離散構造処理系:
その概要と最近の研究状況について

2018年11月5日

湊　真一
（京都大学 情報学研究科）

自己紹介

- **湊　真一　（みなと　しんいち）**

北海道大学 大学院 情報科学研究科 教授

JSPS科研費 基盤(S) 研究代表者
「離散構造処理系の基盤アルゴリズムの研究」

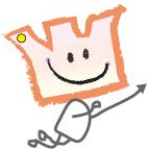
- 石川県出身　金沢大学附属高校卒
- 1990年　京都大学 大学院 修士課程 修了（工・情報・矢島研）
  （1995年　博士（工学）社会人課程）
- 1990年～2004年　NTT研究所（厚木/横須賀）　研究員
- 1997年　米国スタンフォード大学　客員研究員（1年間）
- 2004年～　北海道大学 情報科学研究科 助教授
- 2010年～　同 教授
- **2018年4月～**
  京都大学 情報学研究科 教授

本企画セッションの構成

- [招待講演]
  離散構造処理系:その概要と研究状況について
  - 湊 真一(京大)
- グラフ集合を圧縮して活用するための
  データ構造とアルゴリズム
  - 川原 純(NAIST)
- 離散構造処理系の機械学習への応用
  - 石畠 正和(NTT)

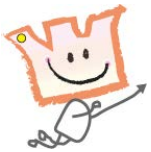
本講演の内容

- 離散構造処理とは
- Decision Diagramによる論理や集合の表現と演算
  - BDDとZDDの基本概念と技術発展の流れ
  - DDの様々な派生形
- DDによるグラフ部分構造の列挙と解析
  - フロンティア法の基本的考え方
  - 社会的に重要な問題への応用
- 最近の研究トピックから
  - 列挙・数え上げの技法に基づく
    確率・統計・データサイエンス分野への応用



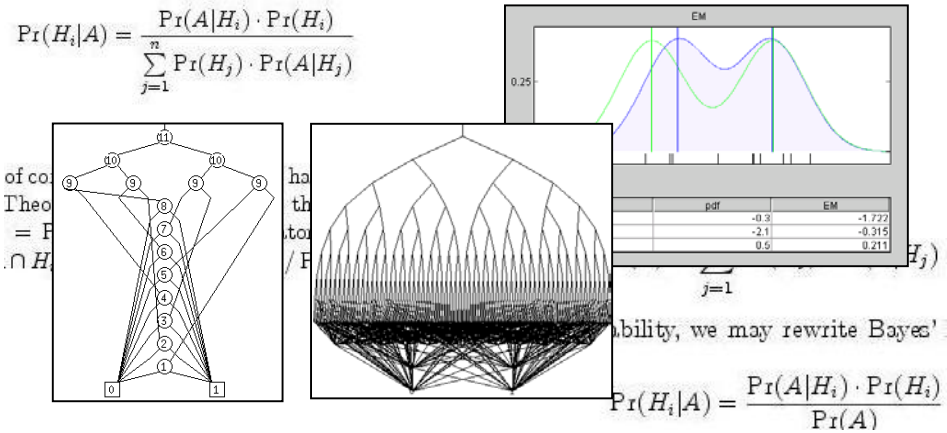
本講演の内容

- 離散構造処理とは
- Decision Diagramによる論理や集合の表現と演算
  - BDDとZDDの基本概念と技術発展の流れ
  - DDの様々な派生形
- DDによるグラフ部分構造の列挙と解析
  - フロンティア法の基本的考え方
  - 社会的に重要な問題への応用
- 最近の研究トピックから
  - 列挙・数え上げの技法に基づく
    確率・統計・データサイエンス分野への応用

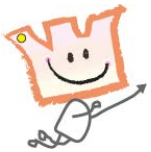


離散構造とその処理

- 離散構造とは
  - **離散数学および計算機科学の基礎をなす数学的構造**
  - 集合理論、記号論理、帰納的証明、グラフ理論、組合せ論、確率論などを含む

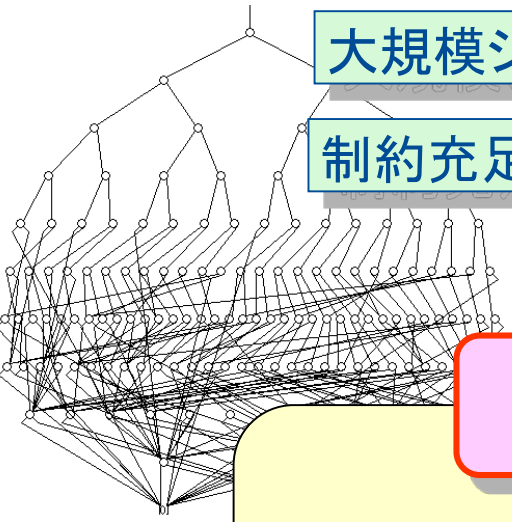


- およそ計算機が扱うあらゆる問題は、単純な基本演算を要素とする離散構造の処理に帰着される。
  - 最終的には、**膨大な数の場合分け処理**を要することが多い。
- 離散構造の処理は、計算機の様々な応用分野に共通する基盤技術
  - 典型的な応用分野： システム設計自動化、大規模システム故障解析、制約充足問題、データマイニングと知識発見、機械学習と自動分類、生命情報科学、web情報解析、etc.
  - **現代情報化社会に対する波及効果は極めて大。**



離散構造処理系プロジェクトの基本構想

- 様々な工学的応用を持つ基盤技術として「**離散構造処理系**」に着目し、研究開発を行う



システム設計自動化
データマイニングと知識発見
大規模システム故障解析
機械学習と自動分類
制約充足問題
生命情報科学
web情報解析

工学的応用
→ 社会への影響大

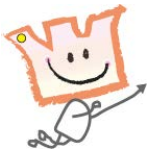
本研究の対象領域

性能向上
(10倍～100倍以上)

離散構造処理系

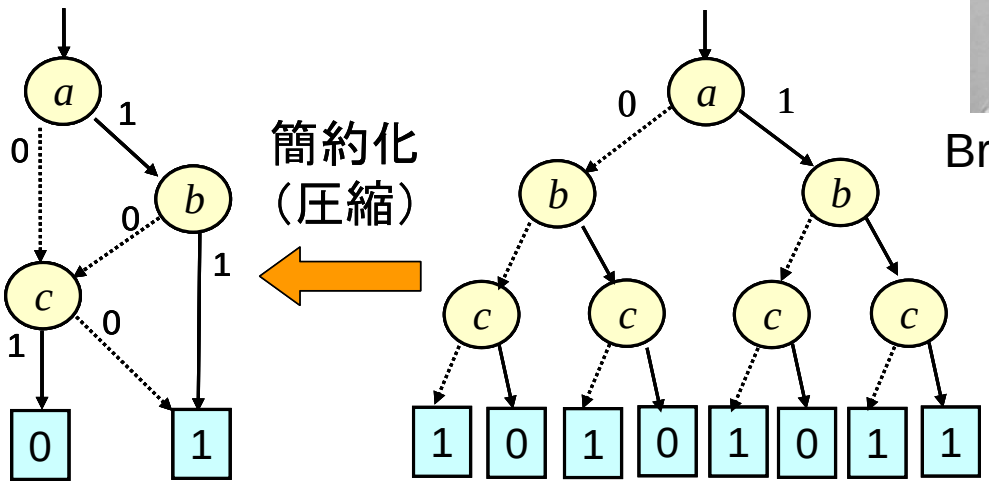
集合理論
記号論理
帰納的証明
組合せ論
グラフ理論
確率論

数学的概念構造



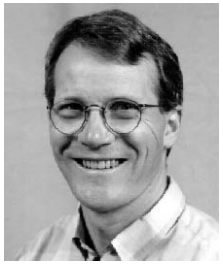
BDD（二分決定グラフ）

離散構造の最も基本的なモデルである「論理関数」の処理技法



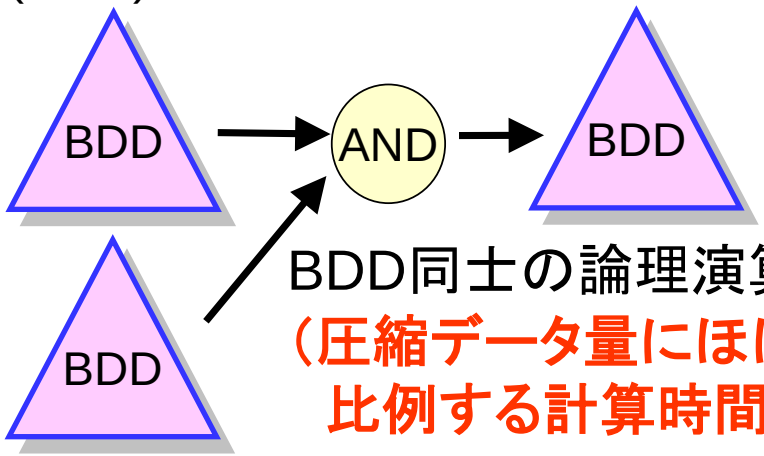
（BDD）　（場合分け二分木）

- 場合分け二分木グラフを簡約化（データ圧縮）
- **多くの実用的な論理データをコンパクトかつ一意に表現。**（数十～数百倍以上の圧縮率が得られる例も）



Bryant (CMU)

