# van Emde Boas Trees

光吉　健汰

北海道大学工学部 情報エレクトロニクス学科 情報理工学コース 4 年
情報知識ネットワーク研究室

June 11, 2019

# Contents

# Contents

こんにちは

# Contents

# van Emde Boas tree とは (1/2)

van Emde Boas tree (以下 vEB 木) は動的集合を扱うデータ構造

**動的集合**

動的集合とは, 集合に対して後述の各種操作が行えるようなデータ構造

- 今回扱う要素は非負整数とする
- vEB 木が保持しうる要素の集合を全体集合 $U$ とし, その大きさを $u$ とする
- vEB 木が現在保持している集合を $V$ とし, その大きさを $n$ とする

# van Emde Boas tree とは (2/2)

van Emde Boas tree (以下 vEB 木) は動的集合を扱うデータ構造

## 操作

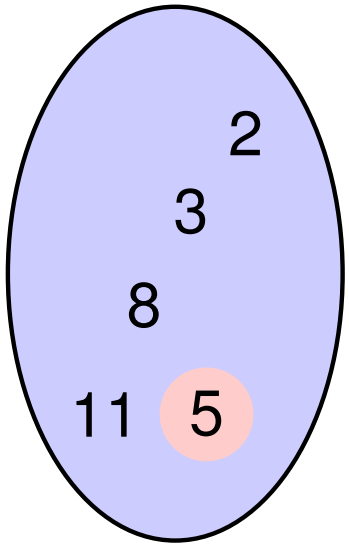
| | |
|---|---|
| $\textsc{Member}(V, x)$ | $V$ に $x$ が存在するかを返す |
| $\textsc{Min}(V)$ | $V$ の要素の最小値を返す |
| $\textsc{Max}(V)$ | $V$ の要素の最大値を返す |
| $\textsc{Successor}(V, x)$ | $V$ の $x$ より大きい最小の要素を返す |
| $\textsc{Predecessor}(V, x)$ | $V$ の $x$ より小さい最大の要素を返す |
| $\textsc{Insert}(V, x)$ | $V$ に $x$ を挿入する |
| $\textsc{Delete}(V, x)$ | $V$ から $x$ を削除する |

これらの操作が最悪時間計算量 $O(\log \log u)$ で実行可能

# 操作 MEMBER$(V, x)$

- MEMBER$(V, x)$ は, $x$ が集合に存在するかを真偽値で返す.

Figure: $V = \{2, 3, 5, 8, 11\}$




- MEMBER$(V, 5)$ = TRUE
- MEMBER$(V, 6)$ = FALSE

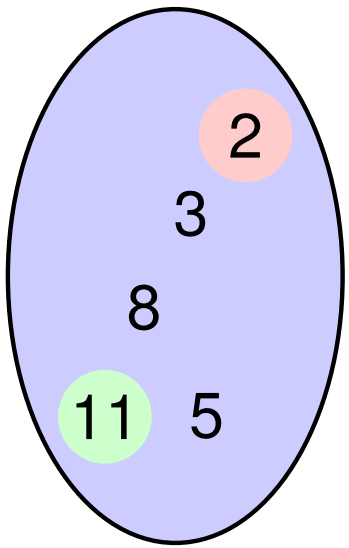
## 各関数の引数

関数に渡す引数は要素として取りうる値のみとする

# 操作 Min($V$), Max($V$)

- Min($V$) は, 集合の要素の最小値を返す.
- Max($V$) は, 集合の要素の最大値を返す.

Figure: $V = \{2, 3, 5, 8, 11\}$

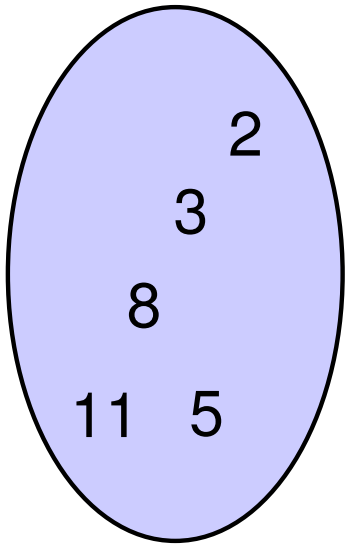



- Min($V$) $= 2$
- Max($V$) $= 11$

# 操作 Successor$(V, x)$, Predecessor$(V, x)$

- Successor$(V, x)$ は, 集合の要素から $x$ より大きい最小の値を返す.
- Predecessor$(V, x)$ は, 集合の要素から $x$ より小さい最大の値を返す.

Figure: $V = \{2, 3, 5, 8, 11\}$




- Successor$(V, 2) = 3$
- Predecessor$(V, 7) = 5$

# 操作 INSERT($V, x$)

- INSERT($V, x$) は, 集合に $x$ を挿入する.

