大規模知識処理特論

二分決定グラフの利用 (3)

北海道大学 情報科学研究院

堀山 貴史

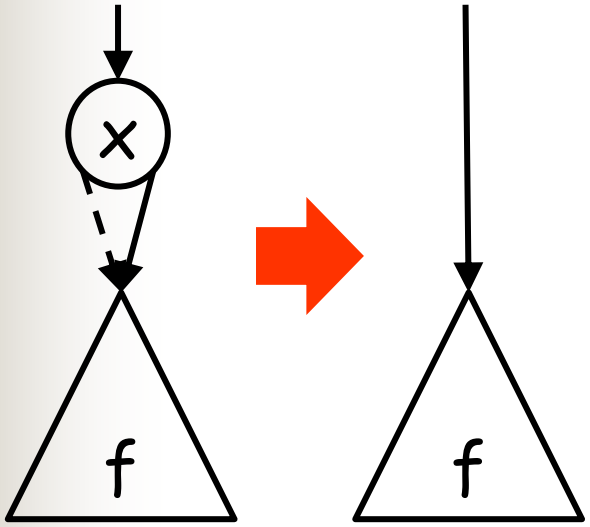
ZDD
(Zero-suppressed BDD)

- ZDD: BDD のバリエーション
- フロンティア法
  (BDD/ZDD のトップダウン構築法)

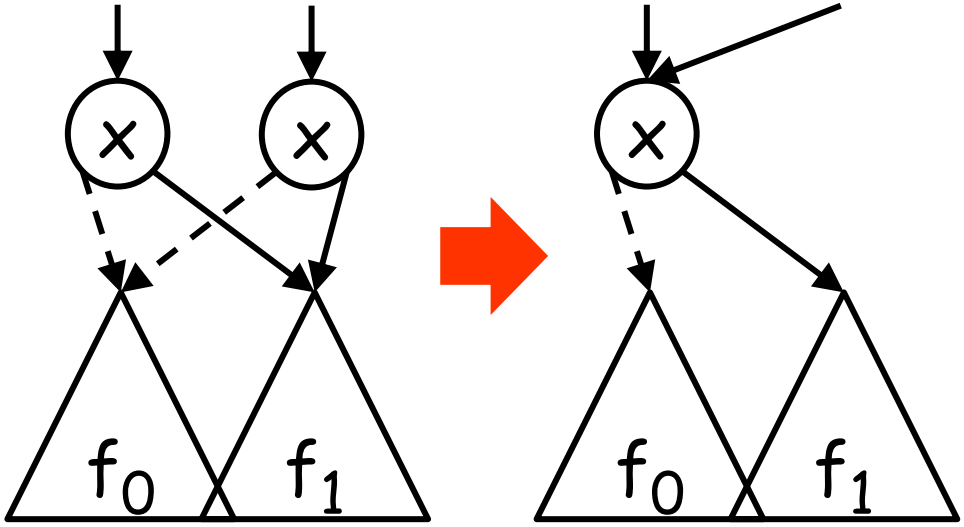
BDD の復習

- **変数順序**： 全順序関係にしたがって、変数が出現
- ２つの**簡約化規則**を適用
  - 既約化：冗長な節点，等価な節点がなくなるまで

冗長な節点の削除



等価な節点の共有

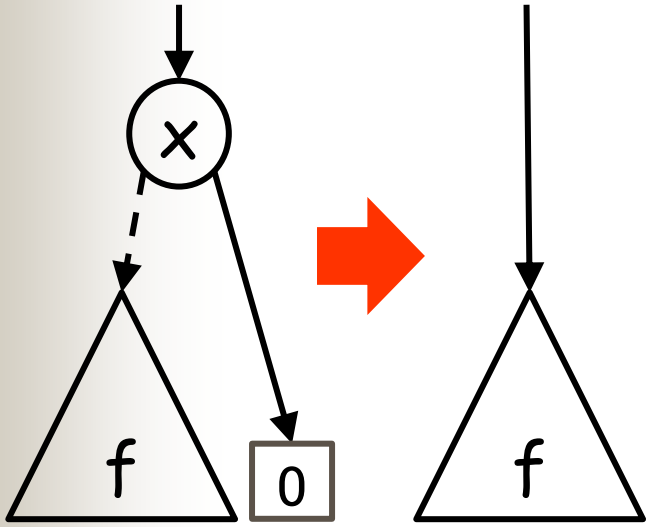


[Minato 1993]

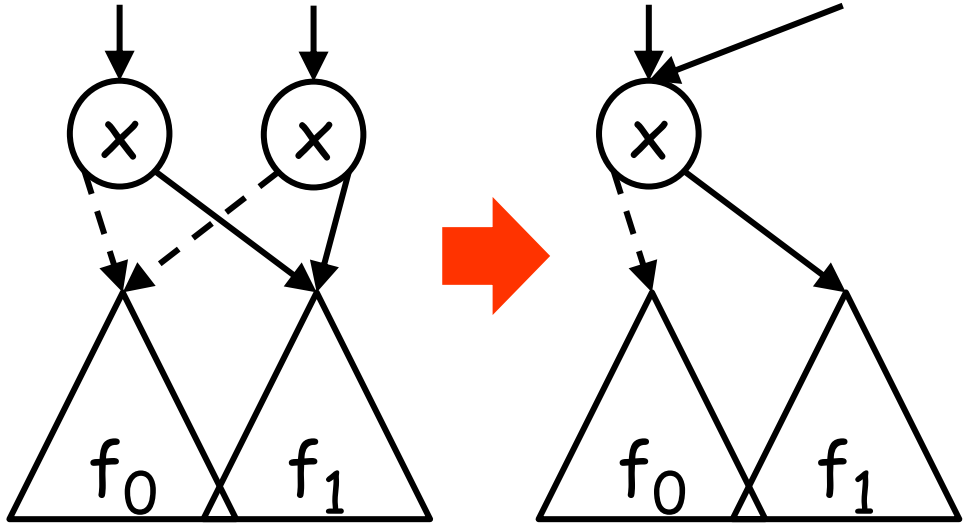
ZDD (Zero-Suppressed BDD)

