac{n}{k}, \cdots, (p+s-1)$

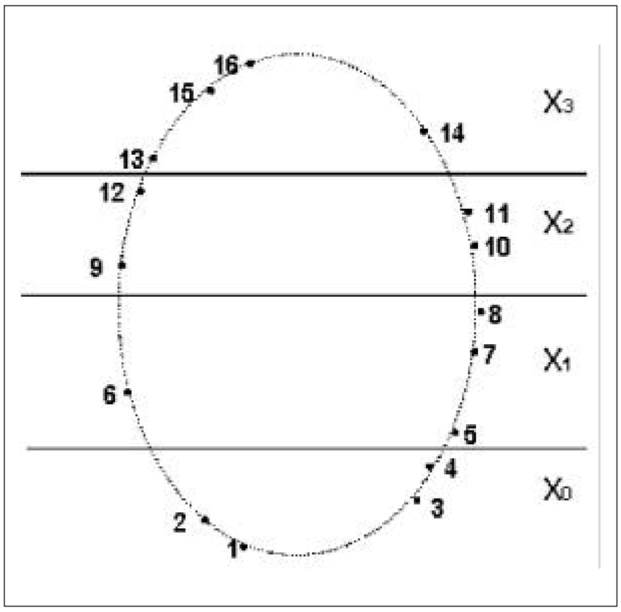


Fig. 3. The bitonic sequence X = (1, 2, 6, 9, 12, 13, 15, 16, 14, 11, 10, 8, 7, 5, 4, 3) on a circle.

$mod\frac{n}{k}, q\ mod\frac{n}{k}, (q+1)\ mod\frac{n}{k}, \cdots, (q+t-1)\ mod\frac{n}{k}\}$, that is, the list of indices of all the elements of X_i applied to $mod\frac{n}{k}$ operation. Then the following lemma shows that M_i is a circularly shifted list of $(0, 1, 2, \cdots, \frac{n}{k}-1)$ (see Fig. 4).

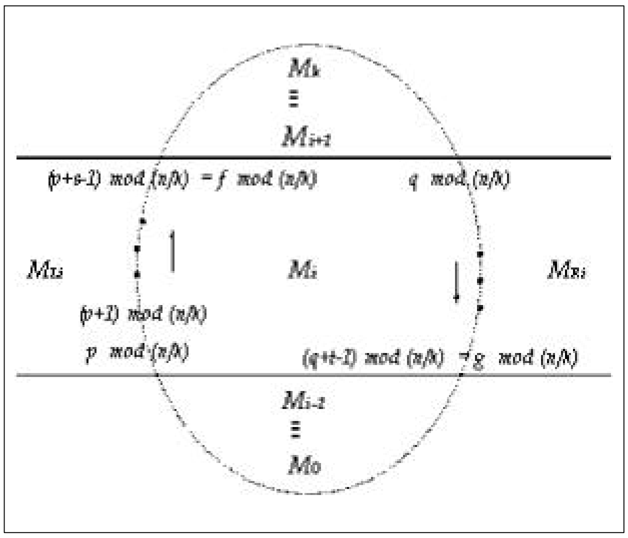

Mk
Mi+1
(p+s-1) mod (n/k) = f mod (n/k)
q mod (n/k)
MLi
Mi
MRi
(p+1) mod (n/k)
p mod (n/k)
(q+t-1) mod (n/k) = g mod (n/k)
Mi-1
M0

Fig. 4. List Mi.