1986年に画期的なBDD演算**（Apply演算アルゴリズム）**を提案。以後急速にBDD技術が発達。（長期間、情報科学の全分野での最多引用文献となった）

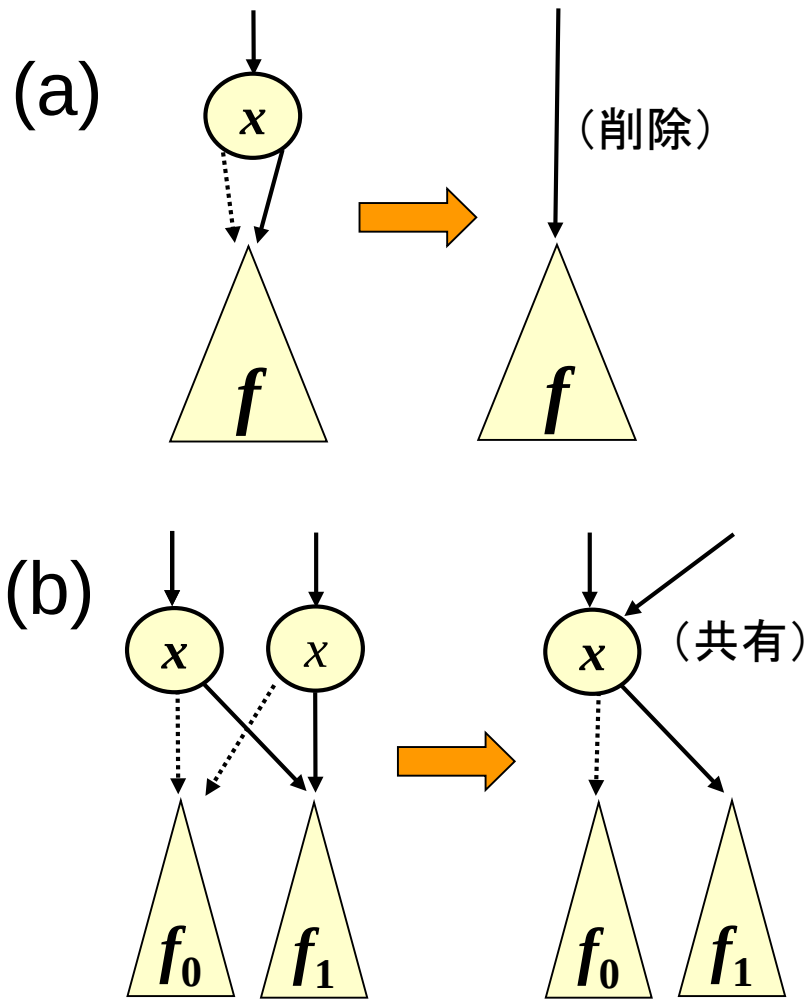


BDD同士の論理演算

（圧縮データ量にほぼ比例する計算時間）

近年のPC主記憶の大規模化により、BDDの適用範囲が拡大（特に2000年以降）

BDDの簡約化規則



(a) 冗長な節点を全て削除

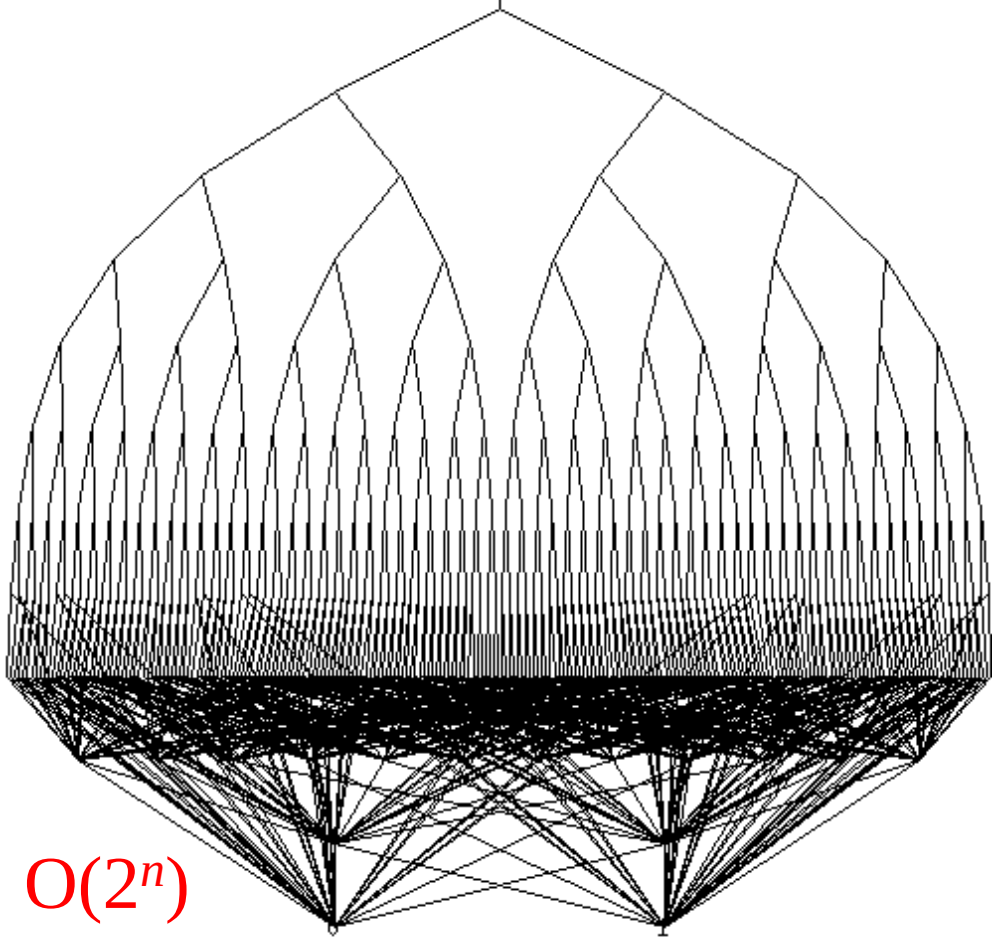
(b) 等価な節点を全て共有

既約なBDDが得られる

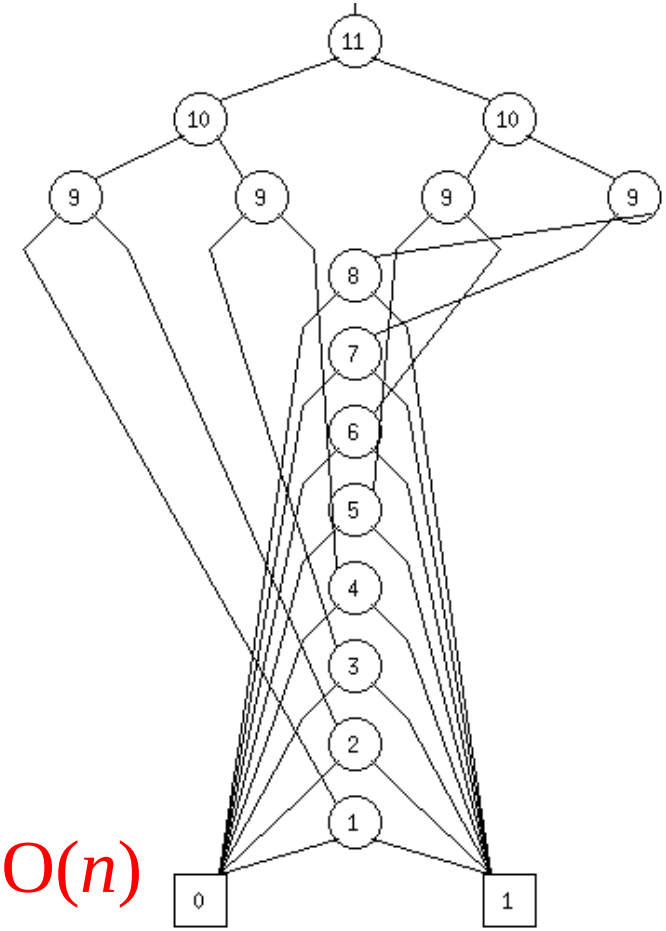
変数順序が固定されていれば
論理関数に対して一意な表現

BDD簡約化の効果

- 特定の問題では、指数関数的な圧縮効果が得られる。
  - 例題に依存するが、多くの実用的な問題で効果がある。

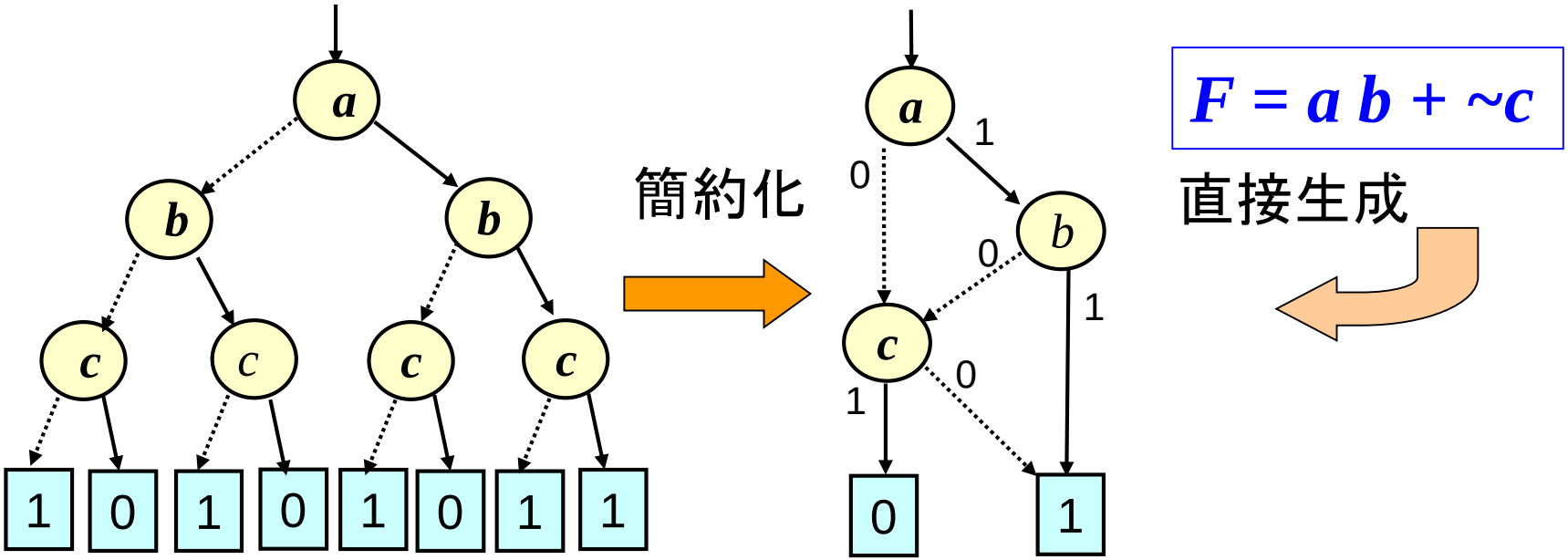


$O(2^n)$



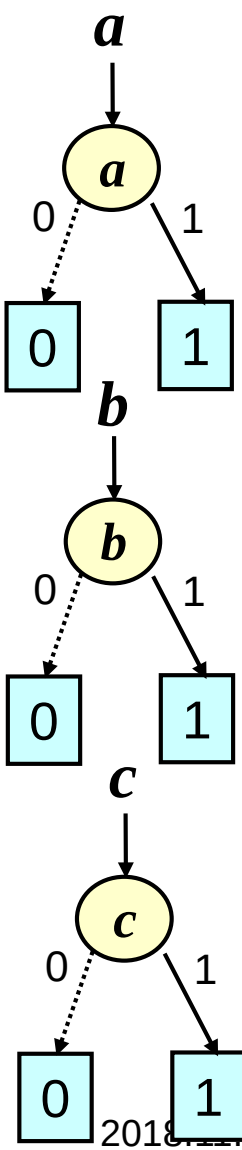
$O(n)$

BDDの生成アルゴリズム

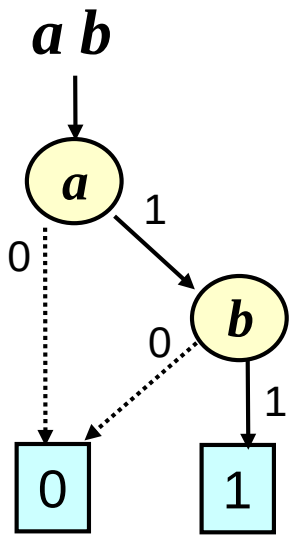


- 真理値表に対応する二分木を簡約化する方法では、常に指数オーダの記憶量と処理時間がかかってしまう。
  → 実用的には、論理式からBDDを直接生成するアルゴリズム[Bryant86]を用いる