Figure: $V = \{2, 3, 5, 8, 11\}$

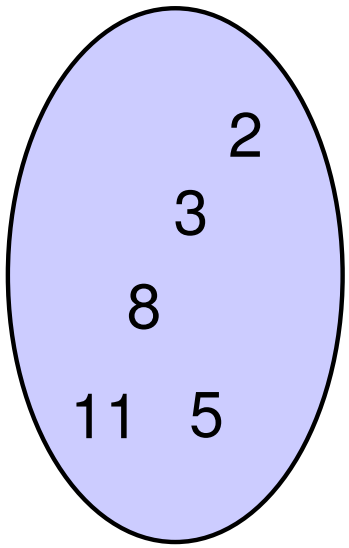
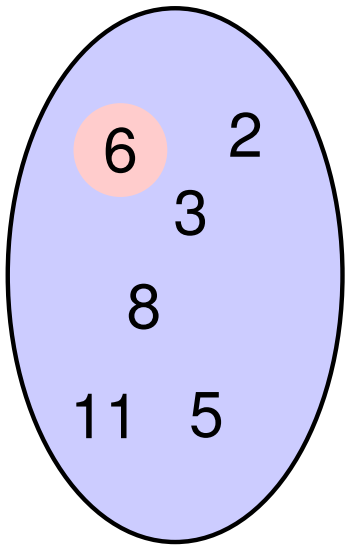



Figure: $V = \{2, 3, 5, \mathbf{6}, 8, 11\}$



# 操作 DELETE$(V, x)$

- DELETE$(V, x)$ は, 集合 $V$ から $x$ を削除する.

Figure: $V = \{2, 3, 5, \mathbf{6}, 8, 11\}$

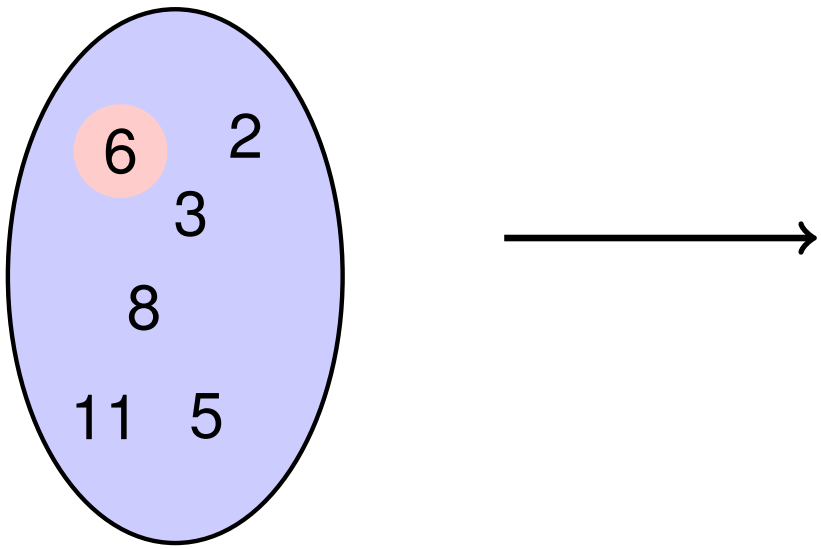
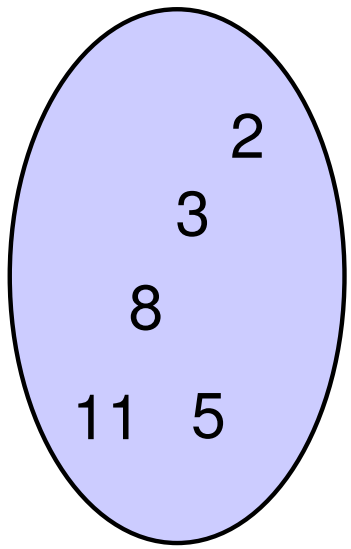



Figure: $V = \{2, 3, 5, 8, 11\}$



# Contents

# 直接アドレス法 (1/2)

空間計算量 $O(u)$ で動的集合を保持する手段として, 直接アドレス法を考える.

## 直接アドレス法

要素の値を配列の添字として利用し, データを保持するテクニック

今回考えているデータ構造では付属データは持たないので,
各配列の要素には要素を保持しているかを bit で格納する.

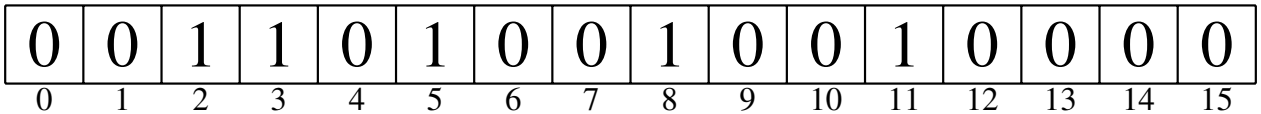



Figure: $V = \{2, 3, 5, 8, 11\}$

# 直接アドレス法 (2/2)

空間計算量 $O(u)$ で動的集合を保持する手段として, 直接アドレス法を考える.

- MEMBER$(V, x)$, INSERT$(V, x)$, DELETE$(V, x)$ の処理は, 配列のランダムアクセスが定数時間であるので, 時間計算量 $O(1)$.
- SUCCESSOR$(V, x)$ の処理は, $x$ の次の値から bit が立っている要素まで
  最悪 $\Theta(u)$ 回探索する必要があるので, 時間計算量 $\Theta(u)$.
  - PREDECESSOR$(V, x)$, MIN$(V)$, MAX$(V)$ も同様の処理を行うため, 時間計算量 $\Theta(u)$.