Lemma 3. *Each list M_i, $0 \le i \le k-1$, is a circularly shifted list of $(0,\ 1,\ 2, \cdots, \frac{n}{k}-1)$.*

Proof: Given $L_i = (x_p, x_{p+1}, \cdots, x_{p+s-1})$ and $R_i = (x_q, x_{q+1}, \cdots, x_{q+t-1})$ of X_i let $M_{L_i} = (p \bmod \frac{n}{k}, (p+1)\ mod\frac{n}{k}, \cdots, (p+s-1)\ mod\frac{n}{k})$, and let $M_{R_i} = (q\ mod\frac{n}{k},\ (q+1)\ mod\frac{n}{k}, \cdots, (q+t-1)\ mod\frac{n}{k})$. If $|L_i| = 0$ (or $|R_i| = 0$), R_i (or L_i) is a continuous subsequence of X of size $\frac{n}{k}$, and clearly M_{R_i}(or M_{L_i}) is a circularly shifted list of $(0,\ 1,\ 2, \cdots, \frac{n}{k}-1)$ by definition. If $|L_i| \neq 0$ and $|R_i| \neq 0$, clearly M_{L_i} and M_{R_i} are continuous sublists of a circularly shifted list of $(0,\ 1,\ 2, \cdots, \frac{n}{k}-1)$. Since the number of the elements between the last element $x_f (f = p+s-1)$ of L_i and the first element x_q of R_i is $(k-i-1)\frac{n}{k}$, $q = f + (k-i-1)\frac{n}{k} + 1$ and $q\ mod\frac{n}{k} = (f+1)\ mod\frac{n}{k}$. We can also show in the same way that

$p \bmod \frac{n}{k} = (g+1) \bmod \frac{n}{k}$, where $x_g \,(g = q+t-1)$ is the last element of R_i and x_p is the first element of L_i. Notice that all the elements of M_i ents of are distinct by Lemma 2. Thus, the concatenated list M_i of M_{L_i} and M_{R_i} is a circularly shifted list of $(0, 1, 2, \cdots, \frac{n}{k}-1)$. □

Let Y_j be a subsequence $(x_j, x_{j+\frac{n}{k}}, \cdots, x_{j+(k-1)\frac{n}{k}})$ that consists of every $\frac{n}{k}$ th element from X, for $0 \le j \le \frac{n}{k}-1$. Then Y_j has the following property.

Lemma 4. *Any two elements from a subsequence Y_j do not belong to the same subsequence X_i, where $0 \le i \le \frac{n}{k}-1$ and $0 \le j \le \frac{n}{k}-1$.*

Proof**:** By Lemma 2, any two elements x_d and x_e of X_i with $d \neq e$ satisfy $d \bmod \frac{n}{k} \neq e \bmod \frac{n}{k}$. Clearly the value resulted from appling $\bmod \frac{n}{k}$ operation to each index of $Y_j = (x_j, x_{j+\frac{n}{k}}, \cdots, x_{j+(k-1)\frac{n}{k}})$ is the same j. Hence, any two elements of Y_j can not belong to X_i at the same time.□

Let x_{i_j} be the i-th smallest element of $Y_i = (x_j, x_{j+\frac{n}{k}}, \cdots, x_{j+(k-1)\frac{n}{k}})$ and let $Z_i = (x_{i_0}, x_{i_1}, \cdots, x_{i_{\frac{n}{k}-1}})$, where $0 \le i \le k-1$ and $0 \le j \le \frac{n}{k}-1$. Then Z_i has the following two properties.

Lemma 5. *Z_i and X_i consist of the same elements, for $0 \le i \le k-1$.*

Proof: Note that each element of X_i is less than each element of X_{i+1}, and is greater than each element of X_{i-1}. By Lemma 4, X_i has exactly one of the elements from the sequence Y_j, $0 \le j \le \frac{n}{k}-1$. Thus, i-th smallest element of Y_j must belong to X_i, and hence the lemma follows. □

By this lemma, we can see that each element of Z_i is less than each element of Z_{i+1} and is greater than each element of Z_{i-1}.