- **変数順序**： 全順序関係にしたがって、変数が出現
- 2つの**簡約化規則**を適用
  - 既約化：冗長な節点，等価な節点がなくなるまで

冗長な節点の削除

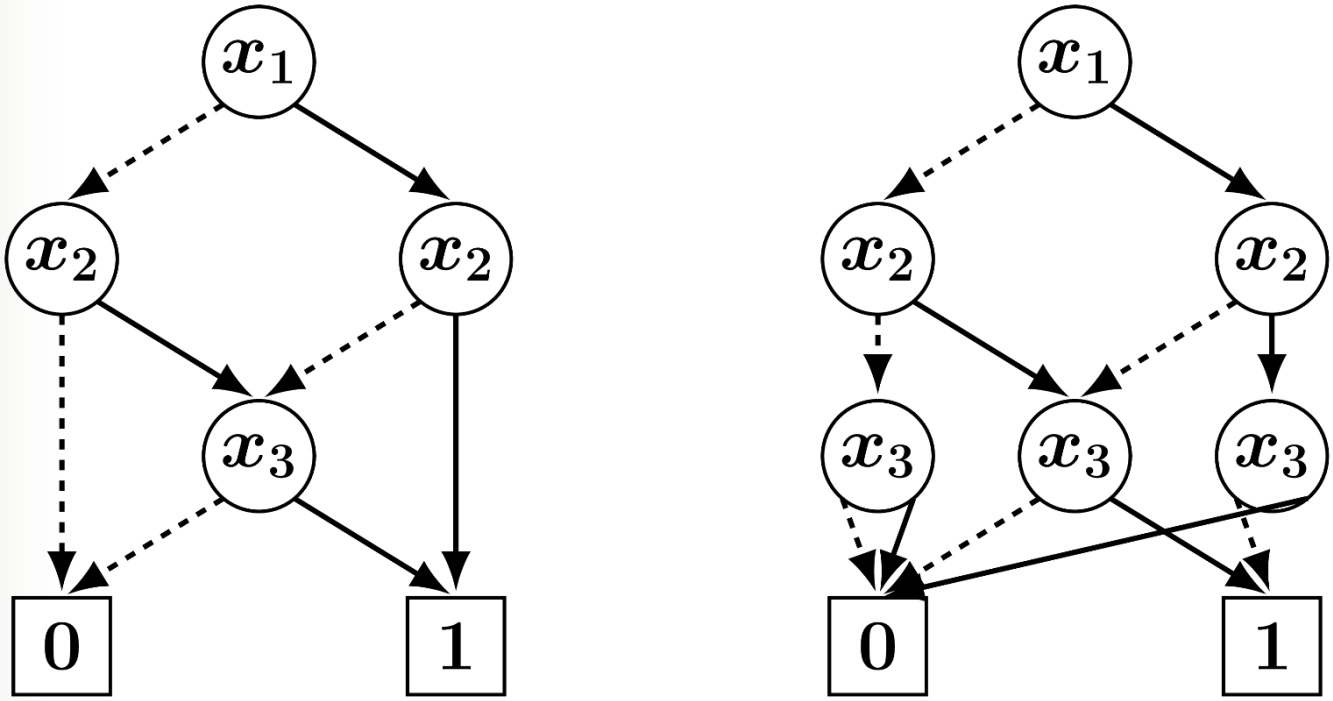


等価な節点の共有



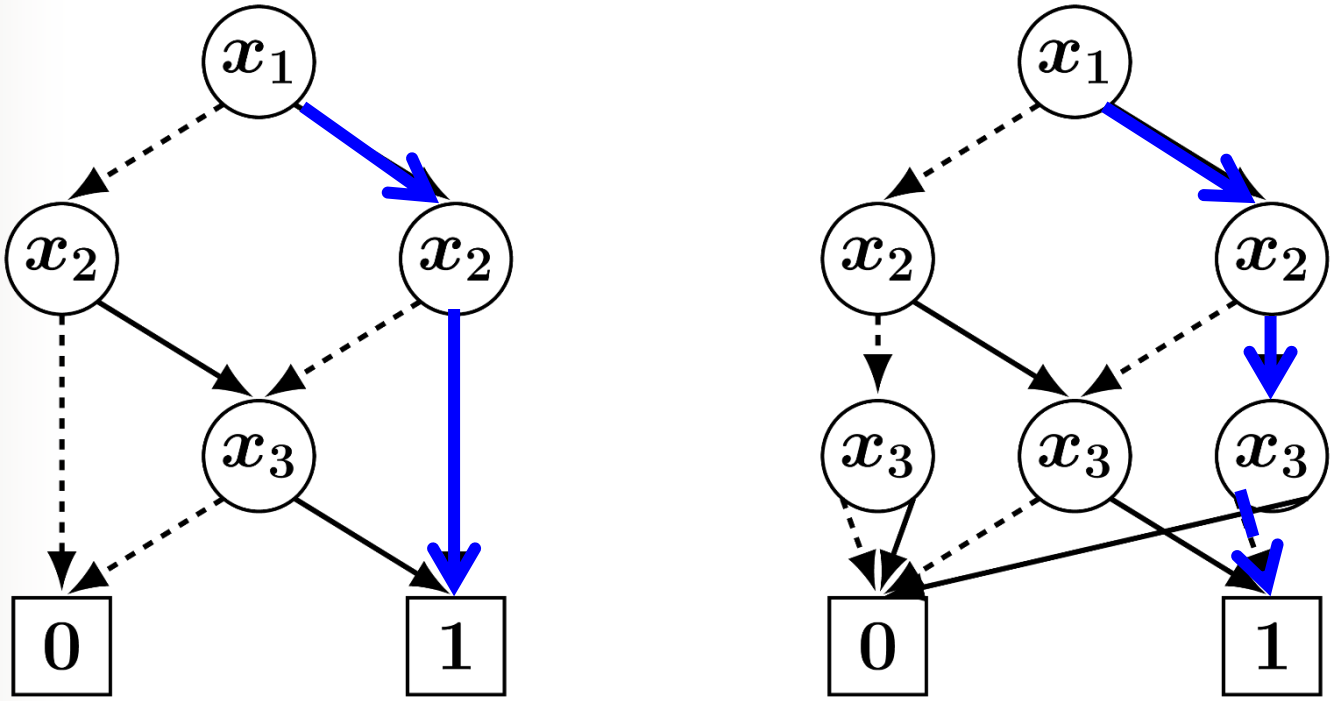
ZDD

例) $\{ \{x_1, x_2\}, \{x_1, x_3\}, \{x_2, x_3\} \}$



ZDD が表す集合族を求める (方法1)

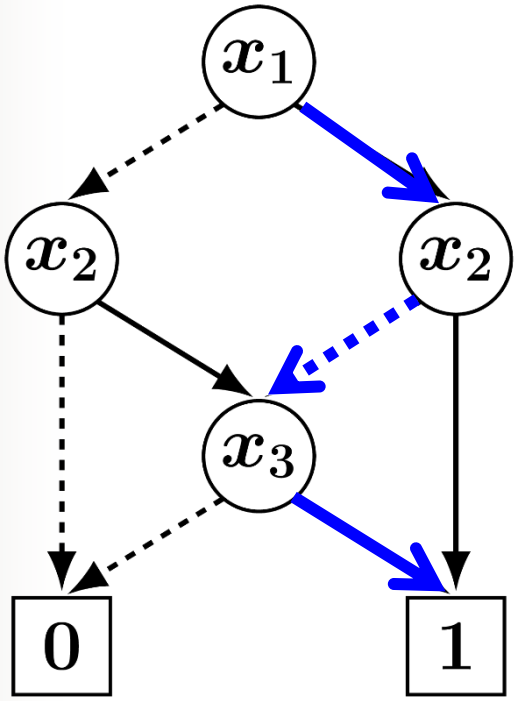
例)　{ $\{x_1, x_2\}$, $\{x_1, x_3\}$, $\{x_2, x_3\}$ }



- 1-path が、それぞれ集合に対応

ZDD が表す集合族を求める (方法1)

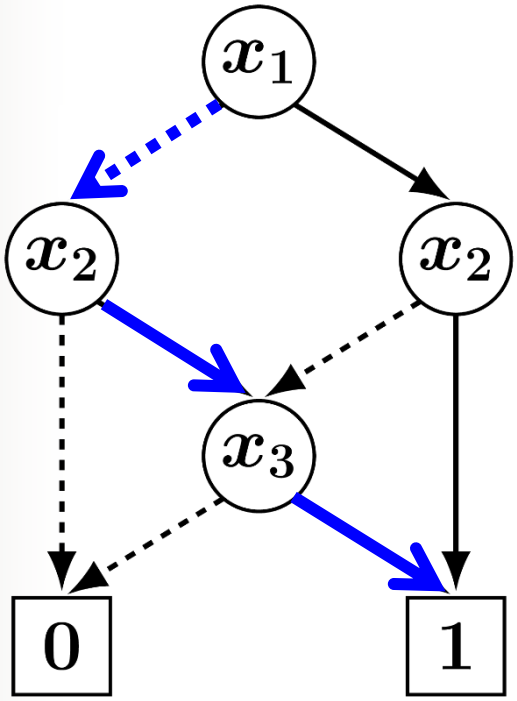
例) $\{ \{x_1, x_2\}, \{x_1, x_3\}, \{x_2, x_3\} \}$



- 1-path が、それぞれ集合に対応

ZDD が表す集合族を求める (方法1)

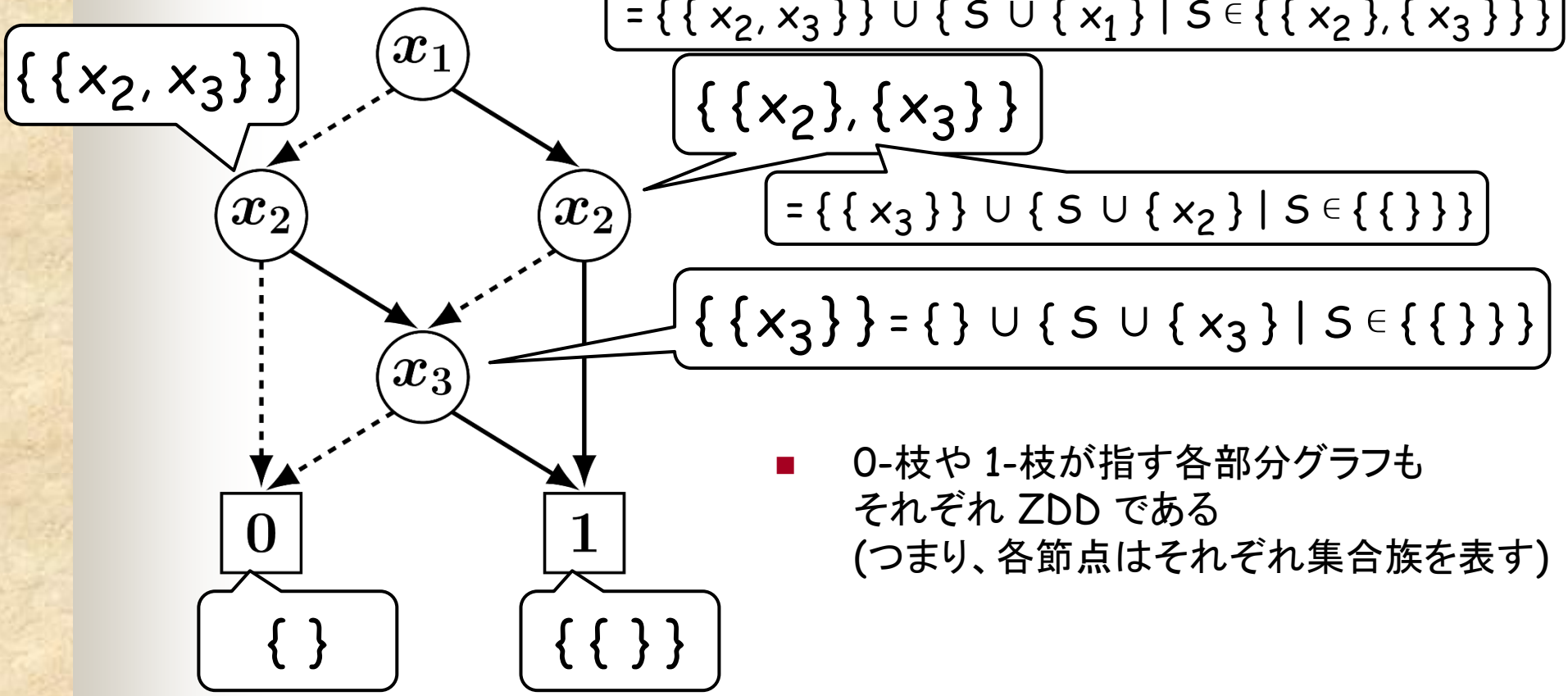
例) $\{ \{x_1, x_2\}, \{x_1, x_3\}, \{x_2, x_3\} \}$



- 1-path が、それぞれ集合に対応

ZDD が表す集合族を求める (方法2)