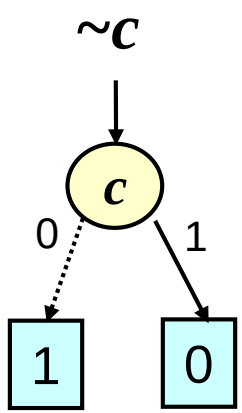
論理式からのBDD生成



AND
演算

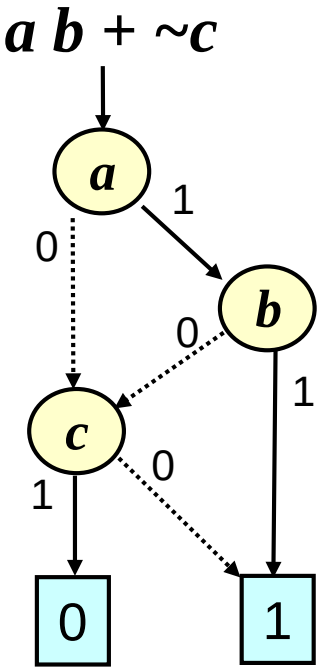


NOT
演算



- 2つのBDDの間の二項論理演算を繰り返して任意のBDDを生成
  （圧縮データ量にほぼ比例する計算時間で実行可能）

OR
演算



論理関数と組合せ集合

a	b	c	F	
0	0	0	0	
1	0	0	0	
0	1	0	0	
1	1	0	1	→ ab
0	0	1	1	→ c
1	0	1	1	→ ac
0	1	1	0	
1	1	1	0	

論理関数:

$$F = (a\ b\ {\sim}c) \vee ({\sim}b\ c)$$

組合せ集合:

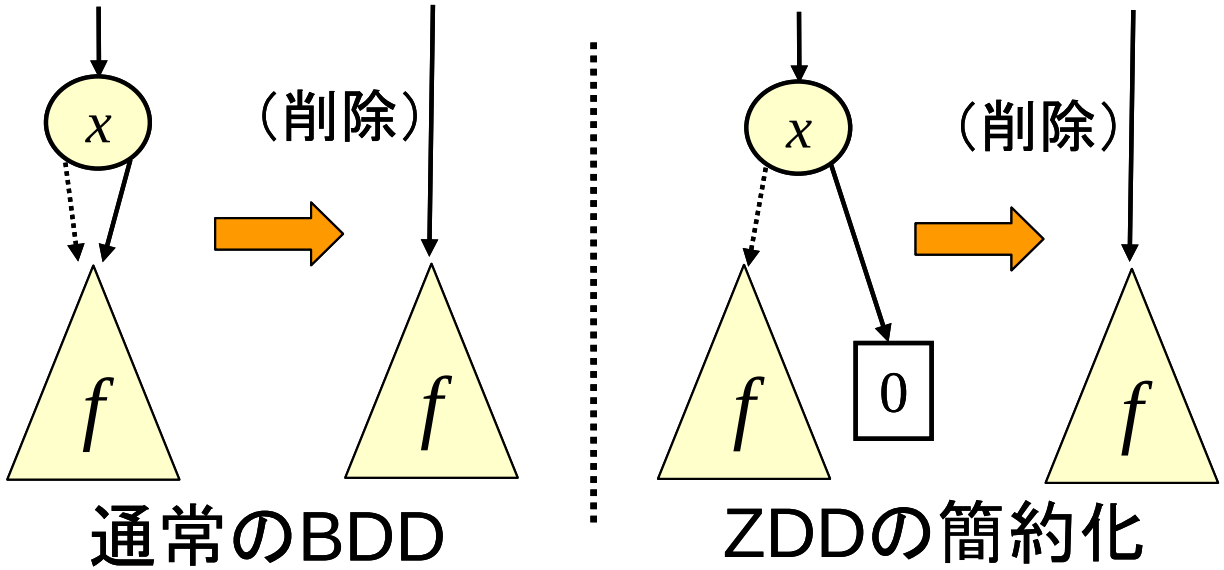
$$F = \{ab, ac, c\}$$

(買い物客の購入品)

- 組合せ集合と論理関数の演算は対応関係がある。
  - Union of sets → logical OR
  - Intersection of sets → logical AND
  - Complement set → logical NOT

ZDD（ゼロサプレス型BDD）による集合の表現

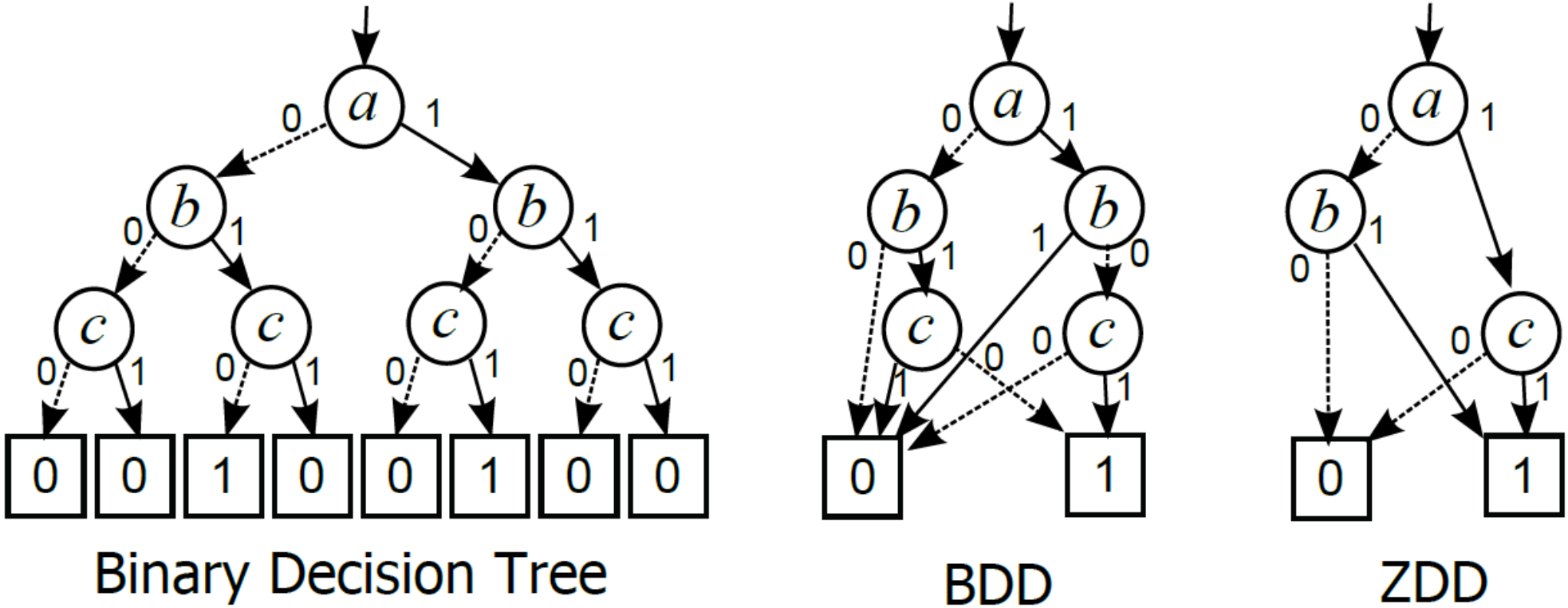
- **「組合せ集合」**を効率的に表現するためのBDDの改良
  - **湊が世界で初めて考案し命名（1993年）**
  - **通常と異なる簡約化規則**を考案。
  - **疎な集合の族を扱う場合に著しい効果が得られる。**
    （例：　商店の陳列アイテム数に比べて1顧客の購入点数は極めて少ない。）



- ZDDはBDDの改良技術として現在、世界的に広く使われている。
  - 最近では、データマイニング分野に応用されて、画期的な有効性が示されている。（数百倍のデータ圧縮率・数十倍の処理高速化）
  - 他にも応用例は増えつつある。

BDDとZDDの比較

同じ二分木から簡約化したBDDとZDD

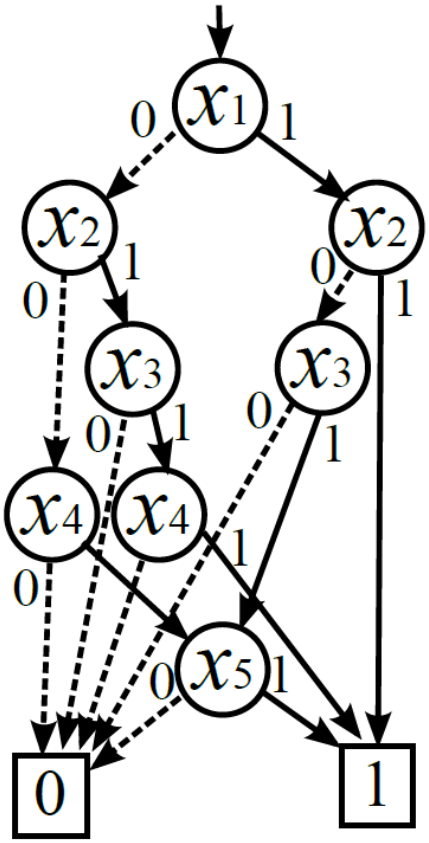


ZDDの特長

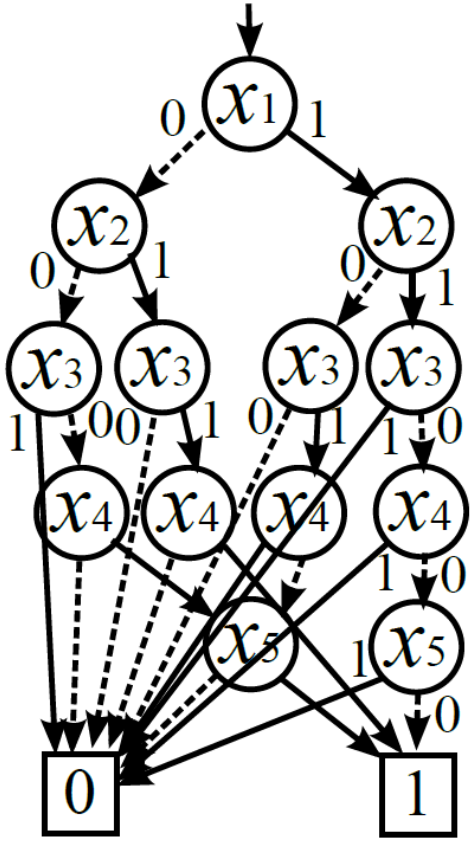
- 組合せ集合に出現しないアイテムに関する節点が自動的に取り除かれる。
  - 通常のBDDでは無駄に残ってしまい、BDDの共有化による効果を帳消しにしてしまう場合があった。
- 疎な組合せの集合に対して顕著な削減効果がある
  - 例えば1000個のアイテムから平均10個しか選ばないような組合せ集合では、通常のBDDに比べて約100倍の節点削減効果
- グラフの根（最上位）節点から1-終端節点へ至る経路の数が、集合の要素数と完全に一致する。
  - 通常のBDDでは必ずしも一致しない。