Lemma 6. *Each Z_i, $0 \le i \le k-1$, is a bitonic sequence.*

Proof: By Lemma 4, X_i has exactly one element from Y_j, $0 \le j \le \frac{n}{k}-1$. Since the indices of all the elements in Y_j have the same j after applying $\bmod \frac{n}{k}$ operation, and since j-th element x_{i_j} of $Z_i = (x_{i_0}, x_{i_1}, \cdots, x_{i_{\frac{n}{k}-1}})$ is an element from Y_j, $i_j \bmod \frac{n}{k} = j$ and hence the resulting list of applying $\bmod \frac{n}{k}$ operation to the index of each element of Z_i is $(0, 1, 2, \cdots, \frac{n}{k})$. All the elements of Z_i are in X_i by Lemma 5, and M_i corresponding to X_i is a circularly shifted list of $(0, 1, 2, \cdots, \frac{n}{k}-1)$ by Lemma 3. Thus, Z_i is one of circularly shifted sequences of X_i. Since X_i is a bitonic sequence, Z_i is also a bitonic sequence by definition. □

Clearly Z_i is i-th(X) defined in eq.(1) and hence we proved the new property claimed in the beginning of this section. In the above lemmas, we assumed that the bitonic sequence $X = (x_0, x_1, \cdots, x_{n-1})$ satisfies $x_0 \le x_1 \le \cdots \le x_l$, $x_{l+1} \ge x_{l+2} \ge \cdots \ge x_{n-1}$, $0 \le l \le n-1$ and $n = k^c$ for some positive integers k and c, and assumed that all the elements of X are distinct. But all these results are true for general bitonic sequences, since all the lemmas in this section are correct for circularly shifted sequences of X and all the general bitonic sequences can be made by circular-shifting of X. We now present an efficient merge algorithm for the postal model that makes use of this property of the bitonic sequence.

III. THE MERGE ALGORITHM

In this section, we first prove that the merge problem requires at least $f_\gamma(\frac{n}{2})$ time in the postal model, and then we present a near-optimal merge algorithm in the model that runs in $2\gamma \frac{\log n}{\log(\gamma+1)} + \gamma - 1$ time by using the new property described in the previous section.

Theorem 1. *In the postal model with n processors and communication latency γ, it requires at least $f_\gamma(\frac{n}{2})$ time to merge two sorted lists A and B of sizes l and m respectively into a sorted list of size n, where $n = l + m$.*

Proof: Without loss of generality, we assume $l \geq m$. Let A_i be the set of elements in A that are between b_i and b_{i+1} of B. In this case, all the elements of A_i have to be directly or indirectly compared with b_i and b_{i+1} for merging A and B, and it requires at least $f_\gamma(|A_i|)$ time which is the time required for broadcasting an element over $|A_i|$ processors [4]. Since $|A_i|$ can be as large as l and $l \geq \frac{n}{2}$, the merge problem requires at least $f_\gamma(\frac{n}{2})$ time. □

We now describe briefly our merge algorithm in the postal model with n processors and communication latency γ. Given a bitonic sequence $X = (x_0, x_1, \cdots, x_{n-1})$, where $n = k^c$, $k = \gamma + 1$, and c is a positive integer. We first sort each subsequence $Y_j = (x_j, x_{j+\frac{n}{k}}, \cdots, x_{j+(k-1)\frac{n}{k}})$ of X into a sorted sequence $Y'_j = (x'_j, x'_{j+\frac{n}{k}}, \cdots, x'_{j+(k-1)\frac{n}{k}})$, for $0 \leq j \leq \frac{n}{k} - 1$. Let $\overline{X} = (x'_0, x'_1, \cdots, x'_{n-1})$ be the sequence constructed by combining the $\frac{n}{k}$ sorted sequences Y'_j's, and let $Z_i = (x'_{i\frac{n}{k}}, x'_{i\frac{n}{k}+1}, \cdots, x'_{(i+1)\frac{n}{k}-1})$ be the *i*-th continuous subsequence of $\overline{X}$, for $0 \leq i \leq k-1$. Then by Lemmas 5 and 6, each Z_i is bitonic, and each element of $Z_{i'}$ is greater than each element of $Z_{i'-1}$ and is less than each element of $Z_{i'+1}$, for $0 < i' < k-1$. Hence, if we merge each Z_i recursively, we can get a sorted list of X. When the size of the bitonic sequence is no greater than $k\,(=\gamma+1)$, we sort it directly as follows.