例) $\{ \{x_1, x_2\}, \{x_1, x_3\}, \{x_2, x_3\} \}$

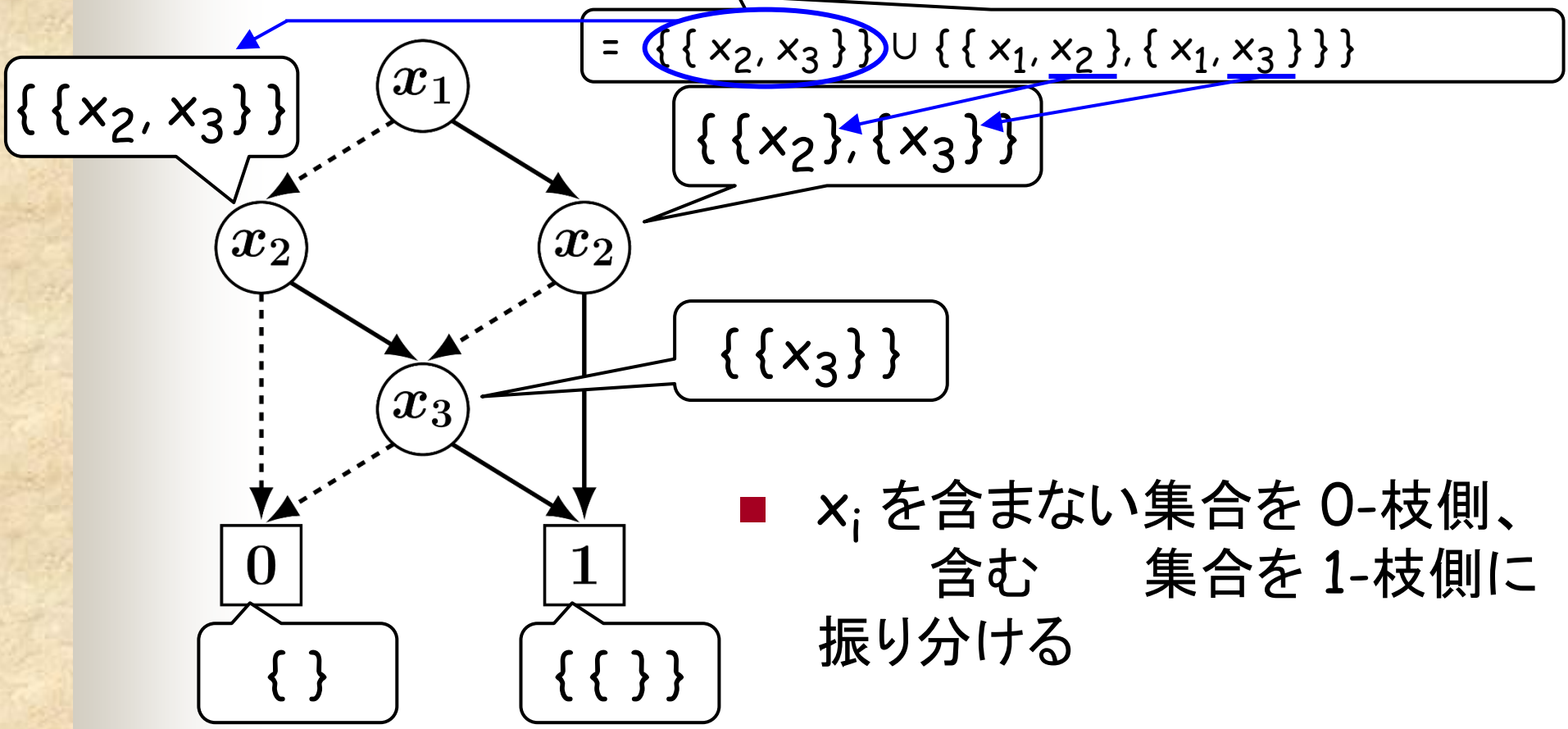


- 0-枝や 1-枝が指す各部分グラフも
  それぞれ ZDD である
  (つまり、各節点はそれぞれ集合族を表す)

- $F = F_0 \cup \{ S \cup \{x_i\} \mid S \in F_1 \}$

集合族から ZDD を求める

例)　$\{ \{x_1, x_2\}, \{x_1, x_3\}, \{x_2, x_3\} \}$

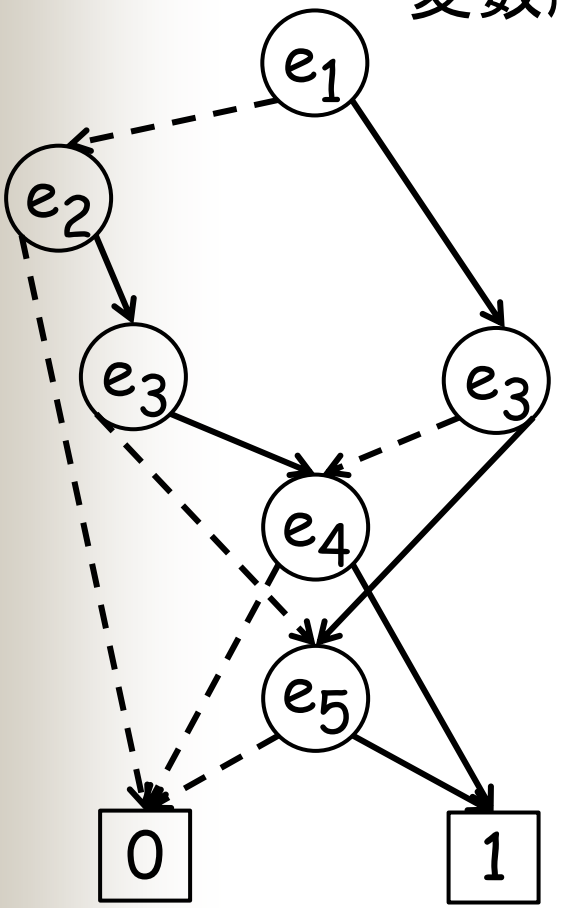


- x_i を含まない集合を 0-枝側、
  含む　集合を 1-枝側に
  振り分ける

- $F = F_0 \cup \{ S \cup \{x_i\} \mid S \in F_1 \}$

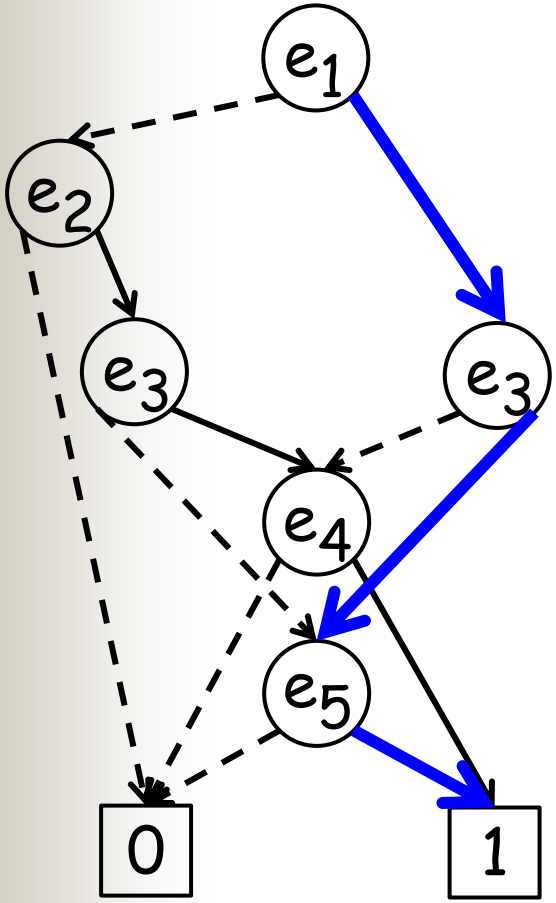
演習問題: ZDD

1. 以下の ZDD が表す集合族を求めなさい
2. その集合族を表す ZDD を、
   変数順序 e_2, e_1, e_3, e_4, e_5 で作りなさい



