

Batcher's Odd-Even Merge sort

Merge (A, B)

1. $C = \text{Merge}(\text{even}(A), \text{odd}(B))$

$D = \text{Merge}(\text{odd}(A), \text{even}(B))$

d.h. $C = c_0, c_1, \dots, c_{M-1}$ und
 $D = d_0, d_1, \dots, d_{M-1}$ sind sortiert

2. Bilde ineinandergeschachtelte Liste

$$L' = c_0, d_0, c_1, d_1, \dots, c_{M-1}, d_{M-1}$$

3. Sortiere alle Paare c_i, d_i aufsteigend!

$$l_{2i} = \min(c_i, d_i)$$

$$l_{2i+1} = \max(c_i, d_i)$$

$$L = l_0, l_1, \dots, l_{2M-1}$$

Lemma: $L = \text{Merge}(A, B)$ ist sortiert

Odd-Even Mergesort

$A = a_0, a_1, \dots, a_{N-1}$ unsortierte Eingabe

Das Sortierverfahren baut mit Hilfe des Mischalgorithmus nacheinander immer längere sortierte Teilfolgen A_i auf.

Anfangs: $A_i^{(\log N)} := a_i$

Sort (A)

for $j = \log N - 1$ to 0 do

for all $i = 0$ to $2^j - 1$ parallel

$A_i^{(j)} := \text{Merge} (A_{2i}^{(j+1)}, A_{2i+1}^{(j+1)})$

Implementation auf dem Butterfly

$$A = a_0, \dots, a_{M/2-1}$$

$$B = b_0, \dots, b_{M/2-1} \quad \text{sortierte Teilfolgen,}$$

die auf einem $(\log M)$ -dimensionalen Butterfly gemischt werden sollen.

(Betrachtet wird dabei ein gespiegelter Butterfly.)

$\text{bin}(i) = i_k \dots i_0$ Binärdarstellung von $i \in \mathbb{N}$
mit aufgefüllten Nullen

$$\mathbb{1}(\text{bin}(i)) = \mathbb{1}i_k \dots i_0$$

Eingabe: für $i = 0, \dots, M/2 - 1$

a_i in Knoten $\langle \mathbb{0} \mathbb{1} \text{bin}(i), \log M \rangle$ oben

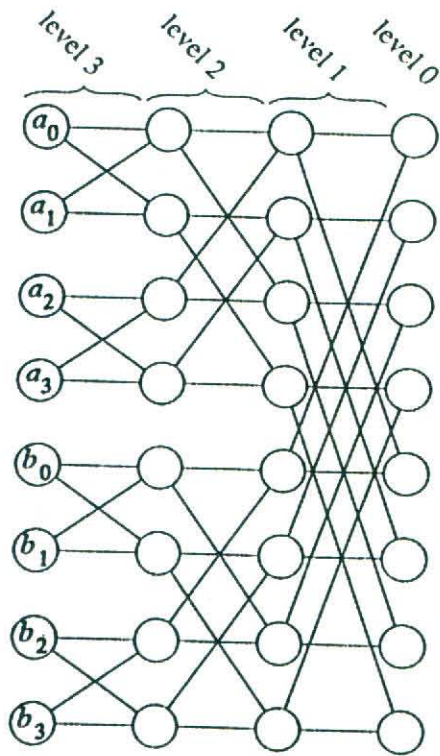
b_i in Knoten $\langle \mathbb{1} \mathbb{1} \text{bin}(i), \log M \rangle$ unten

Dann im 1.sten Schritt:

a_i auf Zeilenkanten nach $\langle \mathbb{0} \mathbb{1} \text{bin}(i), \log M - 1 \rangle$

b_i auf Kreuzungskanten nach $\langle \mathbb{1} i_k \dots i_1 \bar{i}_0, \log M - 1 \rangle$

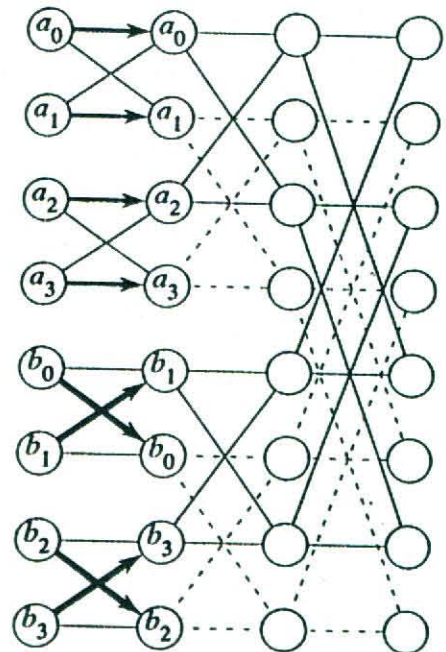
Level $\log M$



initial location of the data

(a)

Level $\log M - 1$



the first step of the merge

(b)

Figure 3-76 Implementation of the Odd-Even Merge algorithm on a butterfly. Initially, the two lists of $M/2$ items to be merged are entered into the leaves of level $\log M$ as shown in (a). During the first step, the data is moved to level $\log M - 1$ as shown in (b). After the first step, we can use recursion to merge even(A) with odd(B) in the $(\log M - 1)$ -dimensional subbutterfly consisting of the even rows (denoted by solid edges) and to merge odd(A) with even(B) in the subbutterfly consisting of odd rows (denoted by dashed edges).