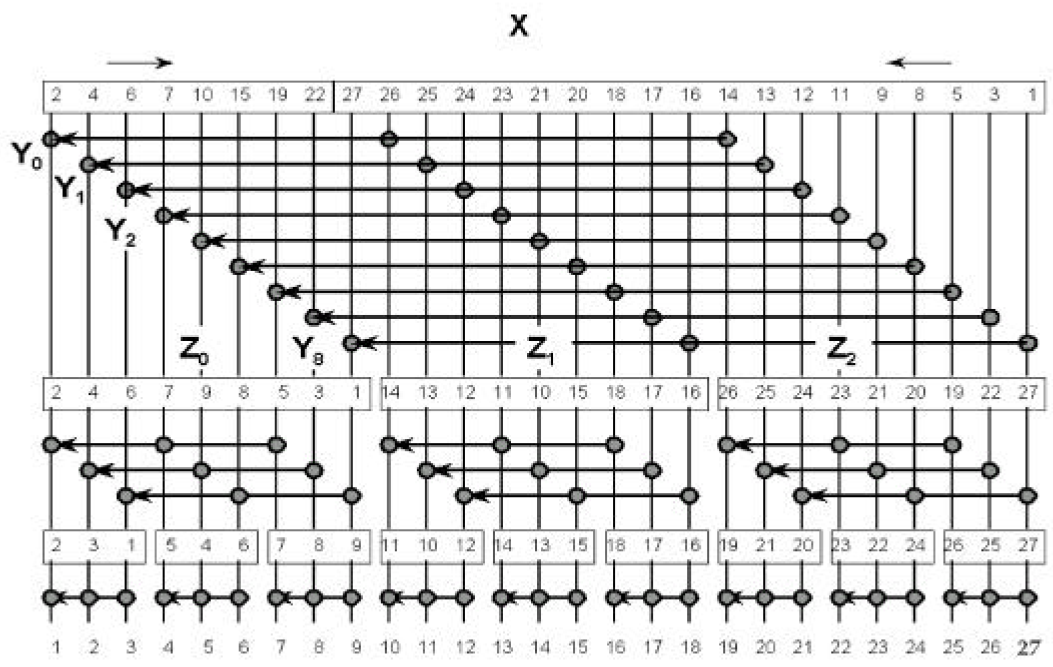
Given $\gamma+1$ elements $x_{j_0}, x_{j_1}, \cdots, x_{j_\gamma}$ in processors $P_{j_0}, P_{j_1}, \cdots, P_{j_\gamma}$ respectively. Each processor P_{j_i} first initializes its counter S_{j_i} to 0, and then sends x_{j_i} to all the other processors P_{j_l} one by one without conflict, where $0 \leq l \leq \gamma$ and $l \neq i$. When P_{j_i} receives x_{j_l}, it increments S_{j_i} by one either if $x_{j_l} < x_{j_i}$ or if $i < l$ and $x_{j_i} = x_{j_l}$. P_{j_i} receives all the γ elements in $2\gamma - 1$ time, and then S_{j_i} has the rank of x_{j_i} among $x_{j_0}, x_{j_1}, \cdots, x_{j_\gamma}$. Finally, P_{j_i} sends x_{j_i} to $P_{S_{j_i}}$, and sorting will be completed in $3\gamma - 1$ time.

The following algorithm is our merge algorithm that sorts a given bitonic sequence $X = (x_0, x_1, \cdots, x_{n-1})$. Each processor P_i initially has x_i, and will finally have the *i*-th smallest element among X.