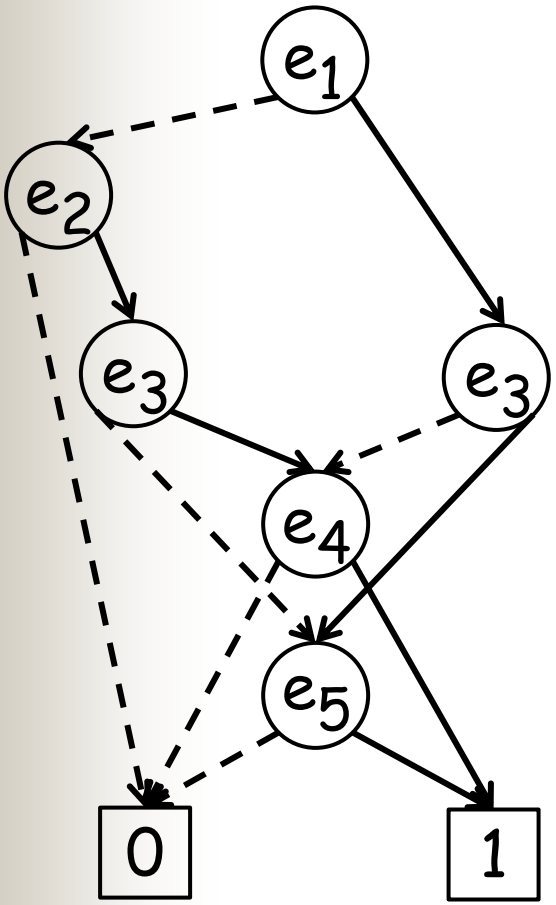
演習問題: ZDD

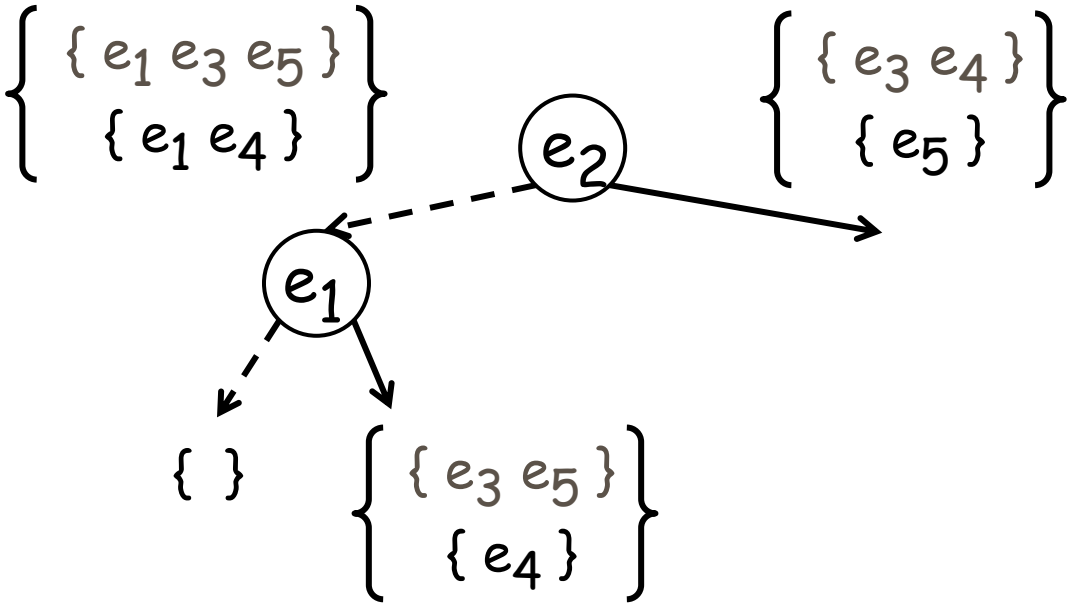
$$\left\{ \begin{array}{ll} \{ e_1\ e_3\ e_5 \} & \{ e_1\ e_4 \} \\ \{ e_2\ e_3\ e_4 \} & \{ e_2\ e_5 \} \end{array} \right\}$$



演習問題: ZDD

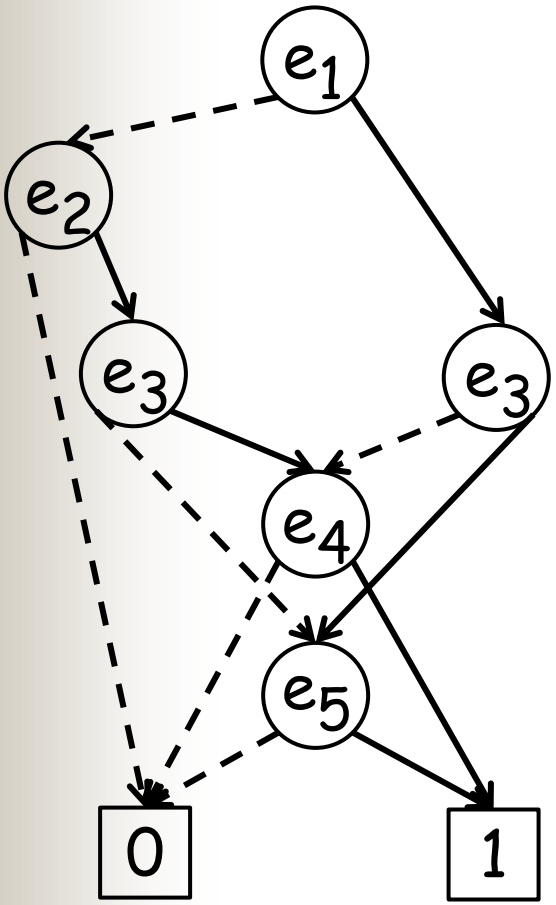
$$\left\{ \begin{array}{ll} \{ e_1\, e_3\, e_5 \} & \{ e_1\, e_4 \} \\ \{ e_2\, e_3\, e_4 \} & \{ e_2\, e_5 \} \end{array} \right\}$$





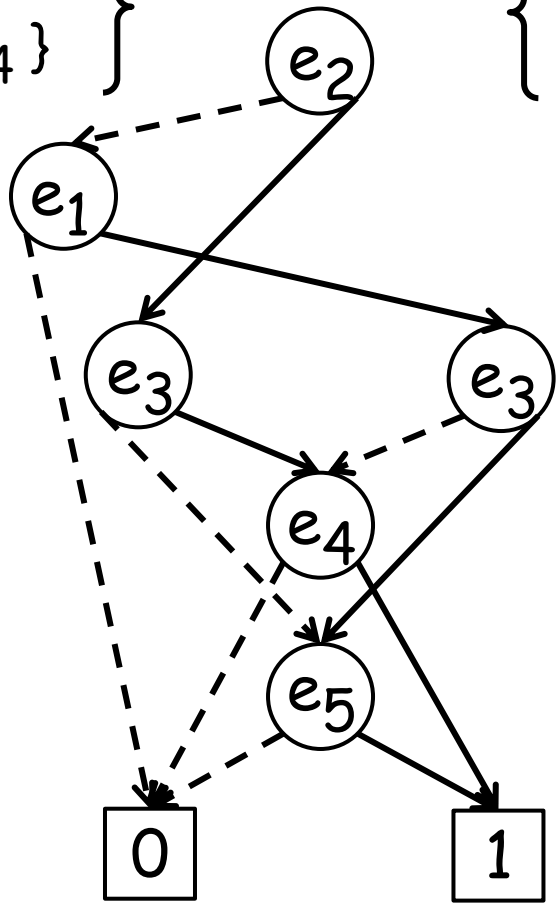
演習問題: ZDD

$$\left\{ \begin{array}{ll} \{ e_1\, e_3\, e_5 \} & \{ e_1\, e_4 \} \\ \{ e_2\, e_3\, e_4 \} & \{ e_2\, e_5 \} \end{array} \right\}$$



$$\left\{ \begin{array}{c} \{ e_1\, e_3\, e_5 \} \\ \{ e_1\, e_4 \} \end{array} \right\}$$

$$\left\{ \begin{array}{c} \{ e_3\, e_4 \} \\ \{ e_5 \} \end{array} \right\}$$



休 憩

- ここで、少し休憩しましょう。
- 深呼吸したり、肩の力を抜いてから、
  次のビデオに進んでください。

フロンティア法
(BDD/ZDD の top-down 構築)

BDD/ZDD の top-down 構築

- **Simpath** [Knuth 2008]
  - s-t パスの列挙
  - **動的計画法**の要領で、ZDD を top-down に構築
- **フロンティア法** [Kawahara et al. 2014, 2017]
  - Simpath 法の一般化
  - 全域木、森、サイクル、ハミルトンサイクル
    ハミルトンパス、などなど

※ 他分野でも、個別の問題で
同様のアプローチが試みられていた

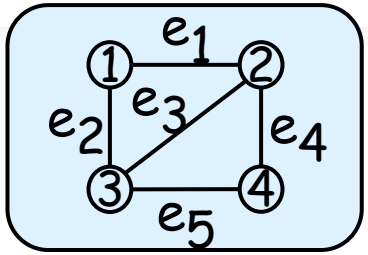
- 結び目理論の Jones 多項式
- 全域木

} [Sekine, Imai 1995]

- ネットワークの信頼性計算 [Hardy]