ZDDとBDDの比較（疎な組合せ集合の場合）

$$\{x_1x_3x_5, x_1x_2, x_2x_3x_4, x_4x_5\}$$



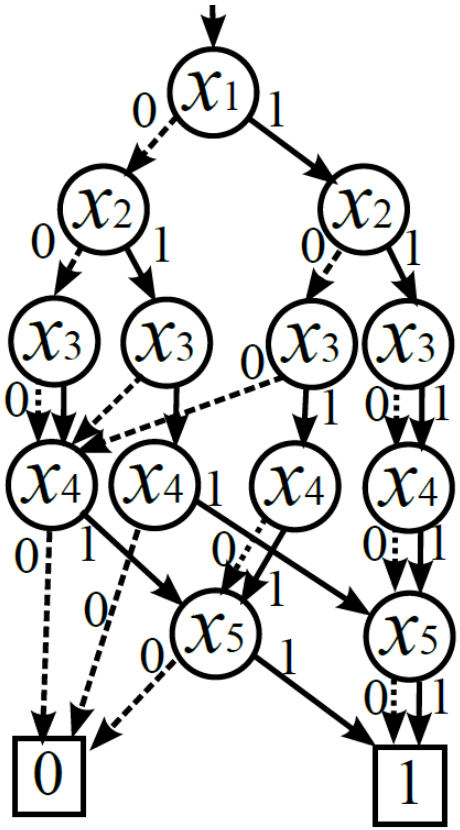
ZDD



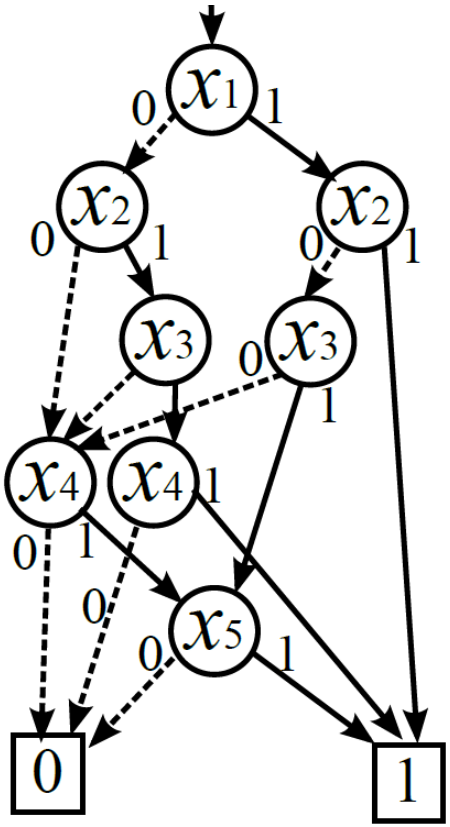
BDD

ZDDとBDDの比較（密な組合せ集合の場合）

$$\{x_1x_2, x_1x_2x_3, x_1x_2x_3x_4, x_1x_2x_3x_4x_5, x_1x_2x_3x_5, x_1x_2x_4, x_1x_2x_4x_5, x_1x_2x_5,$$
$$x_1x_3x_5, x_1x_3x_4x_5, x_2x_3x_4, x_2x_3x_4x_5, x_4x_5, x_1x_4x_5, x_2x_4x_5, x_3x_4x_5\}$$