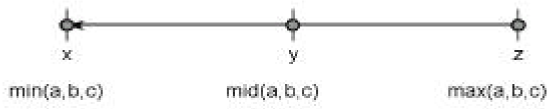
Algorithm Merge

```
Input: A bitonic sequence $X = (x_0, x_1, \cdots, x_{n-1})$, where
        $n = (\gamma+1)^c$ and $c$ is a positive integer.
Output: A sorted list.
  Step 0: If $n \leq \gamma + 1$, then sort the sequence directly and
            return.
          At time $t = 0$, $P_i$ $(0 \leq i \leq n-1)$ sets $S_i = 0$.
          For time $t = 0, n-1$, $P_i$ $(0 \leq i \leq n-1)$
            sends $x_i$ to $P_j$ $(j = (i+t+1) \bmod n)$.
          For time $t = \gamma, \gamma+n-2$,
            if $P_j$ $(0 \leq j \leq n-1)$ receives $M$ and $M > x_j$,
            then $S_j = S_j + 1$.
          At time $t = \gamma+n-2$, $P_j$ $(0 \leq j \leq n-1)$
            sends $x_j$ to $P_{S_j}$.
          At time $t = 2\gamma+n-2$, $P_{S_j}$ $(0 \leq j \leq n-1)$
            receives $M$ and $x_{S_j} = M$.
  Step 1: For $0 \leq j \leq \frac{n}{\gamma+1} - 1$ pardo
            Sort $Y_j = (x_j, x_{j+\frac{n}{\gamma+1}}, \cdots, x_{j+\gamma\frac{n}{\gamma+1}})$ into
            $Y'_j = (x'_j, x'_{j+\frac{n}{\gamma+1}}, \cdots, x'_{j+\gamma\frac{n}{\gamma+1}})$ by
            using the same technique as in Step 0.
            Let the combined sequence of all
            such $\frac{n}{k}$ $Y'_j$'s be $\overline{X} = (x'_0, x'_1, \cdots, x'_{n-1})$.
            Notice that now $x'_i$ is in $P_i$.
  Step 2: For $0 \leq i \leq \gamma$ pardo.
            Merge each consecutive subsequence
            $Z_i = (x'_{i\frac{n}{\gamma+1}}, x'_{i\frac{n}{\gamma+1}+1}, \cdots, x'_{(i+1)\frac{n}{\gamma+1}-1})$
            of $\overline{X}$ recursively.
```

Example 3: Given two sorted lists A = (2, 4, 6, 7, 10, 15, 19, 22) and B = (27, 26, 25, 24, 23, 21, 20, 18, 17, 16, 14, 13, 12, 11, 9, 8, 5, 3, 1), the steps of merging these two lists into a sorted list C = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27) in the postal model with communication latency $\gamma = 2$ are illustrated in Fig. 5. First, the subsequences Y_0 = (2, 26,14), Y_1 = (4, 25, 13), ⋯, Y_8 = (27, 16, 1), each of which consists of every 9th element from X, are sorted respectively into Y'_0 = (2, 14, 26), Y'_1 = (4, 13, 25), ⋯, Y'_8 = (1, 16, 27). The combined sequence $\overline{X}$ of these sorted sequences consists of 3 consecutive bitonic subsequences



(a) the procedure of merge



(b) sorting 3(= λ + 1) elements

Fig. 5. The procedure of merge with $\lambda = 2$ and $n = 27$.

$Z_0 = (2, 4, 6, 7, 9, 8, 5, 3, 1)$, $Z_1 = (14, 15, 12, 11, 10, 15, 18, 17, 16)$, $Z_2 = (26, 25, 24, 23, 21, 20, 19, 22, 27)$ each with 9 elements. Note that each element of Z_1 is greater than each element of Z_0 and is less than each element of Z_2. Next, Z_0, Z_1, and Z_2 are sorted recursively, and we obtain a final sorted list C. □

The following lemma shows that our merge algorithm correctly solves the merge problem.

Lemma 7. *Given a bitonic sequence* $X = (x_0, x_1, \cdots, x_{n-1})$ *, Algorithm Merge produces a sorted sequence correctly in the postal model with* n *processors and communication latency* γ.

Proof: By induction. For $c = 1$, the bitonic sequence of size $\gamma+1$ is sorted directly in Step 0. Next we assume that Algorithm Merge correctly sorts the bitonic sequence of size $n' = (\gamma+1)^{\alpha}$, where α is a positive integer.