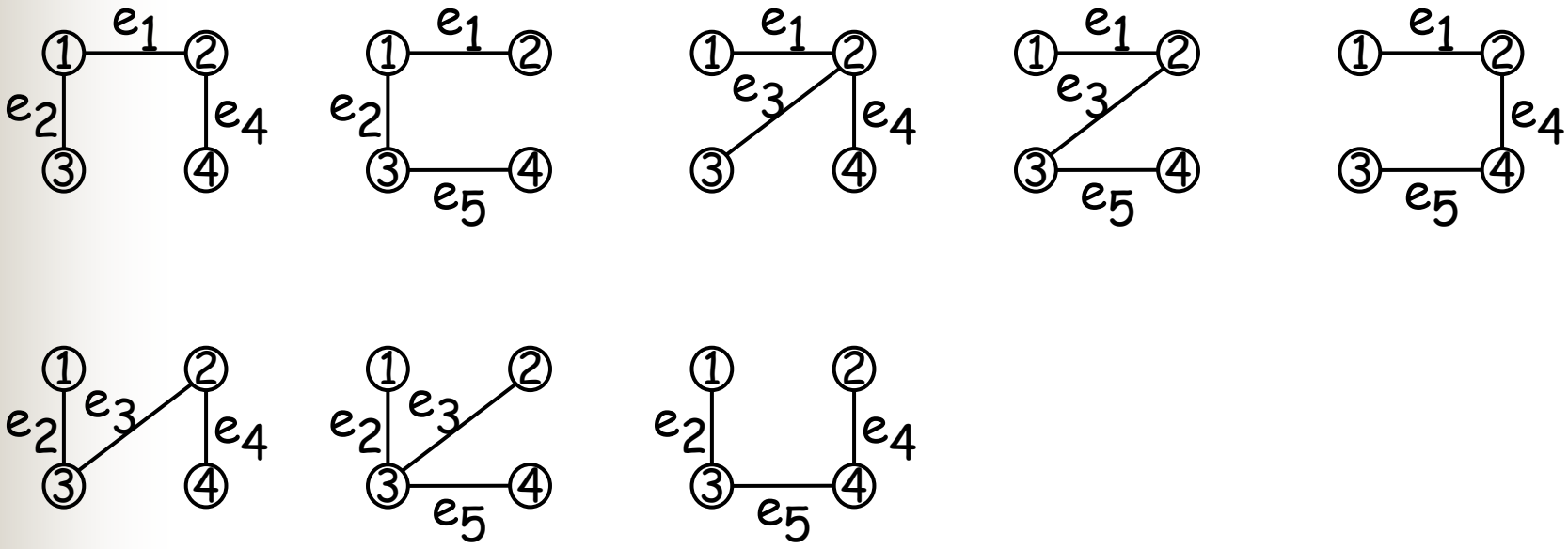
全域木の列挙

- サイクルを含まない
- 全頂点が連結



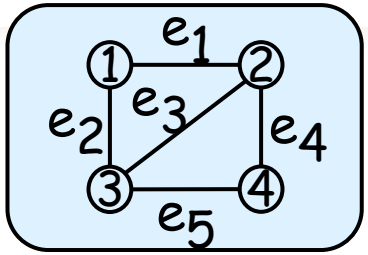
入力：グラフ

出力：入力のグラフの全域木すべて



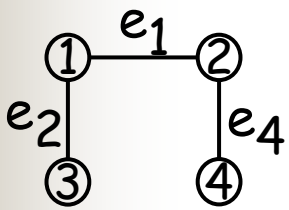
全域木の列挙

- サイクルを含まない
- 全頂点が連結

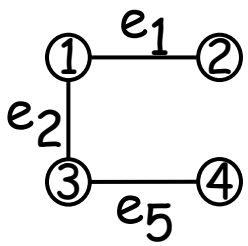


入力：グラフ

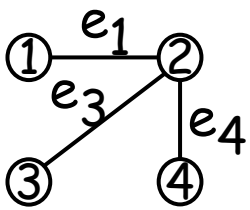
出力：入力のグラフの全域木すべて



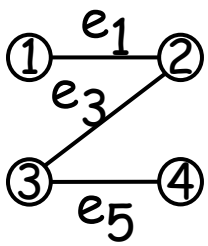
{ e_1 e_2 e_4 }



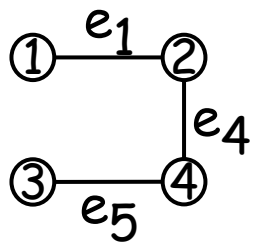
{ e_1 e_2 e_5 }



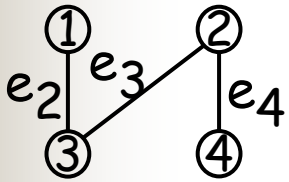