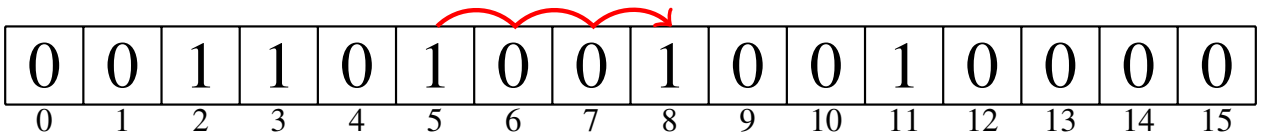



Figure: $V = \{2, 3, 5, 8, 11\}$

# 二分木構造 (1/2)

$Successor(V, x)$ で必要な探索回数を小さくするために,
配列の各要素を葉とした二分木構造を考える.

## 二分木

二分木とは, 葉ではない頂点の子が常に 2 個であるような木

- 木 閉路を持たない, 連結なグラフ
- 根 木の頂点のうちの 1 つに定義する
- 子 ある頂点について, 隣接する頂点のうち根から遠いもの
- 葉 木の頂点のうち, 子を持たないもの

# 二分木構造 (2/2)

$Successor(V, x)$ で必要な探索回数を小さくするために,
配列の各要素を葉とした二分木構造を考える.
葉ではない各頂点には, 子の値の論理和を格納する.

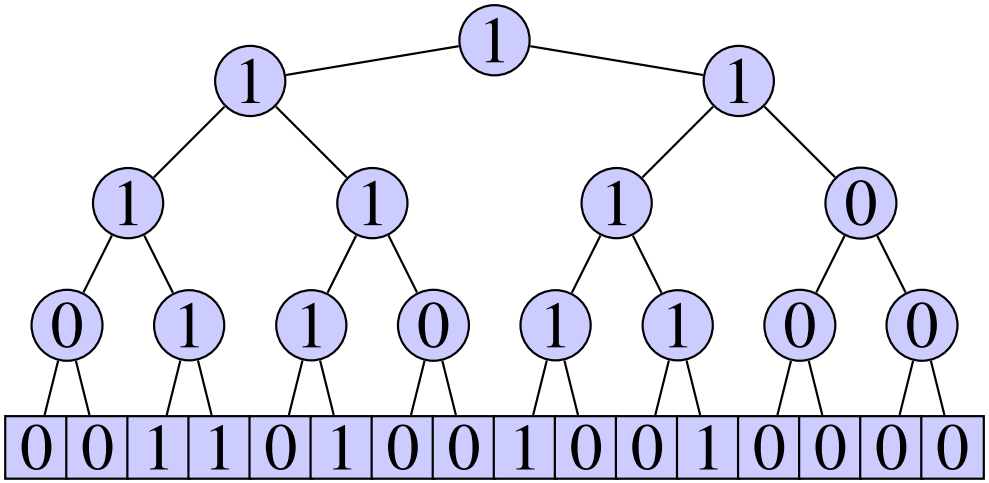



Figure: $V = \{2, 3, 5, 8, 11\}$

# 二分木構造 操作 MIN$(V, x)$

- MIN$(V, x)$ は, 根から左の子が 1 であれば左の子へ, そうでなければ右の子へ降りる操作を再帰的に行う.

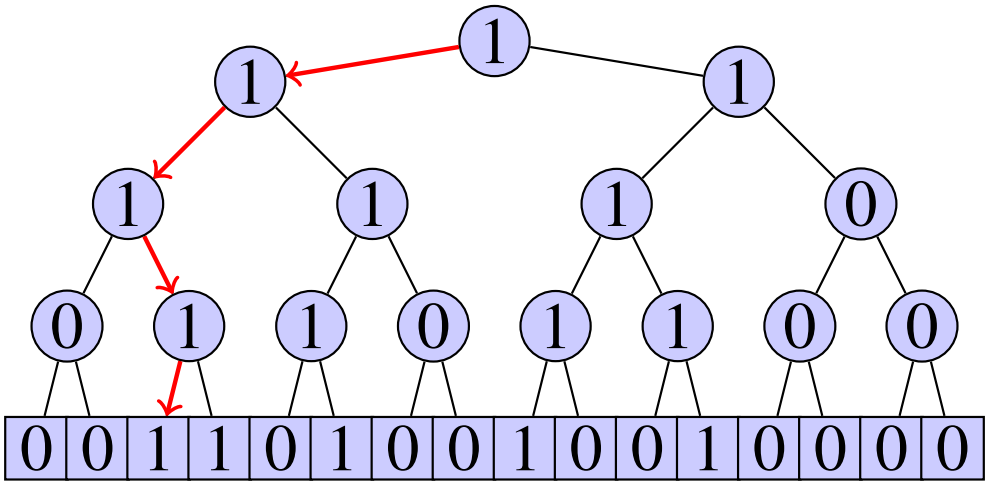



Figure: $V = \{2, 3, 5, 8, 11\}$