Ab Level $\log M - 1$ bilden

die geraden bzw ungeraden Zeilen
jeweils einen Butterfly der Dim. $\log M - 1$

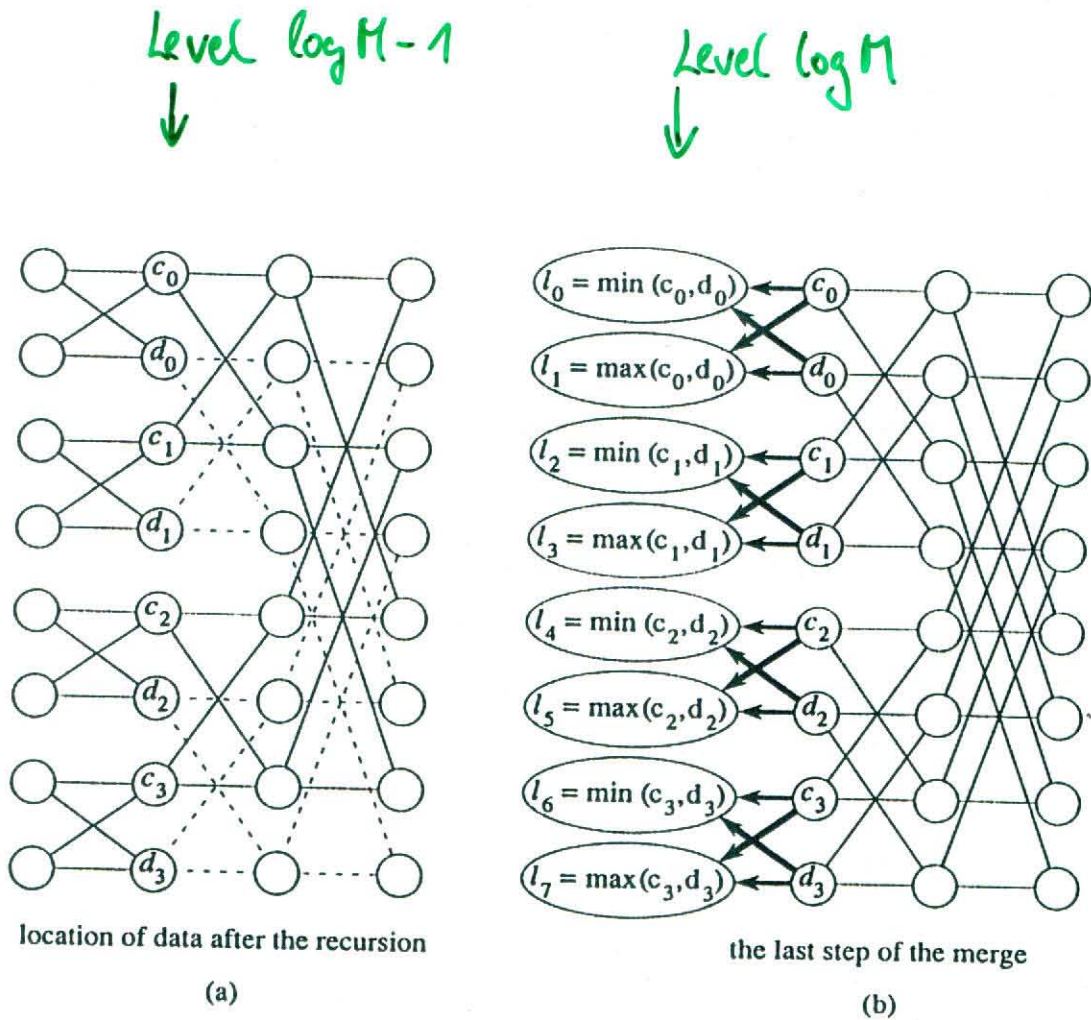


Figure 3-77 Implementation of the Odd-Even Merge algorithm on a butterfly, continued. The location of the data after recursion is shown in (a). During the recursion, we used the $(\log M - 1)$ -dimensional subbutterfly on the even rows (solid edges) to recursively merge even(A) with odd(B) to form C , and the $(\log M - 1)$ -dimensional subbutterfly on the odd rows (dashed edges) to recursively merge odd(A) with even(B) to form D . The merge of A and B is completed by comparing c_i with d_i for each i and switching pairs that are out of order, as shown in (b).

Beispiel: Bitonisches Sortieren

8, 10, 15, 12, 9, 4, 2, 7

min a_i

8

4

2

7

b_i max

9

10

15

12

rekursiv sortiert



2 4 7 8



9 10 12 15

(sortiert)