Given a bitonic sequence X of size $n = (\gamma+1)^{\alpha+1} = n'(\gamma+1)$, each subsequence $Y_j = (x_j, x_{j+\frac{n}{\gamma+1}}, \cdots, x_{j+\gamma\frac{n}{\gamma+1}})$ is sorted first into $Y'_j = (x'_j, x'_{j+\frac{n}{\gamma+1}}, \cdots, x'_{j+\gamma\frac{n}{\gamma+1}})$ for $0 \le j \le \frac{n}{\gamma+1} - 1$. When $\overline{X} = (x'_0, x'_1, \cdots, x'_{n-1})$ made up of all the Y'_j's, each consecutive subsequence $Z_i = (x'_{i\frac{n}{\gamma+1}}, x'_{i\frac{n}{\gamma+1}+1}, \cdots, x'_{(i+1)\frac{n}{\gamma+1}-1})$ of $\overline{X}$ is bitonic, and each element of $Z_{i'}$ is greater than each element of $Z_{i'-1}$ and is less than each element of $Z_{i'+1}$, where $0 \le i \le \gamma$ and $0 \le i' \le \gamma$, by Lemmas 5 and 6. Hence each Z_i has only to be merged recursively, and since $|Z_i| = n'$, the lemma follows by the induction hypothesis. □

Now we show that Algorithm Merge sorts a bitonic sequence in $2\gamma\frac{\log n}{\log(\gamma+1)}+\gamma-1$ time in the next theorem.

Theorem 2. *Algorithm Merge sorts a given bitonic sequence* $X=(x_0,x_1,\cdots,x_{n-1})$ *, where* $n=(\gamma+1)^c$ *and* c *is a positive integer, in* $2\gamma\frac{\log n}{\log(\gamma+1)}+\gamma-1$ *time in the postal model with* n *processors and communication latency* γ*.*

Proof: The size of the bitonic sequence decreases by a factor of $\frac{1}{\gamma+1}$ with each recursion step, until the size of each subsequence becomes $\gamma+1$. In this case, each subsequence is sorted directly in Step 0. Hence, the depth of the recursion is $\frac{\log n}{\log(\gamma+1)}$. Each recursion step can obviously be performed in $3\gamma-1$ time. But this can be reduced to 2γ time as follows.

Let $Y_j=(x_j,\ x_{j+\frac{n}{\gamma+1}},\cdots,\ x_{j+\gamma\frac{n}{\gamma+1}})=(x_{j_0},\ x_{j_1},\cdots,x_{j_\gamma})$.

Then P_{j_i} computes S_{j_i}, the rank of x_{j_i} among $x_{j_0},x_{j_1},\cdots,\ x_{j_\gamma}$, in $2\gamma-1$ time in Step 1. After this, instead of P_{j_i} sending x_{j_i} only to $P_{S_{j_i}}$ as in Step 1, P_{j_i} directly sends it to appropriate γ processors for the next recursion step as well as to $P_{S_{j_i}}$. Hence the total time for Step 1 in each recursion step is 2γ since each processor sends its element to $\gamma+1$ processors. Thus the total time for Algorithm Merge is

$$2\gamma\left(\frac{\log n}{\log(\gamma+1)}-1\right)+3\gamma-1=2\gamma\frac{\log n}{\log(\gamma+1)}+\gamma-1 \quad . \qquad \square$$

The merge problem in the postal model requires at least $f_\gamma(\frac{n}{2})$ time by Theorem 1. Since our algorithm takes $2\gamma\frac{\log n}{\log(\gamma+1)}+\gamma-1$ time, and $\gamma\frac{\log n-\log 2}{\log(\lfloor\gamma\rfloor+1)}\le f_\gamma(\frac{n}{2})$ $\le 2\gamma+2\gamma\frac{\log n-\log 2}{\log(\lfloor\gamma\rfloor+1)}$ by [4], we can say that our algorithm is an almost optimal algorithm.