{ e_1 e_3 e_4 }



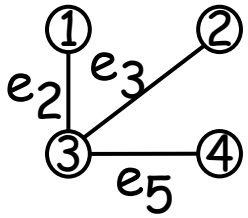
{ e_1 e_3 e_5 }



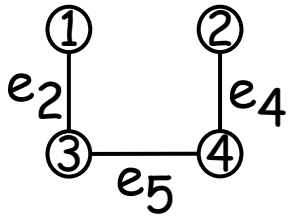
{ e_1 e_4 e_5 }



{ e_2 e_3 e_4 }



{ e_2 e_3 e_5 }

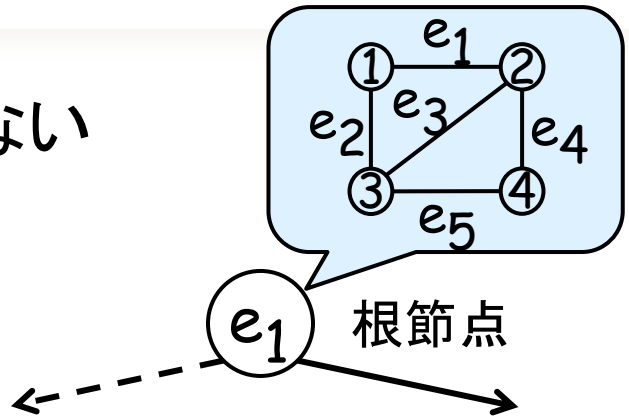


{ e_2 e_4 e_5 }

全域木は、辺の集合で表される

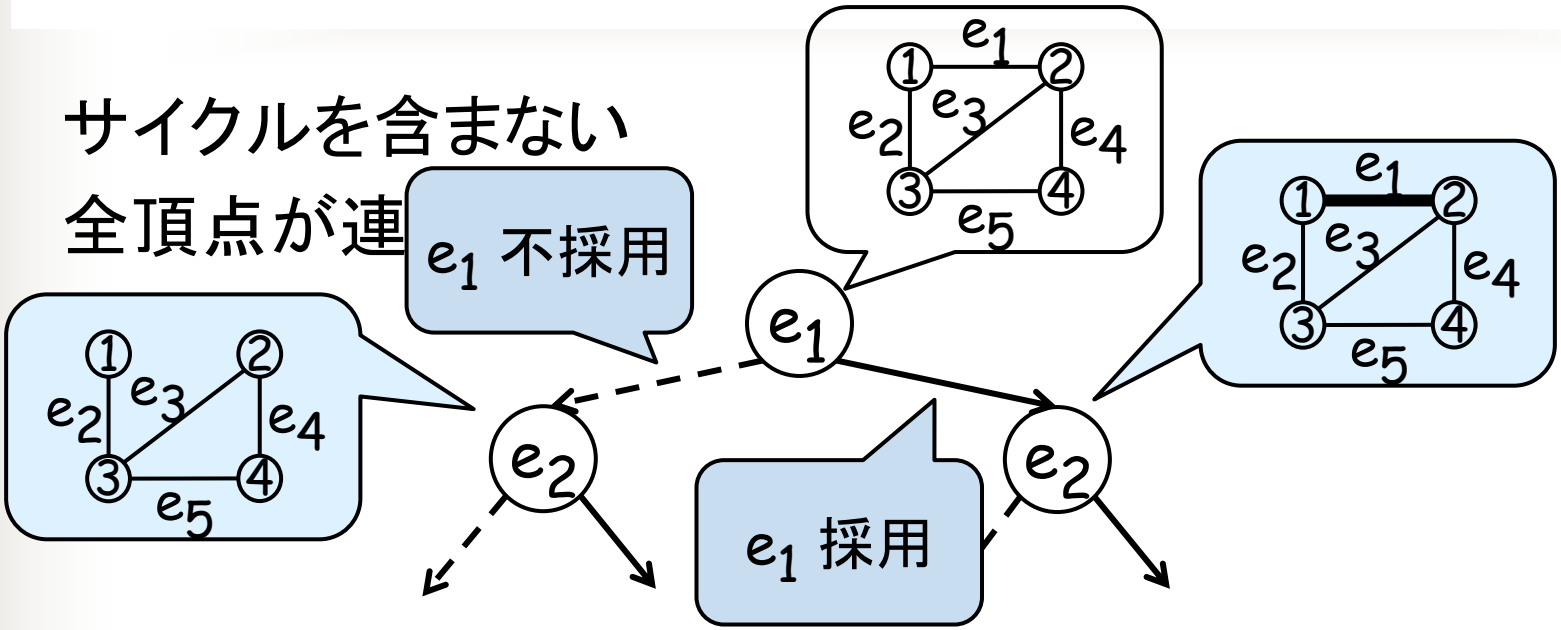
全域木の列挙

- サイクルを含まない
- 全頂点が連結



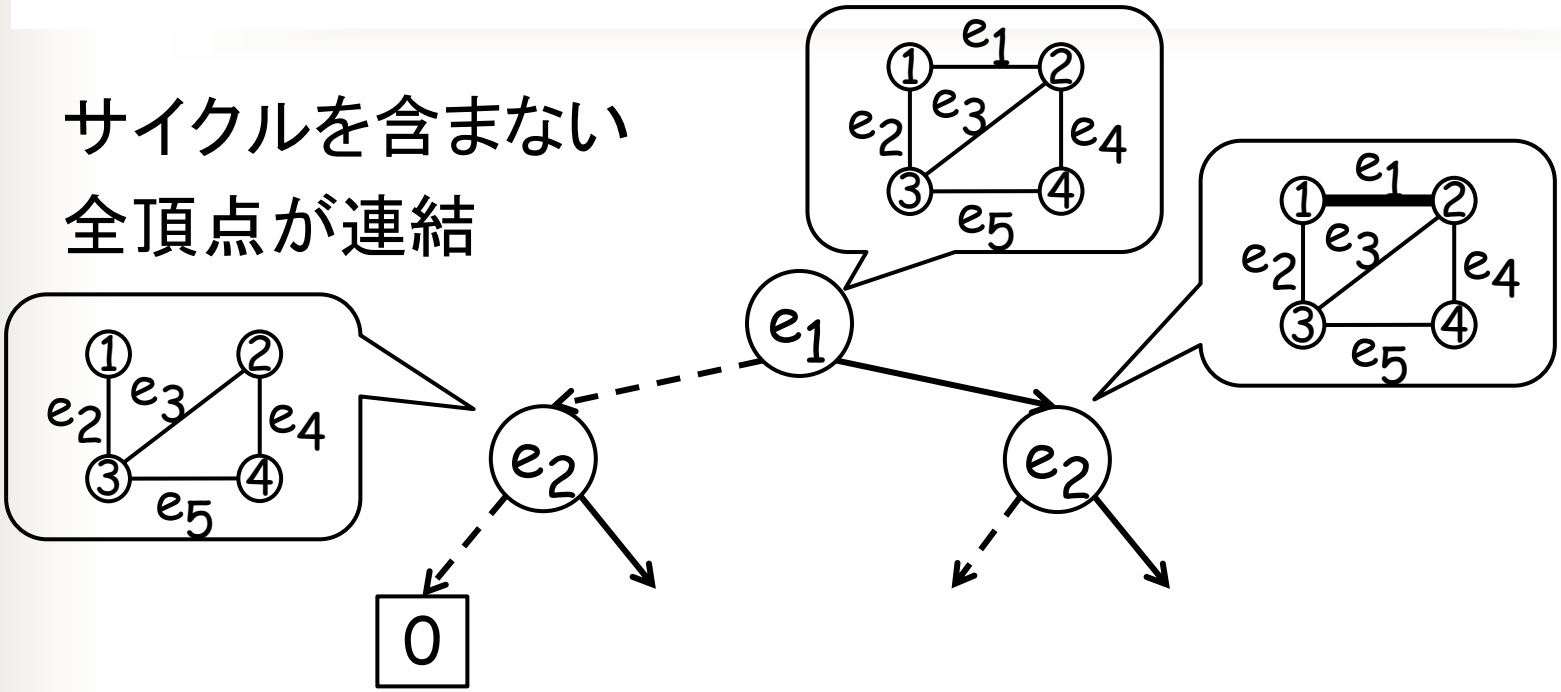
全域木の列挙

- サイクルを含まない
- 全頂点が連



全域木の列挙

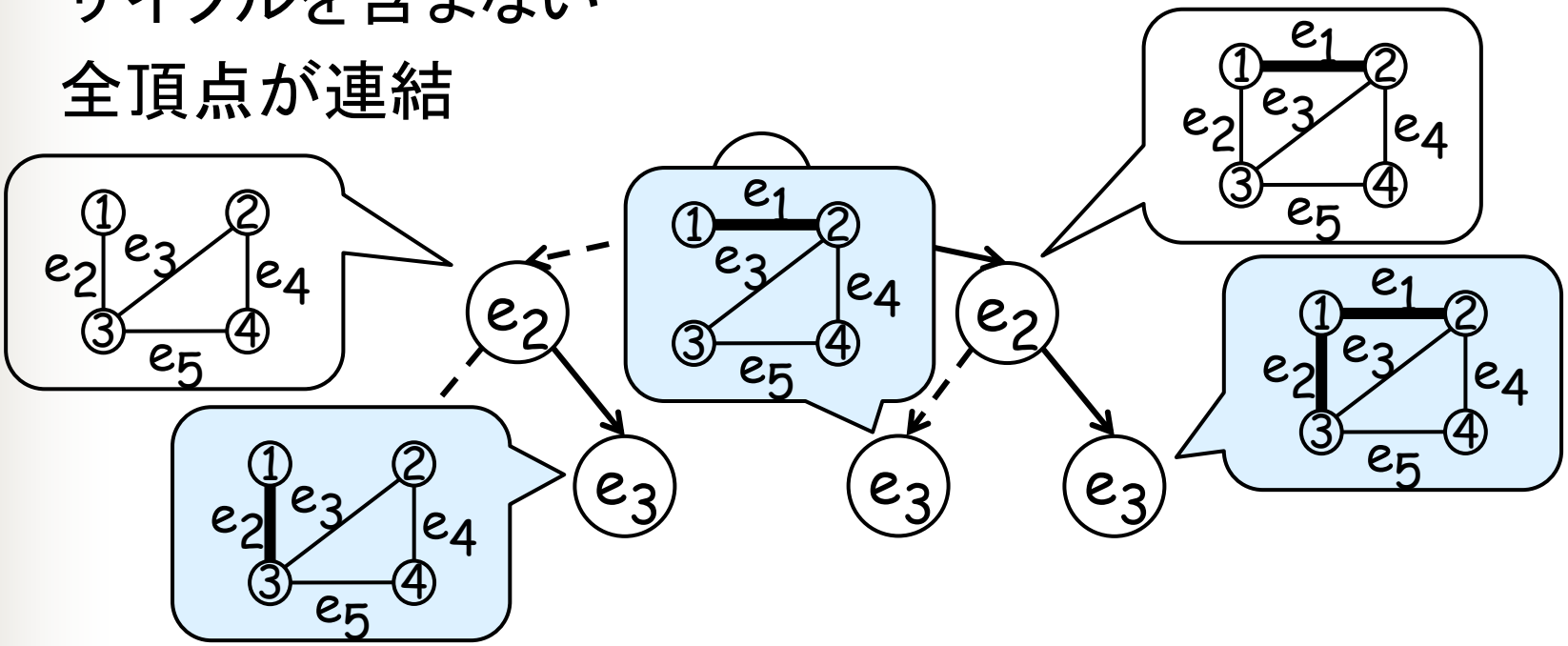
- サイクルを含まない
- 全頂点が連結



頂点 ① が孤立
→ 枝刈り (0 で終端する)