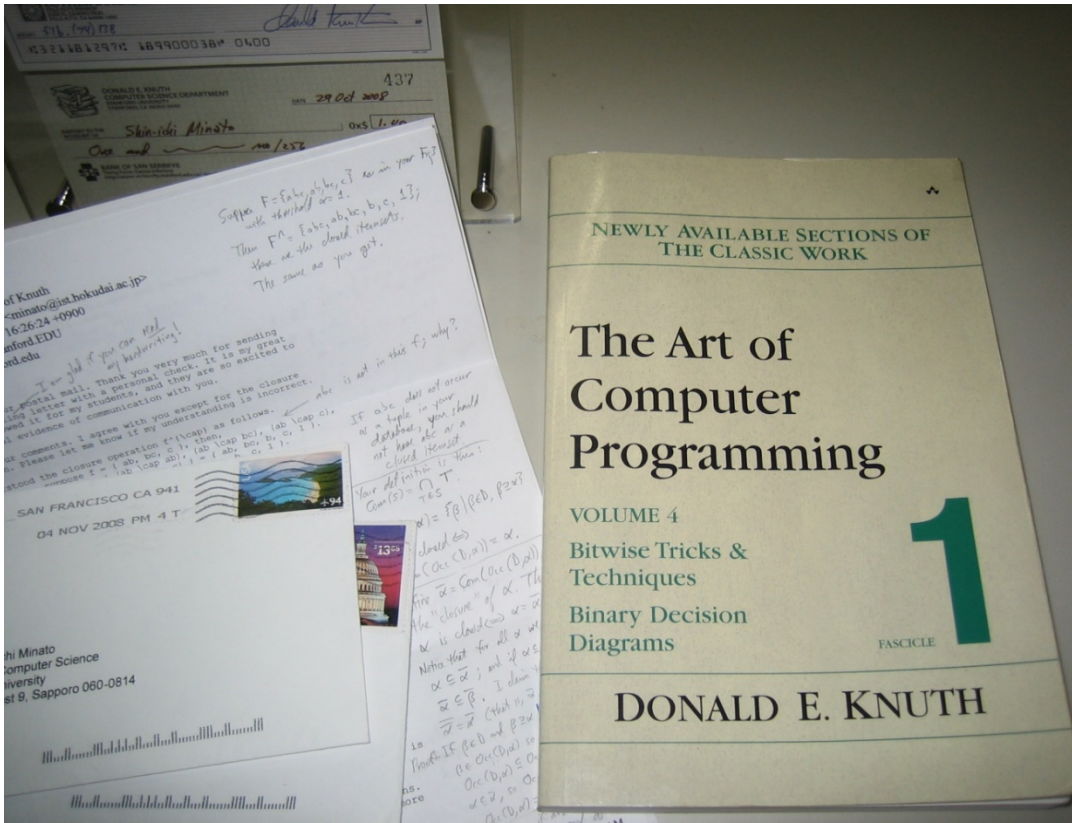
ZDD



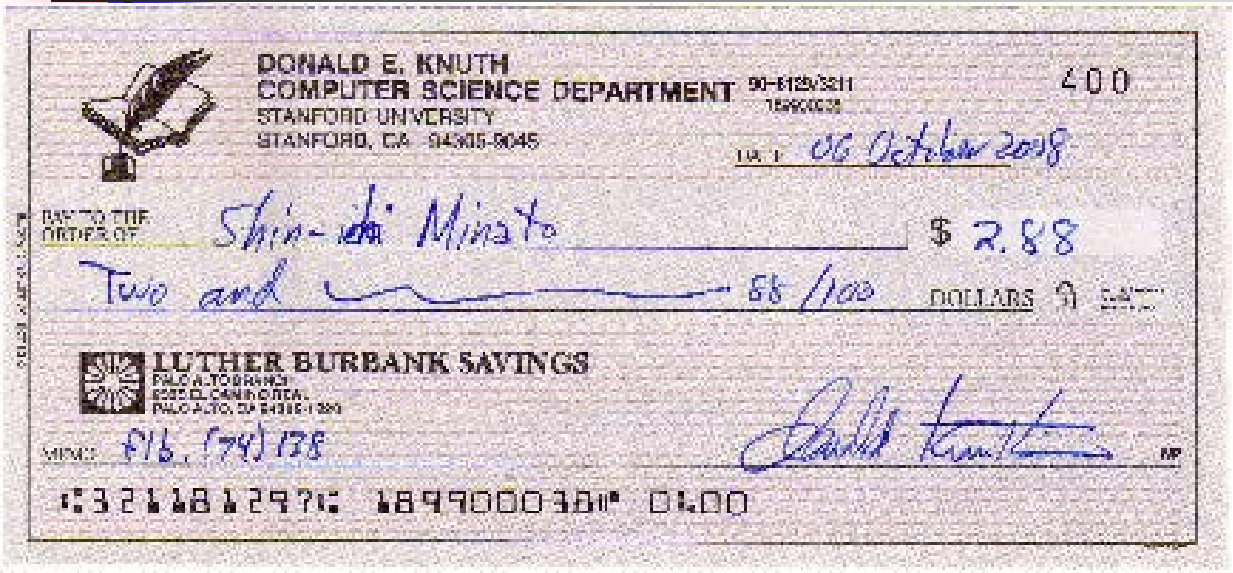
BDD

BDD / ZDD に関する話題

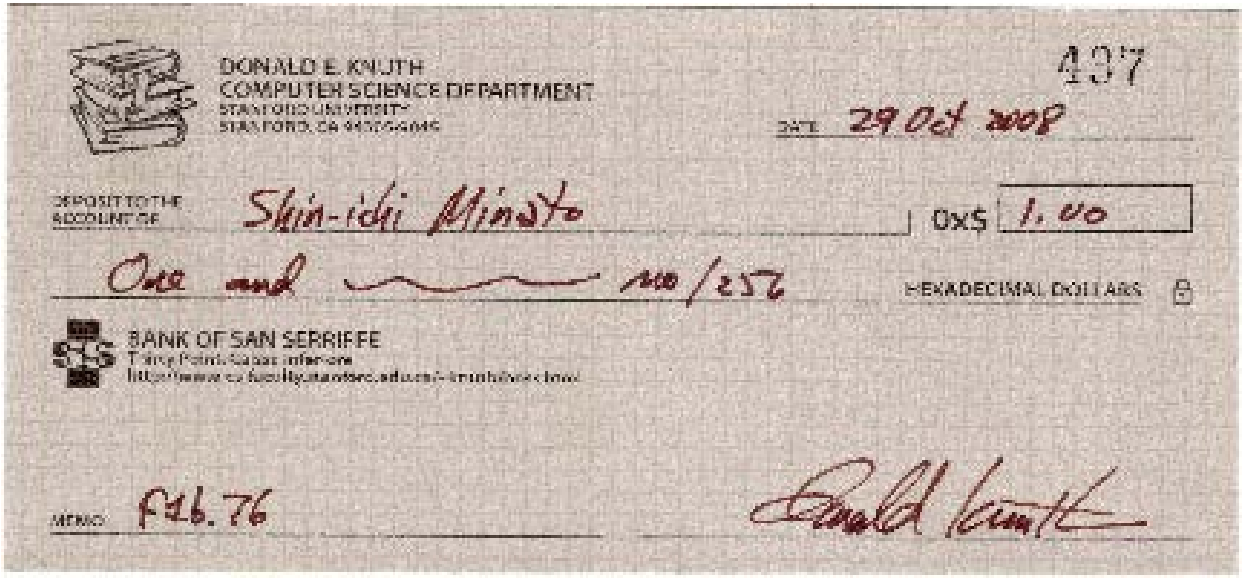
- Knuthの名著「The Art of Computer Programming」の最近の分冊（Vol.4, Fascicle 1, 2009）で、BDDが詳しく解説された。
  - BDDだけで140ページ、演習264問の大作
- 種々のBDD改良技法の中で、**特にZDDを重要視**
  - 章の後半30ページ以上にわたり、ZDDについて記述
  - ZBDDだった呼び名をZDDに短縮。（BDDと同格扱いに）
  - ZDD演算体系を再構築し、「Family Algebra」と命名。
  - **日本人の研究成果が、このシリーズに項目として詳細に掲載されるのは初めて。**
  - Knuth氏本人から、ZDD考案者として校正作業への協力を依頼する長文のメールと直筆の手紙を受領。
  - Knuth氏は、オックスフォードとスタンフォードで「Fun with ZDDs」というタイトルで特別講義。



Knuth先生からの小切手（誤り訂正の証明書）



（2008年10月上旬まで）
実在する銀行の小切手。
誤り指摘1件：2.56ドル
改善提案1件：0.32ドル
（実際は誰も換金しない。）



（2008年10月下旬以降）
架空の銀行の預金証明。
誤り指摘1件：
0x 1.00 (16進)ドル

ZDDの演算系（Algebra）

- Knuth先生はZDDのデータ構造だけでなく
  演算系（Algebra）に強い興味を示された。

φ, {1}	***Empty*** and ***singleton set***. (0/1-terminal)
P.top	Returns the ***item-ID*** at the top node of *P*.
P.onset(v) *P.offset(v)*	Selects the subset of itemsets including or excluding *v*.
P.change(v)	Switching *v* (***add / delete***) on each itemset.
∪, ∩, ＼	Returns ***union, intersection,*** and ***set difference***.
P.count	***Counts number*** of combinations in P.
*P * Q*	***Cross-product set*** of *P* and *Q*.
P / Q	***Quotient set*** of *P* divided by *Q*.
P % Q	***Remainder set*** of *P* divided by *Q*.

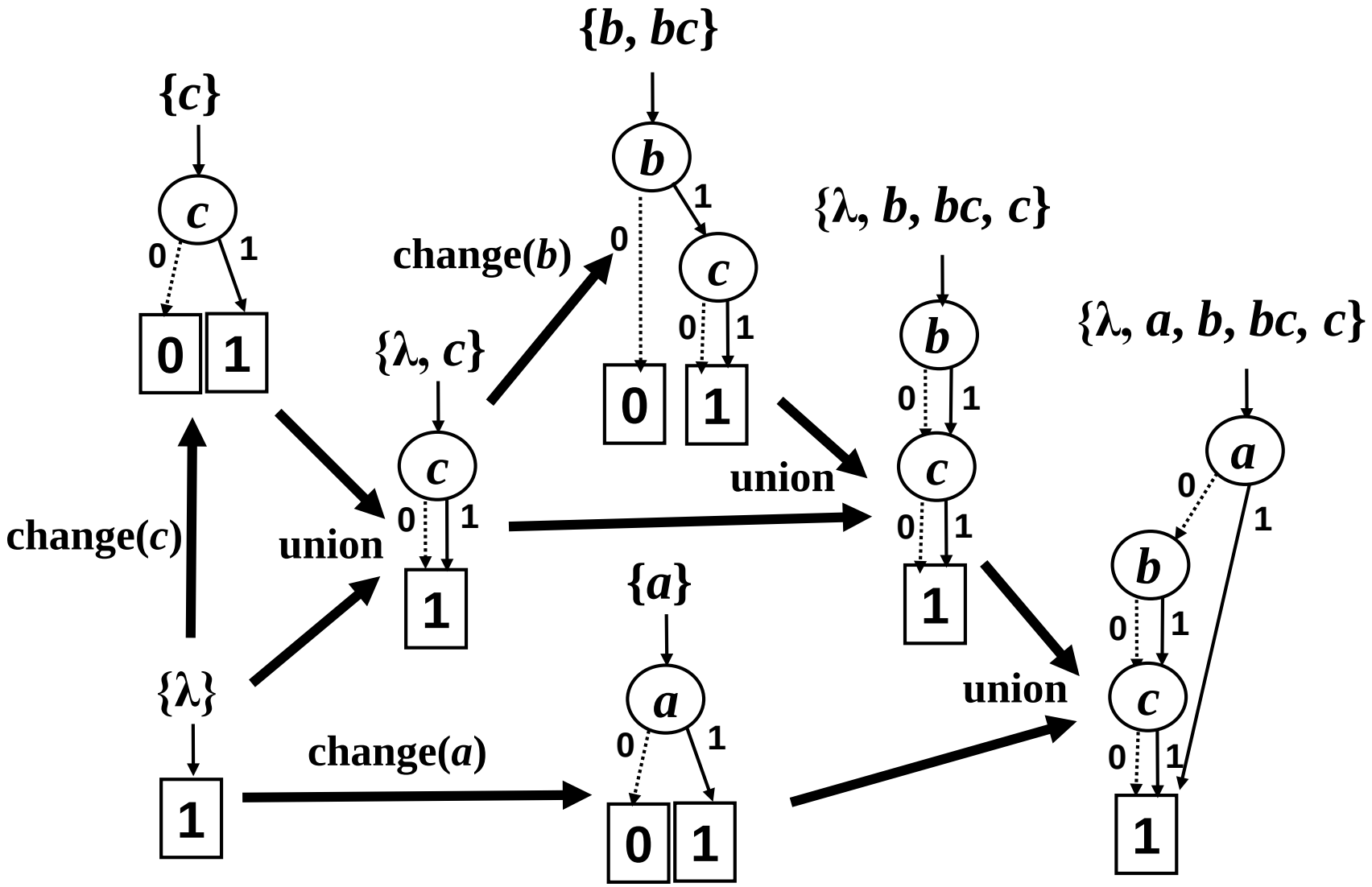
基本演算
（ブール代数に対応）

組合せ集合に
特有の演算
（湊が新しく導入）

多くの応用に役立つ

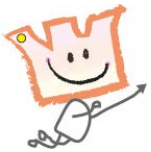
当初は「unate cube set algebra」と呼んでいたが
Knuthにより「family algebra」として再構成された

ZDDの構築例

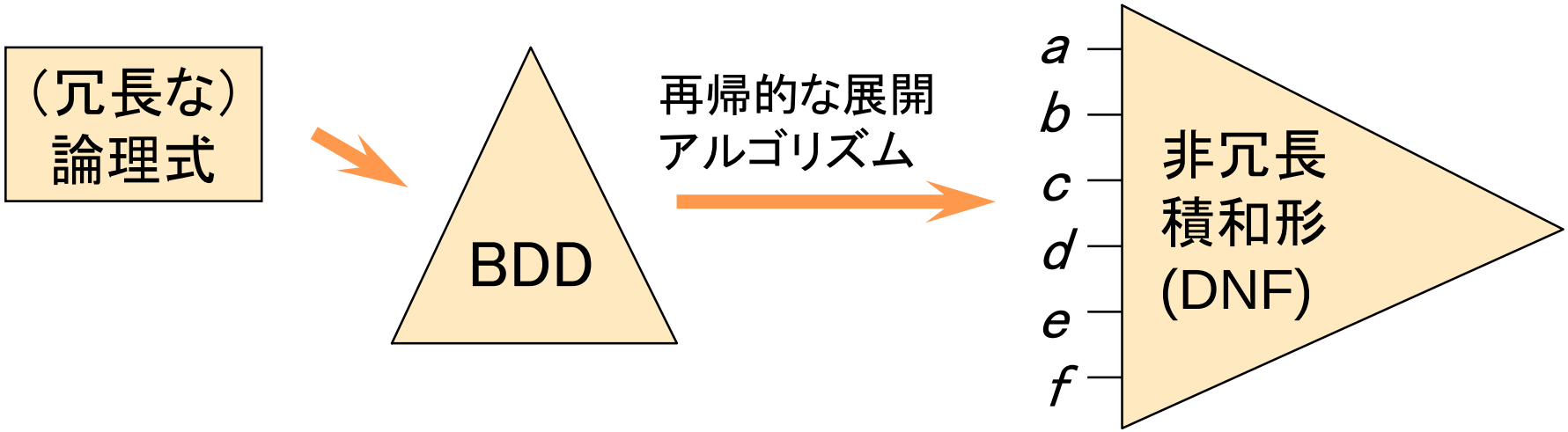

{c}
c
0
1
0
1
{b, bc}
b
1
0
c
0
1
0
1
{λ, b, bc, c}
b
0
1
c
0
1
1
{λ, a, b, bc, c}
a
0
1
b
0
1
c
0
1
1
change(b)
{λ, c}
c
0
1
1
union
union
change(c)
{λ}
1
change(a)
{a}
a
0
1
0
1
union

ZDDの応用

- 元々はLSI設計の論理式（CNF/DNF)の簡単化に利用
  - 膨大な項数の論理式を高速に因数分解する方法[Minato96]
  - 算術式の因数分解法[Minato97]
- データマイニングへの応用
  - ZDDを用いた頻出パタンマイニング[Minato2005]
  - LCM over ZDDs アルゴリズム[Minato-Uno2008]
  - 時分割データベースからのパタン変化の検出[Minato2010]
- グラフに関する種々の組合せ問題への応用
  - 最大クリーク問題、彩色問題、カバー問題等[Coudert97]
  - ZDDを用いた超高速パス列挙アルゴリズム[Knuth2009]
  - 電力ネットワークの制御、避難所配置問題、通信NW、他