Corollary 1. *In the postal model with n processors and communication latency* γ*, Algorithm Merge merges two sorted lists of size* l *and* m *into a sorted list of size* n *in* $2\gamma\frac{\log n}{\log(\gamma+1)}+\gamma-1$ *time, where* $n=l+m$ *and* $n=(\gamma+1)^c$ *, for a positive integer* c*.*

In Algorithm Merge, it is assumed that $n=(\gamma+1)^c$ where c is a positive integer, but this algorithm can be slightly modified to solve the merge problem for more general bitonic sequences.

Corollary 2. *In the postal model with* n *processors and communication latency* γ*, Algorithm Merge merges two sorted lists of size* l *and* m *into a sorted list of size* n *in* $2\gamma\lfloor\frac{\log n}{\log(\gamma+1)}\rfloor+\gamma-1$ *time, where* $n=l+m$ *.*

Our merge algorithm can be used to sort an arbitrary list $X=(x_0,x_1,\cdots,x_{n-1})$ efficiently in the postal model. Each processor P_i initially has x_i, and finally has the *i*-th smallest element among X .

Corollary 3. *In the postal model with* n *processors and communication latency* γ*, sorting* n *elements can be performed in* $(2\gamma\lfloor\frac{\log n}{\log(\gamma+1)}\rfloor+\gamma-1)\lfloor\log n\rfloor$ *time.*

REFERENCES

[1] A. Aggarwal, A. Chandra and M. Snir, "On Communication Latency in PRAM Computations," *Proc. 1st ACM Symposium on Parallel Algorithms and Architectures,* June 1989, pp. 11－21.

[2] A. Aggarwal, A. Chandra and M. Snir, "Hierarchical Memory with Block Transfer," *Proc. 28th Annual Symposium on Foundations of Computer Science,* October 1987, pp. 204－216.

[3] A. Bar-Noy, J. Bruck, C. T. Ho, S. Kipnis and B. Schieber, "Efficient Global Combine Operations in the Multi-Port Postal Model," *IEEE Trans. on Parallel and Distributed Systems,* 6(8), August 1995, pp. 896－900.

[4] A. Bar-Noy and S. Kipnis, "Designing Broadcasting Algorithms in the Postal Model for Message-Passing System," *Proc. 4th Annual Symposium on Parallel Algorithms and Architectures,* ACM, June 1992, pp. 13－22.

[5] A. Bar-Noy and S. Kipnis, "Multiple Message Broadcasting in the Postal Model," *Proc. 7th International Parallel Processing Symposium,* April 1993, pp. 463－470.

[6] K. Batcher, "Sorting Networks and Their Applications," *AFIPS Spring Joint Computing Conference,* Atlantic City, NJ, 1968, pp. 307－314.

[7] J. Bruck, L. D. Coster, N. Dewulf, C. Ho, and R. Lauwereins, "On the Design and Implementation of Broadcast and Global Combine Operations Using the Postal Model," *IEEE Transaction on Parallel and Distributed Systems,* 7(3), March 1996, pp. 256－265.

[8] I. Cidon and I. Gopal, "PARIS: An Approach to Integrated High-Speed Private Networks," *International Journal of Digital and Analog Cabled Systems,* 1(2), April-June 1988, pp. 77－85.

[9] D. Clark, B. Davie, D. Farber, I. Gopal, B. Kadaba, D. Sincoskie, J. Smith, and D. Tennenhouse, "The AURORA Gigabit Testbed," *Computer Networks and ISDN,* 1991.

[10] D. E. Culler, R. Karp, D. Patterson, A. Sahay, K. E. Schauser, E. Santos, R.Subramonian, and T. von Eicken, "LogP: Towards a Realistic Model of Parallel Computation," *Proc. 4th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming,* May 1993, pp. 1－12.

[11] W. J. Dally, A. Chien, S. Fiske, W. Horwat, J. Keen, M. Larivee, R. Lethin, P. Nuth, S. Wills, P. Carrick, and G. Fyler, "The J-Machine: A Find-Grain Concurrent Computer," *Information Processing 89,* Elsevier Science Publishers, IFIP, 1989, pp. 1147－1153.