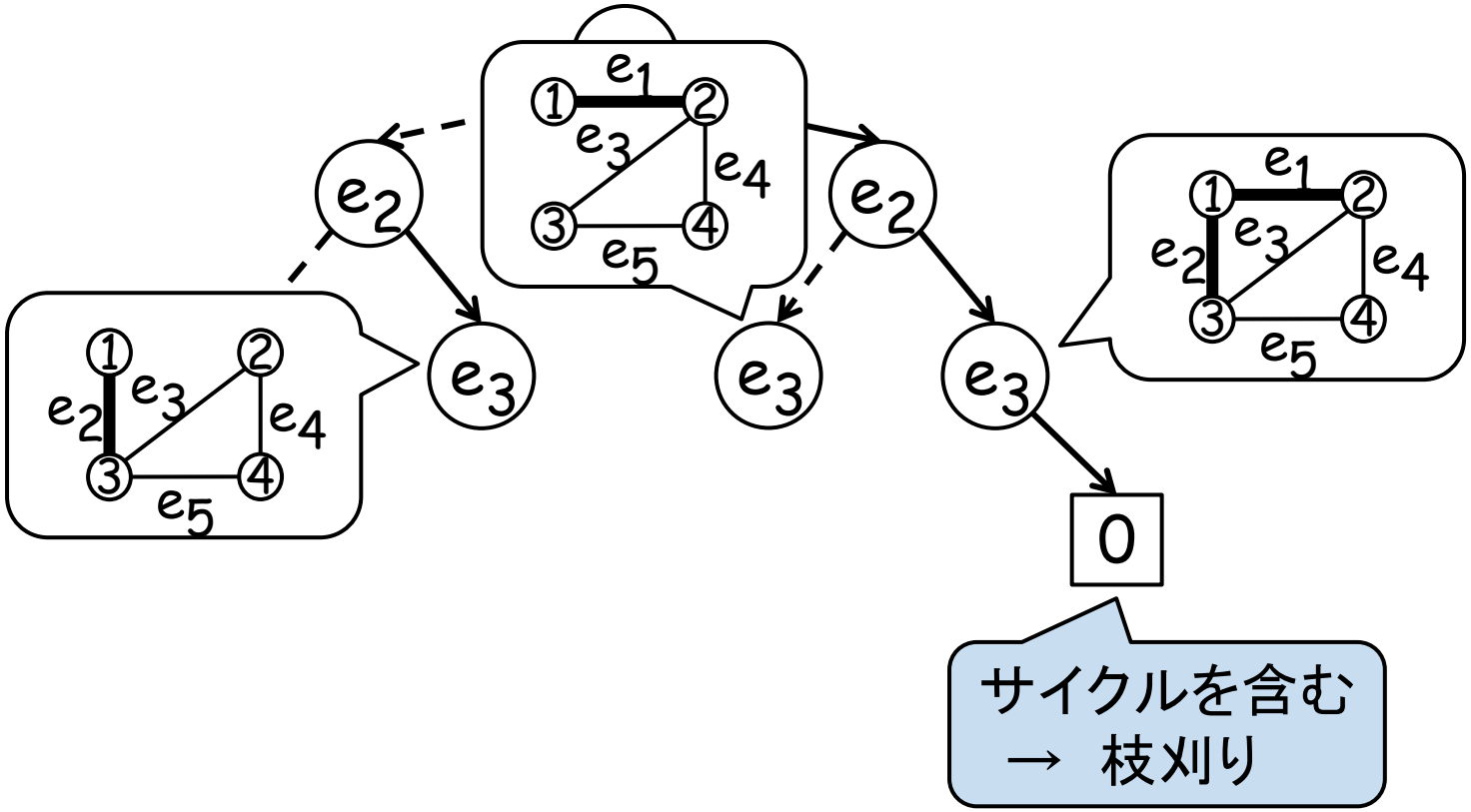
全域木の列挙

- サイクルを含まない
- 全頂点が連結



全域木の列挙

- サイクルを含まない
- 全頂点が連結

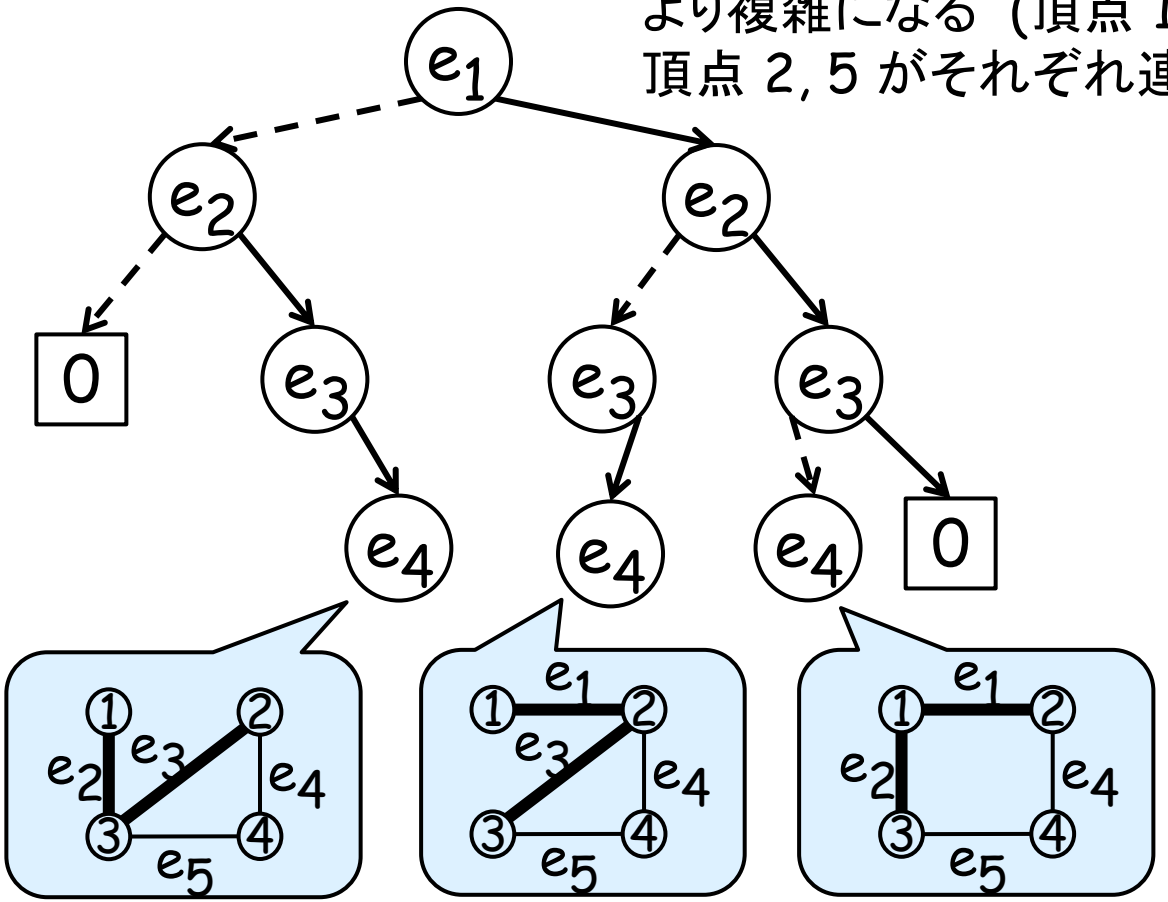


全域木の列挙

- サイクルを含まない
- 全頂点が連結

3つのZDDノードは本質的に同じ
- 頂点 1, 2, 3 が連結
- e_4, e_5 に関する
  残りの部分問題が同じ

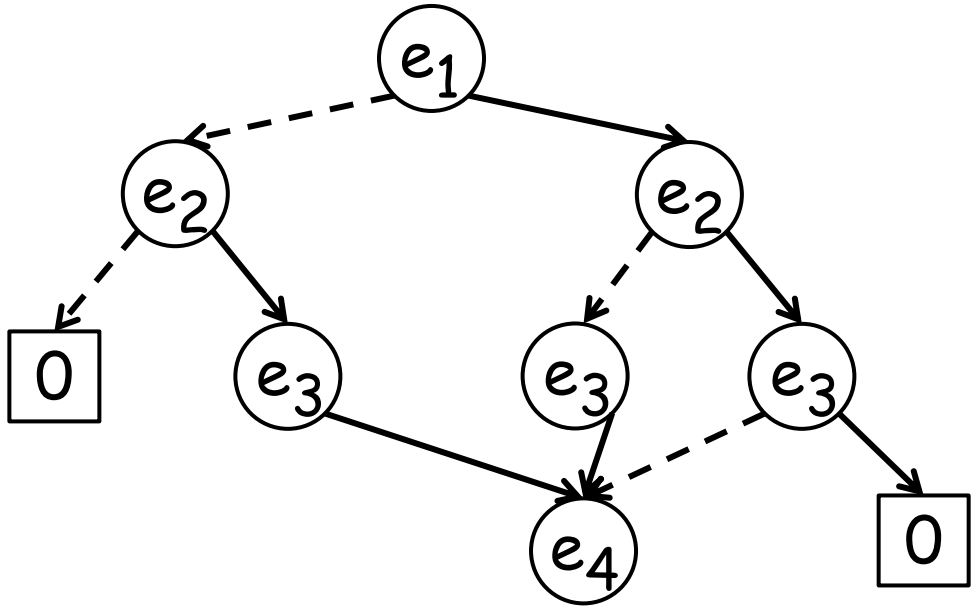
※ 大きなグラフでは、連結性は
より複雑になる (頂点 1, 3, 7、
頂点 2, 5 がそれぞれ連結など)



全域木の列挙

- サイクルを含まない
- 全頂点が連結

3つのZDDノードは本質的に同じ
- 頂点 1, 2, 3 が連結
- e_4, e_5 に関する
  残りの部分問題が同じ



ZDD ノードを共有
→ 高速化、省メモリ化

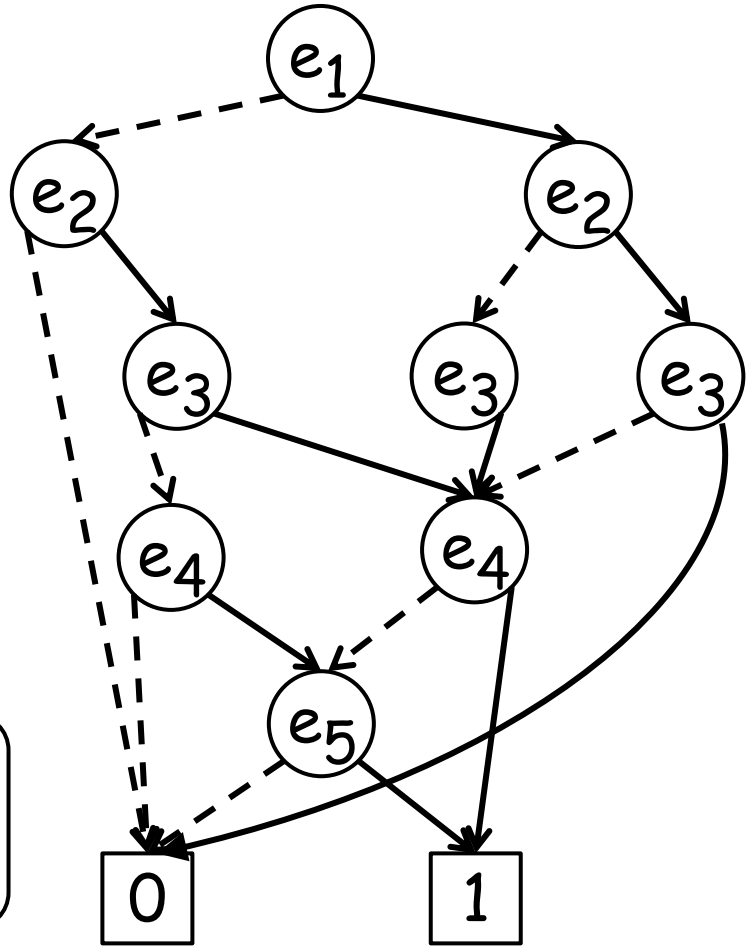
全域木の列挙

- サイクルを含まない
- 全頂点が連結

3つのZDDノードは本質的に同じ
- 頂点 1, 2, 3 が連結
- e_4, e_5 に関する
  残りの部分問題が同じ

どうチェックする？

(各ZDDノードでグラフをすべて持って、毎回そのグラフを探索するのは、非効率)

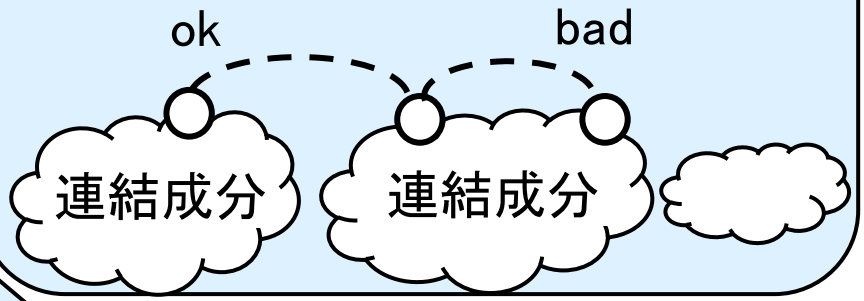


ZDD ノードを共有
→ 高速化、省メモリ化

枝刈りの条件 (0-定数節点に行く条件)

- サイクルを含まない
- 全頂点が連結

同じ連結成分内の頂点を
つなげようとしたら、枝刈り



どうチェックする？

(各ZDDノードでグラフをすべて持って、毎回そのグラフを探索するのは、非効率)