BDD/ZDDを用いた大規模二段論理合成法[Minato96]



- BDDから非冗長積和形（DNF）を直接合成する方法
  - 非冗長積和形　→　どの1つの積項または変数を削除しても（＝どの信号線を削除しても）出力に影響がある形。最小とは限らないが、比較的良い局所解。
- 積項数が百万を超えるような超大規模な積和論理式を現実的な時間と記憶量で合成することが可能に
- **合成結果はZDDを用いて圧縮した状態で出力（百万文字を超える論理式を数千ノードのZDDで表現）**
- さらにZDD上の演算で高速に因数分解して多段論理回路を合成

ZDD応用例：頻出アイテム集合抽出問題

- データマイニングの最も基本的な問題
  - 最小出現頻度α以上のレコードに含まれるアイテム組合せの部分集合を抽出・列挙する問題

レコード番号	アイテム集合
1	a b c
2	a b
3	a b c
4	b c
5	a b
6	a b c
7	c
8	a b c
9	a b c
10	a b
11	b c

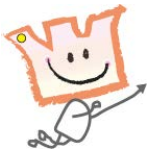
最小頻度 α = 10 → { b }

最小頻度 α = 8 → { ab, a, b, c }

最小頻度 α = 7 → { ab, bc, a, b, c }

最小頻度 α = 5 → { abc, ab, bc, ac, a, b, c }

最小頻度 α = 1 → { abc, ab, bc, ac, a, b, c }



LCM-ZDD 法による高速化

- 計算結果の頻出アイテム集合を、メモリ上に圧縮してZDDで表現し、そのポインタのみを返す。
  - 計算結果をファイルに出力しない。

レコード番号	アイテム集合
1	*a b c*
2	*a b*
3	*a b c*
4	*b c*
5	*a b*
6	*a b c*
7	*c*
8	*a b c*
9	*a b c*
10	*a b*
11	*b c*

LCM-ZDD法

(最小頻度 $\alpha = 7$)

$\{ ab, bc, a, b, c \}$

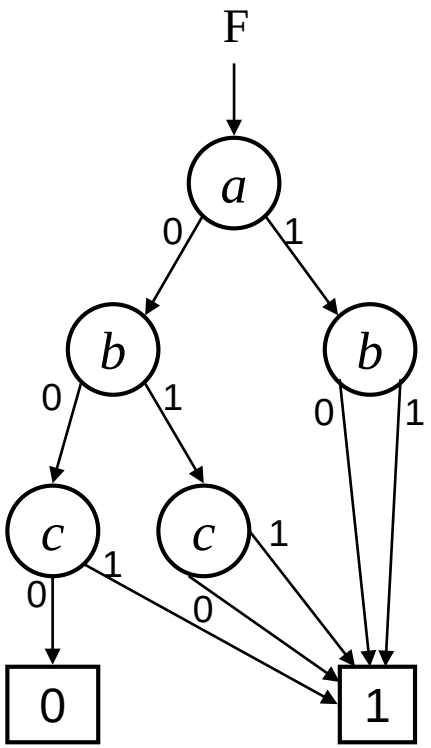


Table 2. Co CM ov ith the prev

solutions

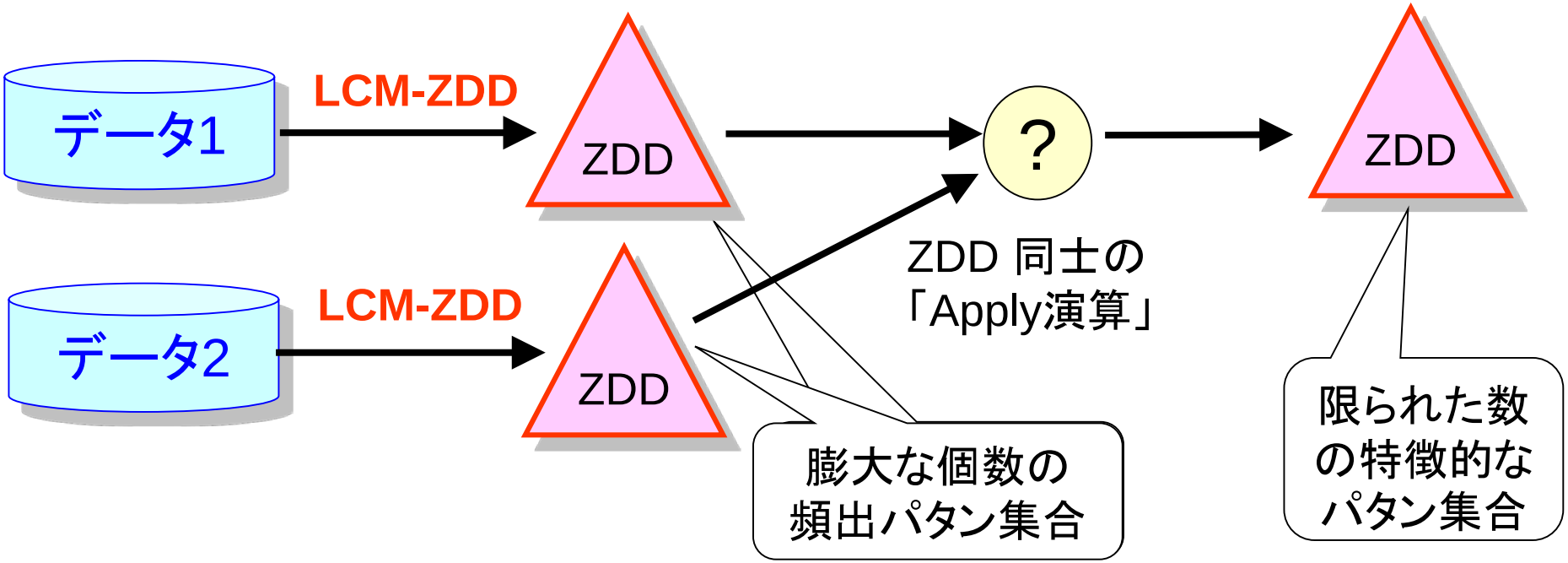
LCM-ZDD

Original LCM

Dataset name: min. support	# frequent itemsets	LCMoverZBDDs |ZBDD|	LCMoverZBDDs Time(s)	LCM-count Time(s)	LCM-dump Time(s)	ZBDD-growth Time(s)
mushroom: 1,000	123,287	760	0.50	0.49	0.64	1.78
500	1,442,504	2,254	1.32	1.30	3.29	3.49
300	5,259,786	4,412	2.25	2.22	9.96	5.11
200	18,094,822	6,383	3.21	3.13	31.63	6.24
100	66,076,586	11,584	5.06	4.87	114.21	6.72
70	153,336,056	14,307	7.16	7.08	277.15	6.97
50	198,169,866	17,830	8.17	7.86	357.27	6.39
T10I4D100K: 100	27,533	8,482	0.85	0.85	0.86	209.82
50	53,386	16,872	0.97	0.92	0.98	242.31
20	129,876	58,413	1.13	1.08	1.20	290.78
10	411,366	173,422	1.55	1.36	1.64	332.22
5	1,923,260	628,491	2.86	2.08	3.54	370.54
3	6,169,854	1,576,184	5.20	3.15	8.14	386.72
2	19,561,715	3,270,977	9.68	5.09	22.66	384.60
BMS-WebView-1: 50	8,192	3,415	0.11	0.11	0.12	29.46
40	48,544	10,755	0.18	0.18	0.22	48.54
36	461,522	28,964	0.49	0.42	0.98	67.16
35	1,177,608	38,164	0.80	0.69	2.24	73.64
34	4,849,466	49,377	1.30	1.07	8.58	83.36
33	69,417,074	59,119	3.53	3.13	144.98	91.62
32	1,531,980,298	71,574	31.90	29.73	3,843.06	92.47
chess: 1,000	29,442,849	53,338	197.58	197.10	248.18	1,500.78
connect: 40,000	23,981,184	3,067	5.42	5.40	49.21	212.84
pumsb: 32,000	7,733,322	5,443	60.65	60.42	75.29	4,189.09
BMS-WebView-2: 5	26,946,004	353,091	4.84	3.62	51.28	118.01

2018.11.05

ZDDを用いたパタン集合演算



- 数十億ものパタンを含む集合を圧縮して表現し、ZDDの集合演算を使って効率よく絞込みを行える。
  - 従来の明示的な表現方法では，意味のある解析処理を現実的な時間で行うことは不可能

論理関数 vs. 組合せ集合

データモデル	論理関数	アイテム集合 (組合せ集合)
主な応用	論理設計	データマイニング
データ構造	BDD （二分決定グラフ）	ZDD （ゼロサプレス型BDD）
基本演算	論理演算: AND/OR/EXOR	集合演算: Intersection/ Union / Difference

BDDとZDDの比較

- Many of real-life problems are likely **asymmetric**.

(Asymmetric world)

- Data mining & Knowledge discovery
- Market data analysis
- Web data analysis
- Formal concept analysis
- Risk analysis
- Calculation with Bayesian networks
- Machine learning & clustering

etc.

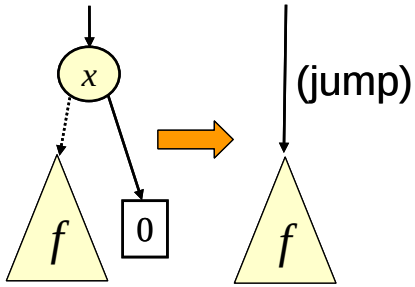
(Symmetric world)

- VLSI Logic Design
- Formal Verification
- etc.

