



1

Merge Sort

計算機アルゴリズム特論：2017年度

只木進一

Merge Sortの基本的考え方

- 規模の小さな問題は容易に解ける
 - 長さ1のリストは、sort不要
- すでに解かれた部分問題から、解を得るのは容易な場合がある
 - sort済みの二つのリストから、sortされた一つのリストを作るのは容易

Merge Sort

リストの分離

3	8	5	2	7	6	1	4
---	---	---	---	---	---	---	---

3	8	5	2
---	---	---	---

7	6	1	4
---	---	---	---

3	8
5	2

7	6
---	---

1	4
---	---

3	8	5	2
---	---	---	---

7	6	1	4
---	---	---	---

- 簡単のため、リスト長を $n = 2^m$ とする。
- 分割の回数は $\log_2 n = m$
- 各階層でのリストへの追加は n 回。
- 分割時の工数： $n \log_2 n$

Merge Sort

リストの結合

3	8	5	2
---	---	---	---

7	6	1	4
---	---	---	---

3	8
---	---

2	5
---	---

6	7
---	---

1	4
---	---

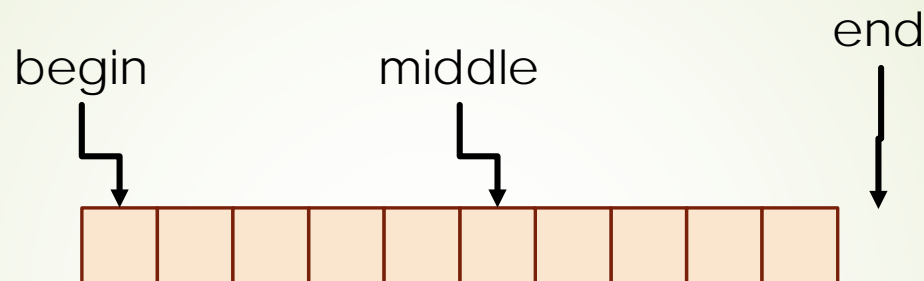
2	3	5	8
---	---	---	---

1	4	6	7
---	---	---	---

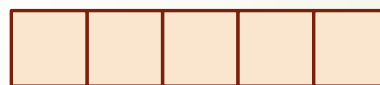
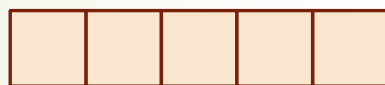
1	2	3	4	5	6	7	8
---	---	---	---	---	---	---	---

- 結合の回数は $\log_2 n = m$
- 各階層でのリストへの追加は n 回
- 結合時の工数 : $n \log_2 n$

基本的考え方



分割



それぞれ整列

結合した配列を作成



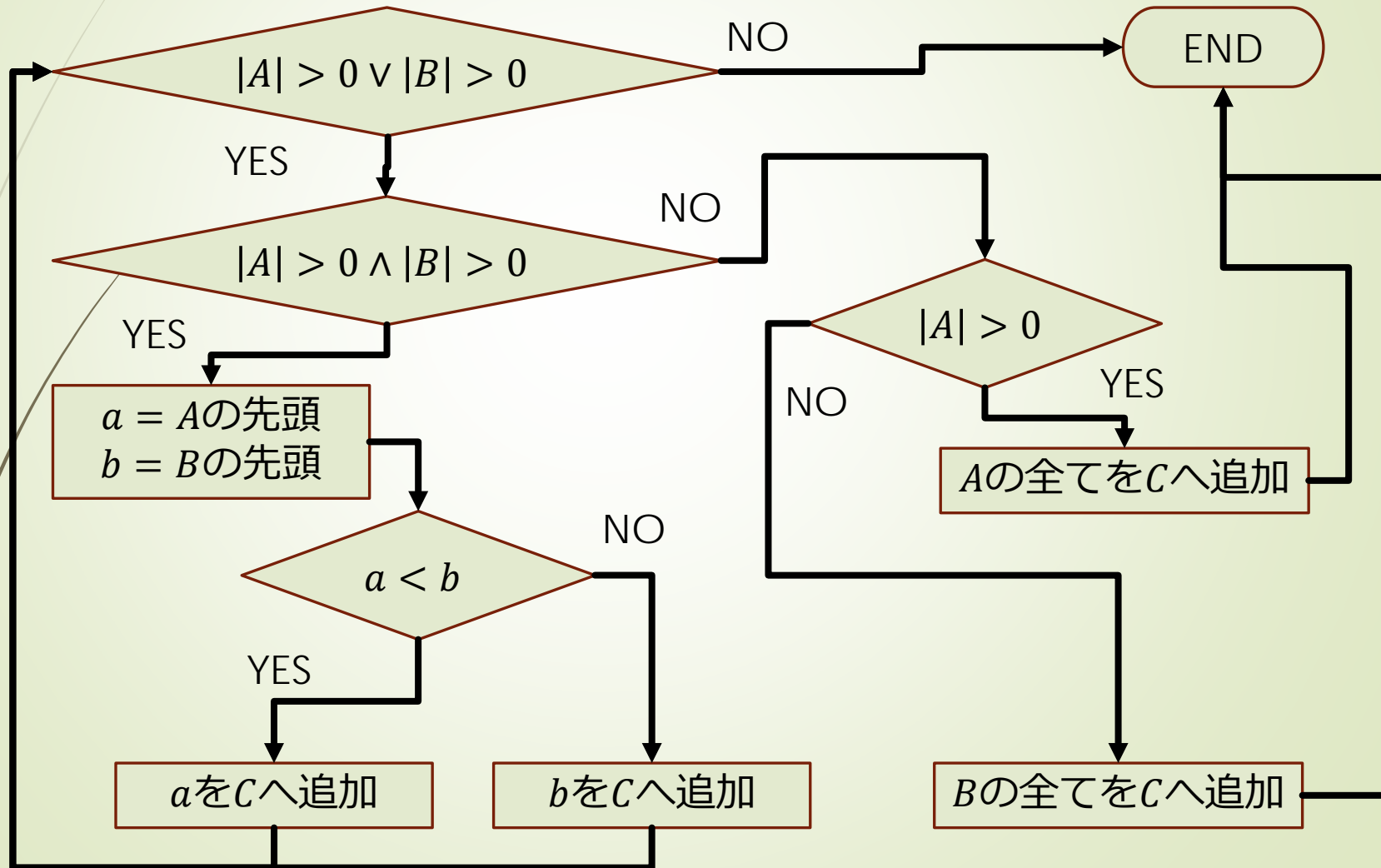
元の配列への上書き

Merge Sortアルゴリズム 再帰での記述

```
protected void sortSub(int begin, int end) {  
    if (end <= begin) {  
        throw new IllegalArgumentException("illegal range");  
    }  
    if (end == begin + 1) {  
        return;  
    }  
    int middle = (end + begin) / 2;  
    sortSub(begin, middle);  
    sortSub(middle, end);  
    T tmpList[] = mergeArray(begin, middle, end);  
    for (int i = 0; i < tmpList.length; i++) {  
        data[begin + i] = tmpList[i];  
    }  
}
```


二つの整列済みリストの結合

リストAとリストBを結合してCへ



要素数 N に対する比較の回数 $S(N)$