[12] T. Hagerup and C. Rub, "Optimal Merging and Sorting on the EREW PRAM," *Information Processing Letters 33,* December 1989, pp. 181－185.

[13] J. F. JaJa and K. W. Ryu, "The Block Distributed Memory for Shared Memory Multiprocessor," *Proc. 8th International Parallel Processing Symposium,* April 1994 pp. 752－756.

[14] C. P. Kruskal, "Searching, Merging, and Sorting in Parallel Computation," *IEEE Transaction on Computers,* c-32(10), October 1983, pp. 942－946.

[15] C. E. Leiserson, Z. S. Abuhamdeh, D. C. Douglas, C. R. Feynman, M. N. Ganmukhi, J. V. Hill, W. D. Hillis, B. C. Kuszmaul, M. A. St. Pierre, D. S. Wells, M. C. Wong, S. W. Yang, and R. Zak, "The Network Architecture of the Connection Machine CM-5," *Proc. 4th Annual Symposium on Parallel Algorithms and Architectures*, ACM, June 1992, pp. 272－285.

[16] H. K. Park and K. W. Ryu, "An Optimal Parallel Algorithm for Solving the Prefix-Sums Problem in the Postal Model," *The 4th Annual Australasian Conference on Parallel And Real-Time Systems (PART'97),* September 1997, pp. 181－190.

[17] S. Report and R. Special, "Gigabit Network Testbeds," *Special Report, IEEE Computer Magazine,* 23(9), 1990, pp. 77－80.

[18] M. Snir, "On Parallel Searching," *SIAM Journal of Computing,* 14, 1985, pp. 688－708.

[19] C. D. Thompson and H. T. Kung, "Sorting on a Mesh-Connected Parallel Computer," *Communications of the ACM,* 20(4), April 1977, pp. 263－271.

[20] L. G. Valiant, "Parallelism in Comparison Problems," *SIAM Journal of Computing,* 4(3), 1975, pp. 348－355.

Hae-Kyeong Park received the B.S. degree in computer science from the Changwon National University, Changwon in 1990, and the M.S. and Ph.D. degree in computer engineering from the Kyungpook National University, Taegu in 1992 and 1997, respectively. In 1997, she joined ETRI as a Senior Member of Research Staff. She is presently in Internet Technology Department, Switching and Transmission Laboratory, ETRI. Her research interests are design and analysis of parallel algorithms for combinatorial problems and fast address lookup algorithm for IP routing.

Dong-Hae Chi received the Bachelor of Science degree in 1982 from Kyung Hee University in Seoul, and the Master and Ph.D. degree of Engineering from Iowa State University in Ames, Iowa in 1986 and 1989, respectively. In 1989, he joined ETRI as a Senior Member of Research Staff. He is presently the head of Programming Environments Team at Computer-Software Technology Division. He has been joined in the development of national host computer system (so called TICOM) project as a Project Manager over 10 years. His current research interests are the streaming media software, such as a Multimedia Filesystem, an Integrated Multimedia Framework (MPEG 4 part 1, 6), and an Application Server Framework, etc.

Dong-Kyoo Lee received the B.S. and M.S. degrees in computer engineering from the Kyungpook National University, Taegu, Korea in 1996 and 1998, respectively. Currently, he is a Ph.D. student in computer engineering, Kyungpook National University. His interest is design and analysis of parallel algorithms.

Kwan-Woo Ryu received the B.S. degree in electrical engineering from Kyungpook National University, Daegu, Korea in 1980, and the M.S. degree in computer science from the Korea Advanced Institute of Science and Technology, Seoul in 1982. He received his Ph.D. degree from the Department of Computer Science at the University of Maryland, College Park, in 1990. Currently, he is an associate professor in the Department of Computer Engineering, Kyungpook National University, Korea. His main research interests are design and analysis of parallel algorithms for combinatorial problems, and computer graphics.