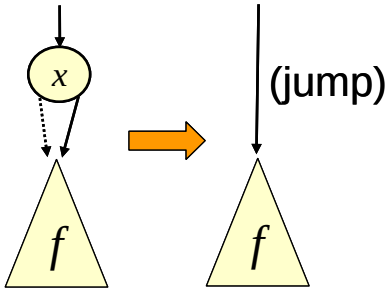
ZDD

BDD

Asymmetric reduction

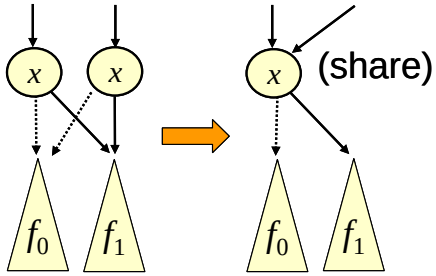


Symmetric reduction

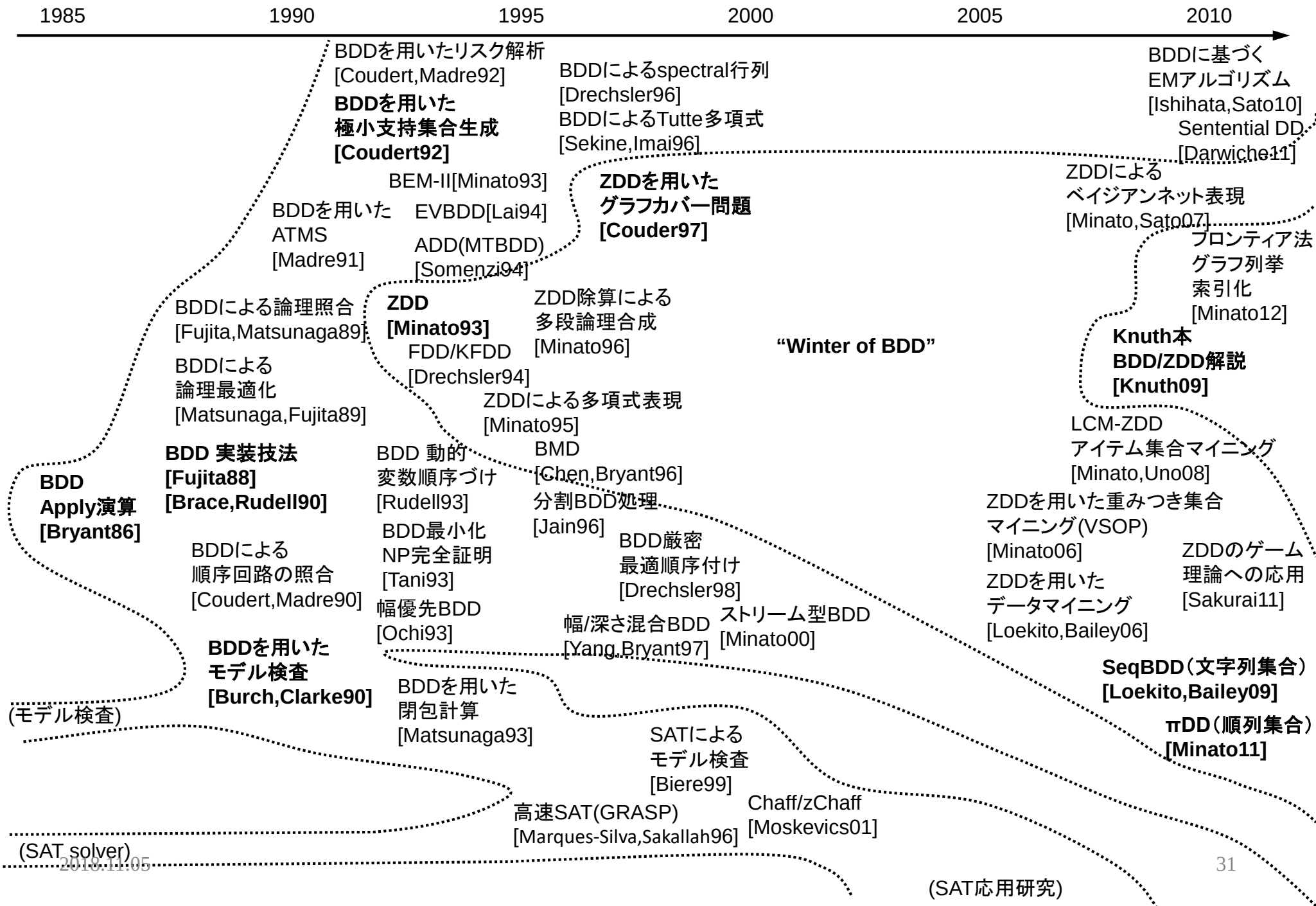


Quasi-reduced BDD

Sub-graph sharing (Dictionary-based)



Naïve representation


1985
1990
1995
2000
2005
2010
BDDを用いたリスク解析
[Coudert,Madre92]
BDDを用いた
極小支持集合生成
[Coudert92]
BDDによるspectral行列
[Drechsler96]
BDDによるTutte多項式
[Sekine,Imai96]
BDDに基づく
EMアルゴリズム
[Ishihata,Sato10]
Sentential DD
[Darwiche11]
BEM-II[Minato93]
ZDDを用いた
グラフカバー問題
[Couder97]
ZDDによる
ベイジアンネット表現
[Minato,Sato07]
BDDを用いた
ATMS
[Madre91]
EVBDD[Lai94]
ADD(MTBDD)
[Somenzi94]
プロンティア法
グラフ列挙
索引化
[Minato12]
BDDによる論理照合
[Fujita,Matsunaga89]
ZDD
[Minato93]
ZDD除算による
多段論理合成
[Minato96]
“Winter of BDD”
Knuth本
BDD/ZDD解説
[Knuth09]
FDD/KFDD
[Drechsler94]
BDDによる
論理最適化
[Matsunaga,Fujita89]
ZDDによる多項式表現
[Minato95]
LCM-ZDD
アイテム集合マイニング
[Minato,Uno08]
BDD 実装技法
[Fujita88]
[Brace,Rudell90]
BDD 動的
変数順序づけ
[Rudell93]
BMD
[Chen,Bryant96]
分割BDD処理
[Jain96]
BDD
Apply演算
[Bryant86]
ZDDを用いた重みつき集合
マイニング(VSOP)
[Minato06]
BDD最小化
NP完全証明
[Tani93]
BDD厳密
最適順序付け
[Drechsler98]
ZDDのゲーム
理論への応用
[Sakurai11]
BDDによる
順序回路の照合
[Coudert,Madre90]
ZDDを用いた
データマイニング
[Loekito,Bailey06]
幅優先BDD
[Ochi93]
幅/深さ混合BDD
[Yang,Bryant97]
ストリーム型BDD
[Minato00]
BDDを用いた
モデル検査
[Burch,Clarke90]
SeqBDD（文字列集合）
[Loekito,Bailey09]
BDDを用いた
閉包計算
[Matsunaga93]
(モデル検査)
πDD（順列集合）
[Minato11]
SATによる
モデル検査
[Biere99]
高速SAT(GRASP)
[Marques-Silva,Sakallah96]
Chaff/zChaff
[Moskevics01]
(SAT solver)
(SAT応用研究)

LSI設計分野へのBDDの応用

1985 1990 1995 2000 2005 2010

BDDを用いたリスク解析 [Coudert,Madre92]

BDDを用いた極小支持集合生成 [Coudert92]

BDDによるspectral行列 [Drechsler96]

BDDによるTutte多項式 [Sekine,Imai96]

BDDに基づくEMアルゴリズム [Ishihata,Sato10]

Sentential DD [Darwiche11]

BEM-II[Minato93]

ZDDを用いたグラフカバー問題 [Couder97]

ZDDによるベイジアンネット表現 [Minato,Sato07]

BDDを用いたATMS [Madre91]

EVBDD[Lai94]

ADD(MTBDD) [Somenzi94]

プロンティア法グラフ列挙索引化 [Minato12]

BDDによる論理照合 [Fujita,Matsunaga89]

ZDD [Minato93]

ZDD除算による多段論理合成 [Minato96]

"Winter of BDD"

Knuth本 BDD/ZDD解説 [Knuth09]

BDDによる論理最適化 [Matsunaga,Fujita89]

FDD/KFDD [Drechsler94]

ZDDによる多項式表現 [Minato95]

LCM-ZDD アイテム集合マイニング [Minato,Uno08]

BDD 実装技法 [Fujita88] [Brace,Rudell90]

BDD 動的変数順序づけ [Rudell93]

BMD [Chen,Bryant96]

分割BDD処理 [Jain96]

BDD Apply演算 [Bryant86]

ZDDを用いた重みつき集合マイニング(VSOP) [Minato06]

BDD最小化NP完全証明 [Tani93]

BDD厳密最適順序付け [Drechsler98]

ZDDのゲーム理論への応用 [Sakurai11]

BDDによる順序回路の照合 [Coudert,Madre90]

幅優先BDD [Ochi93]

幅/深さ混合BDD [Yang,Bryant97]

ストリーム型BDD [Minato00]

ZDDを用いたデータマイニング [Loekito,Bailey06]

BDDを用いたモデル検査 [Burch,Clarke90]

SeqBDD(文字列集合) [Loekito,Bailey09]

(モデル検査)

BDDを用いた閉包計算 [Matsunaga93]

πDD(順列集合) [Minato11]

SATによるモデル検査 [Biere99]

高速SAT(GRASP) [Marques-Silva,Sakallah96]

Chaff/zChaff [Moskevics01]

(SAT solver)

(SAT応用研究)

1985 1990 1995 2000 2005 2010

BDDを用いたリスク解析
[Coudert,Madre92]

BDDを用いた
極小支持集合生成
[Coudert92]

BDDによるspectral行列
[Drechsler96]
BDDによるTutte多項式
[Sekine,Imai96]

BDDに基づく
EMアルゴリズム
[Ishihata,Sato10]

Sentential DD
[Darwiche11]

BEM-II[Minato93]

ZDDを用いた
グラフカバー問題
[Couder97]

ZDDによる
ベイジアンネット表現
[Minato,Sato07]

BDDを用いた
ATMS
[Madre91]

EVBDD[Lai94]

ADD(MTBDD)
[Somenzi94]

プロンティア法
グラフ列挙
索引化
[Minato12]

BDDによる論理照合
[Fujita,Matsunaga89]

ZDD
[Minato93]

FDD/KFDD
[Drechsler94]

ZDD除算による
多段論理合成
[Minato96]

“Winter of BDD”

Knuth本
BDD/ZDD解説
[Knuth09]

BDDによる
論理最適化
[Matsunaga,Fujita89]

ZDDによる多項式表現
[Minato95]

LCM-ZDD
アイテム集合マイニング
[Minato,Uno08]

BDD 実装技法
[Fujita88]
[Brace,Rudell90]

BDD 動的
変数順序づけ
[Rudell93]

BMD
[Chen,Bryant96]

分割BDD処理
[Jain96]

BDD
Apply演算
[Bryant86]

ZDDを用いた重みつき集合
マイニング(VSOP)
[Minato06]

BDD最小化
NP完全証明
[Tani93]

BDD厳密
最適順序付け
[Drechsler98]

ZDDのゲーム
理論への応用
[Sakurai11]

BDDによる
順序回路の照合
[Coudert,Madre90]

ZDDを用いた
データマイニング
[Loekito,Bailey06]

幅優先BDD
[Ochi93]

幅/深さ混合BDD
[Yang,Bryant97]

ストリーム型BDD
[Minato00]

BDDを用いた
モデル検査
[Burch,Clarke90]

SeqBDD（文字列集合）
[Loekito,Bailey09]

BDDを用いた
閉包計算
[Matsunaga93]

(モデル検査)

πDD（順列集合）
[Minato11]

SATによる
モデル検査
[Biere99]