連結成分が1つにまとまらないと確定したら、枝刈り

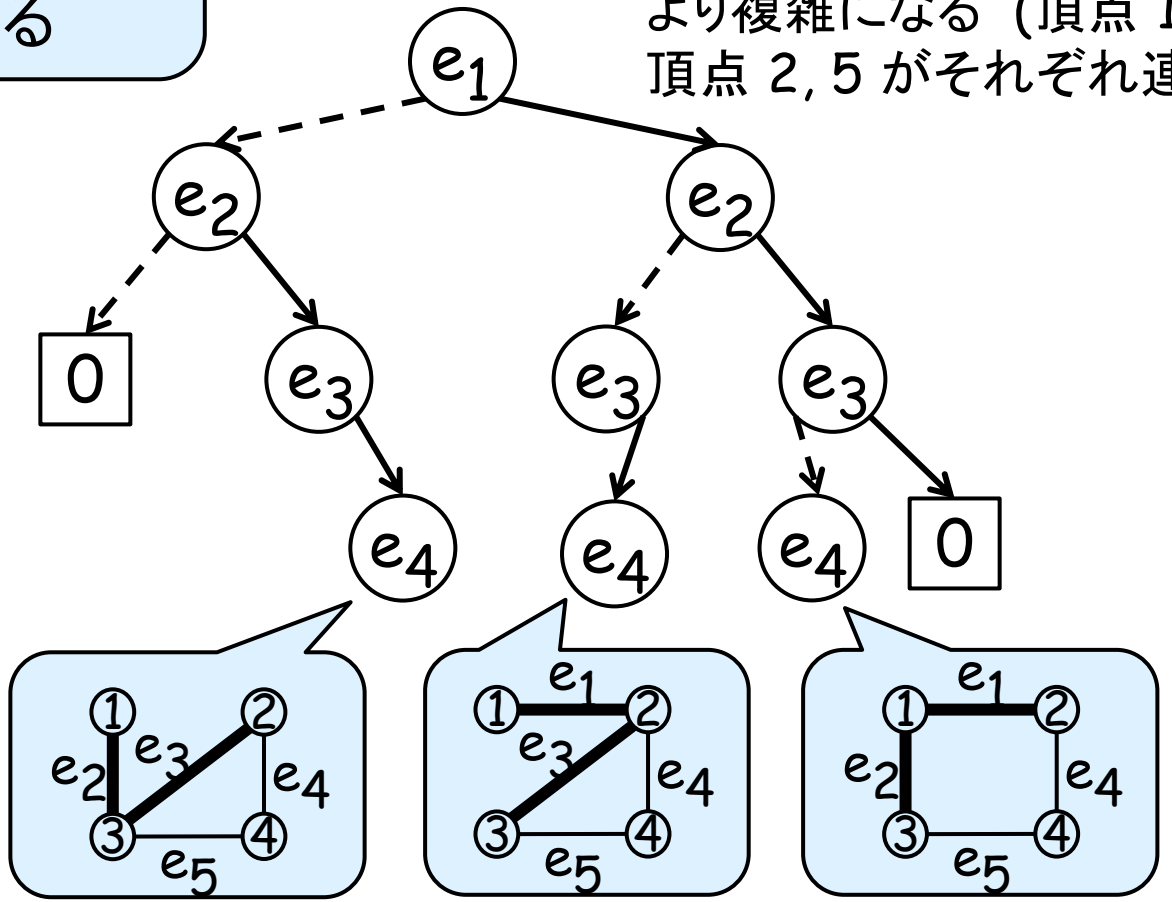
アイデア:
- グラフすべてを持つのではなく、**どの頂点がどの連結成分に属すか**を持つ
- 最初は、すべての頂点が孤立
- 辺を採用したら、2つの連結成分をつなげる

ZDDノードの共有

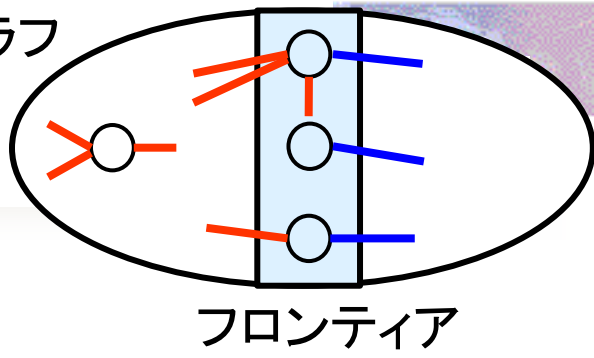
3つのZDDノードは本質的に同じ
- 頂点 1, 2, 3 が連結
- e_4, e_5 に関する
  残りの部分問題が同じ

どの頂点が
どの連結成分に属すか
が、そのまま使える

※ 大きなグラフでは、連結性は
より複雑になる（頂点 1, 3, 7、
頂点 2, 5 がそれぞれ連結など）



フロンティア



どの頂点が
どの連結成分に属すか

覚える量を、
もっと減らしたい

- 頂点を 3 種類に分類して扱う　(ZDDのレベルごとに、どの頂点について覚えるかは共通)
  - **未探索の辺**にのみ隣接する頂点
    - 必ず孤立している → 覚えない
  - **未探索の辺**と**探索済みの辺**に隣接する頂点
    (フロンティア上の頂点)
    - 覚える
  - **探索済みの辺**にのみ隣接する頂点
    - フロンティア上に、同じ連結成分に属す頂点がいる
      (そうでなければ、全頂点が連結でないため、既に枝刈り)
    - フロンティア上の頂点の動向を追えばよい → 覚えない

以上を踏まえて、ZDD の構築を再確認すると、理解が深まる

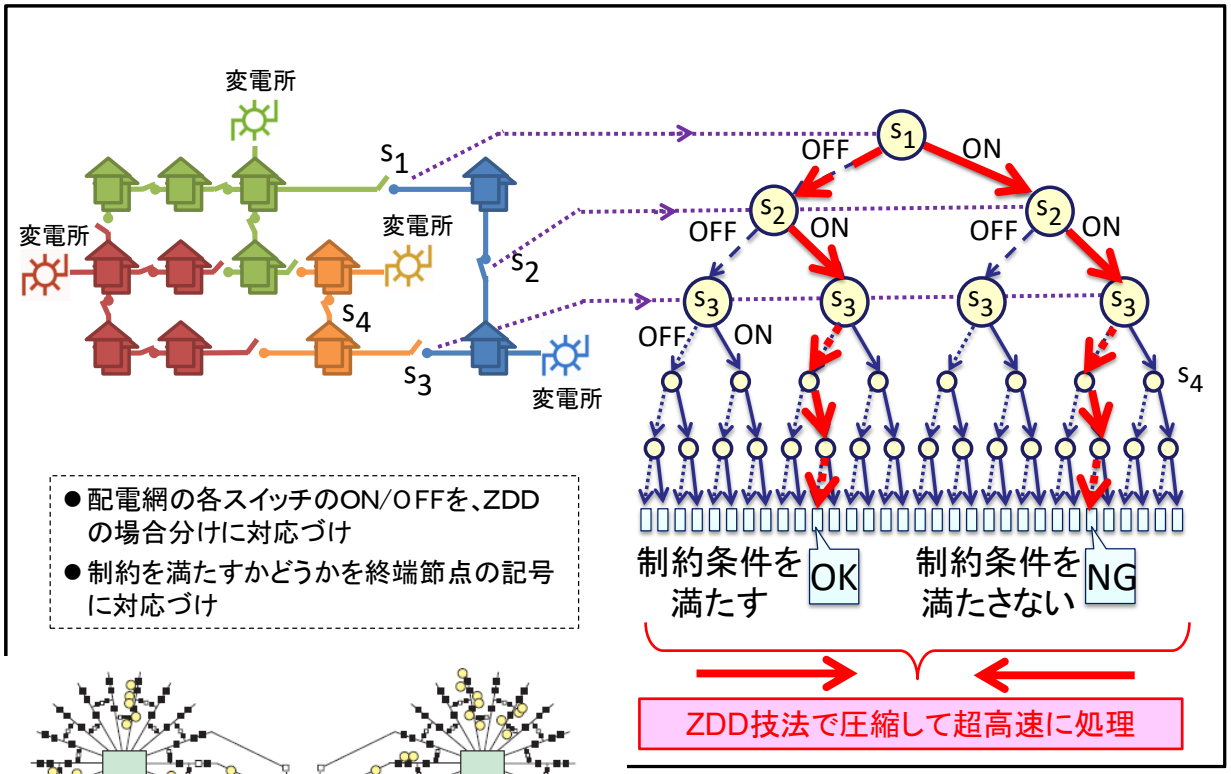
ZDD の応用例: 電力網設計解析

電力網のスイッチ制御

- 標準的な配電網の例

468個のスイッチ

10^{140} の組合せ

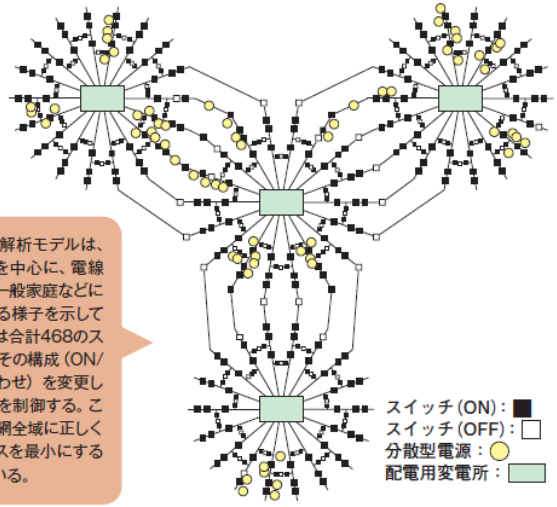


グラフ理論的な制約

+

電気的な制約

(電圧降下など)



配電網の標準解析モデルは、4つの変電所を中心に、電線に接続された一般家庭などに電力を供給する様子を示している。電線には合計468のスイッチがあり、その構成（ON/OFFの組み合わせ）を変更して電力の流れを制御する。この図では配電網全域に正しく給電、送電ロスを最小にする構成を表している。

大震災後に重要性を増した電力網の解析技法の高速化・大規模化・高信頼化

より詳しく知りたい人は

- ERATO湊離散構造処理系プロジェクト 著，
  湊真一 編，**超高速グラフ列挙アルゴリズム**
  ‹フカシギの数え方› が拓く
  組合せ問題への新アプローチ，森北出版，2015.
- R. E. Bryant, Graph-based algorithms for Boolean function manipulation. IEEE Trans. Comput. C-35(8), pp. 677–691, 1986.
- S. Minato, Zero-suppressed BDDs for set manipulation in combinatorial problems, In Proc. of the 30th International Design Automation Conference, 1993, pp. 272–277.
- J. Kawahara, T. Inoue, H. Iwashita, S. Minato, Frontier-Based Search for Enumerating All Constrained Subgraphs with Compressed Representation, IEICE Transactions, 100-A(9), pp. 1773-1784, 2017.

大規模知識処理特論

- この授業の目標
  - 知識の編集・分類・解析・索引化等の知的な情報処理を行うために不可欠な知識処理の技法について学びます
- 大規模知識処理について、以下の観点から
  並行して or 順に 学習を進めます
  - 最適化技法
  - 論理関数と計算量理論の基礎
  - 厳密アルゴリズムと近似アルゴリズム
  - BDD/ZDDによる離散構造処理
- 適当な時期にレポート課題を課します