高速SAT(GRASP)
[Marques-Silva,Sakallah96]

Chaff/zChaff
[Moskevics01]

(SAT solver)

(SAT応用研究)

1985 1990 1995 2000 2005 2010

BDDを用いたリスク解析
[Coudert,Madre92]

BDDを用いた
極小支持集合生成
[Coudert92]

BDDによるspectral行列
[Drechsler96]
BDDによるTutte多項式
[Sekine,Imai96]

新しい
応用分野の
広がり

BDDに基づく
EMアルゴリズム
[Ishihata,Sato10]

Sentential DD
[Darwiche11]

BEM-II[Minato93]

ZDDを用いた
グラフカバー問題
[Couder97]

ZDDによる
ベイジアンネット表現
[Minato,Sato07]

BDDを用いた
ATMS
[Madre91]

EVBDD[Lai94]

ADD(MTBDD)
[Somenzi94]

プロンティア法
グラフ列挙
索引化
[Minato12]

BDDによる論理照合
[Fujita,Matsunaga89]

ZDD
[Minato93]

FDD/KFDD
[Drechsler94]

ZDD除算による
多段論理合成
[Minato96]

“Winter of BDD”

Knuth本
BDD/ZDD解説
[Knuth09]

BDDによる
論理最適化
[Matsunaga,Fujita89]

ZDDによる多項式表現
[Minato95]

LCM-ZDD
アイテム集合マイニング
[Minato,Uno08]

BDD 実装技法
[Fujita88]
[Brace,Rudell90]

BDD 動的
変数順序づけ
[Rudell93]

BMD
[Chen,Bryant96]

分割BDD処理
[Jain96]

BDD
Apply演算
[Bryant86]

ZDDを用いた重みつき集合
マイニング(VSOP)
[Minato06]

BDDによる
順序回路の照合
[Coudert,Madre90]

BDD最小化
NP完全証明
[Tani93]

BDD厳密
最適順序付け
[Drechsler98]

ZDDのゲーム
理論への応用
[Sakurai11]

ZDDを用いた
データマイニング
[Loekito,Bailey06]

幅優先BDD
[Ochi93]

幅/深さ混合BDD
[Yang,Bryant97]

ストリーム型BDD
[Minato00]

BDDを用いた
モデル検査
[Burch,Clarke90]

SeqBDD（文字列集合）
[Loekito,Bailey09]

BDDを用いた
閉包計算
[Matsunaga93]

(モデル検査)

SATによる
モデル検査
[Biere99]

πDD（順列集合）
[Minato11]

高速SAT(GRASP)
[Marques-Silva,Sakallah96]

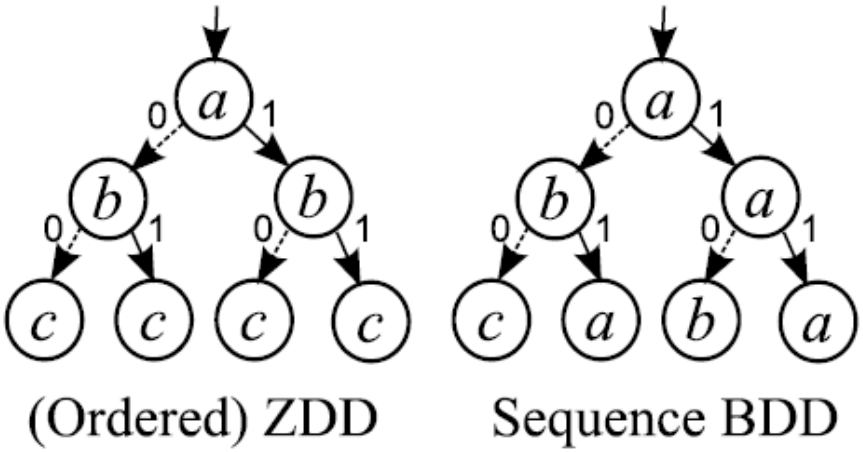
Chaff/zChaff
[Moskevics01]

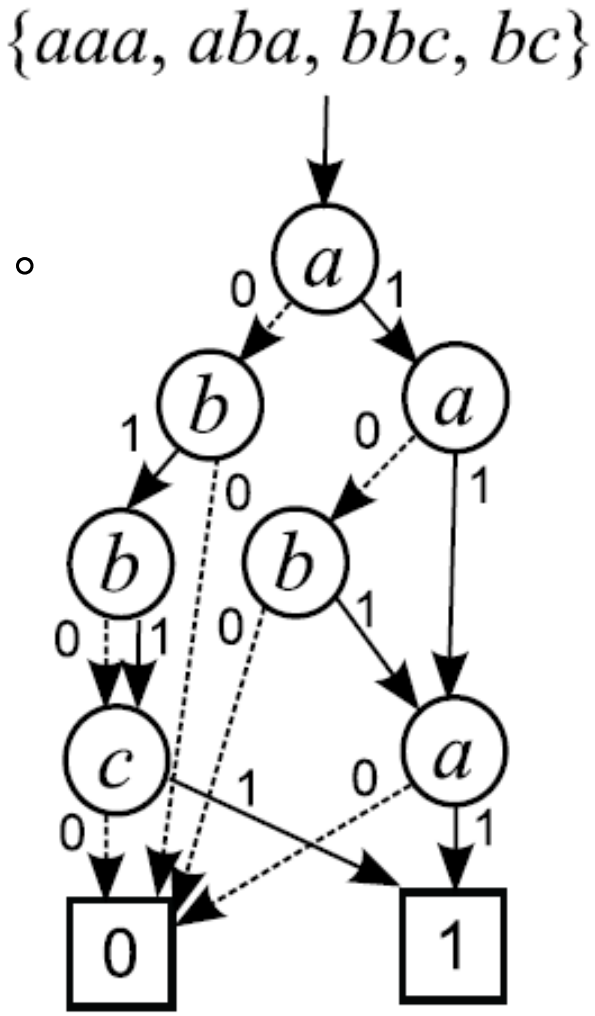
(SAT solver)

(SAT応用研究)

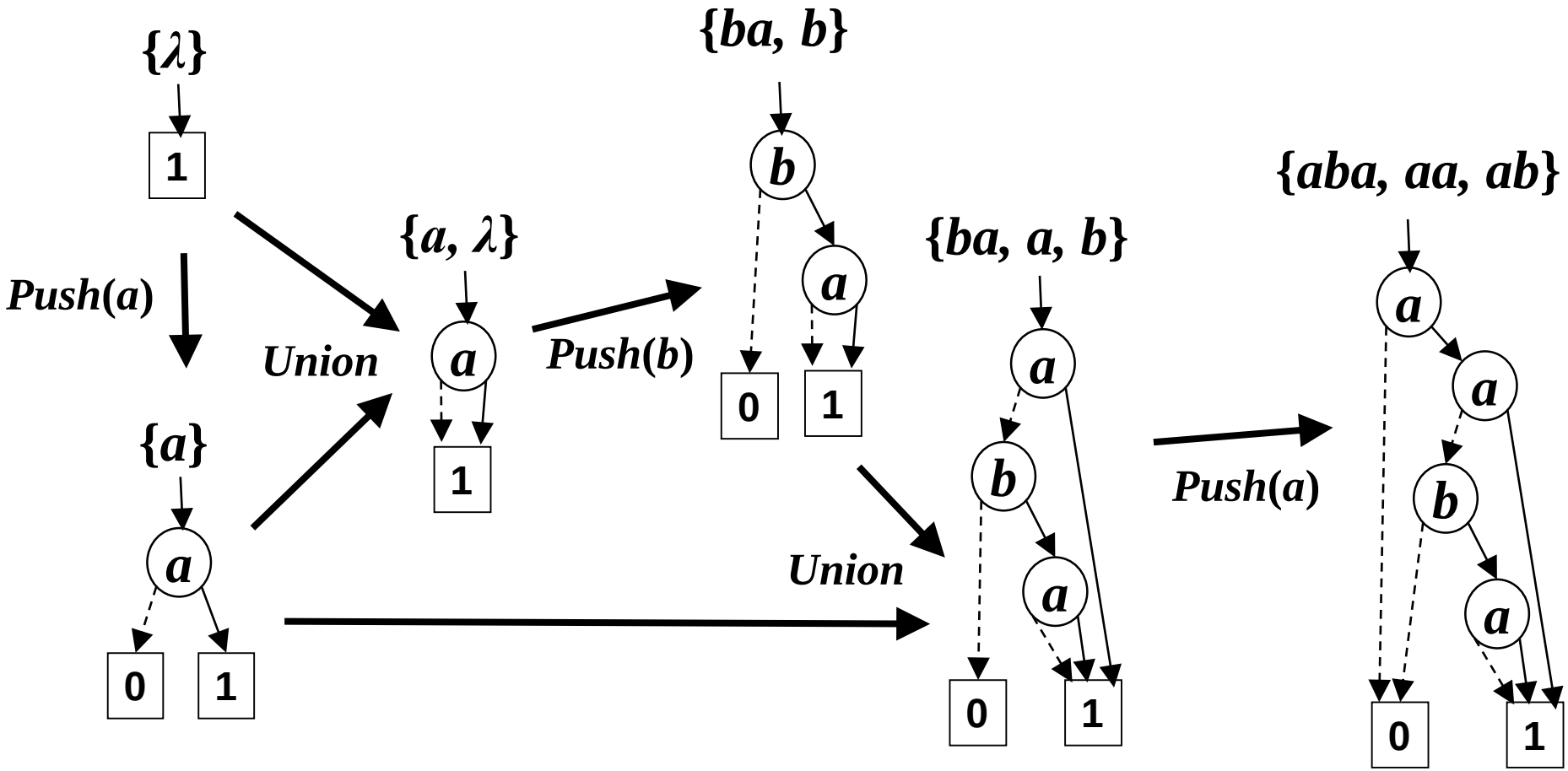
文字列集合を表す Sequence BDD (SeqBDD)

- Loekito, Bailey, and Pei [2009]
  - 簡約化規則はZDDと同じ.
  - **0-枝のサイドだけ変数順序を守る。**
  - 1-枝のサイドの変数順序は自由（重複も可）。
  - 文字列集合を一意に表現できる。
  - 根節点から1-終端節点へ至る経路が、集合要素の文字列に対応する。



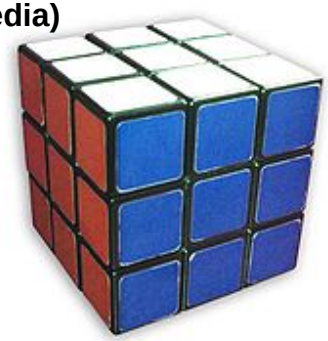


SeqBDDの構築（基本演算の適用）



順列集合の処理

(© Wikipedia)

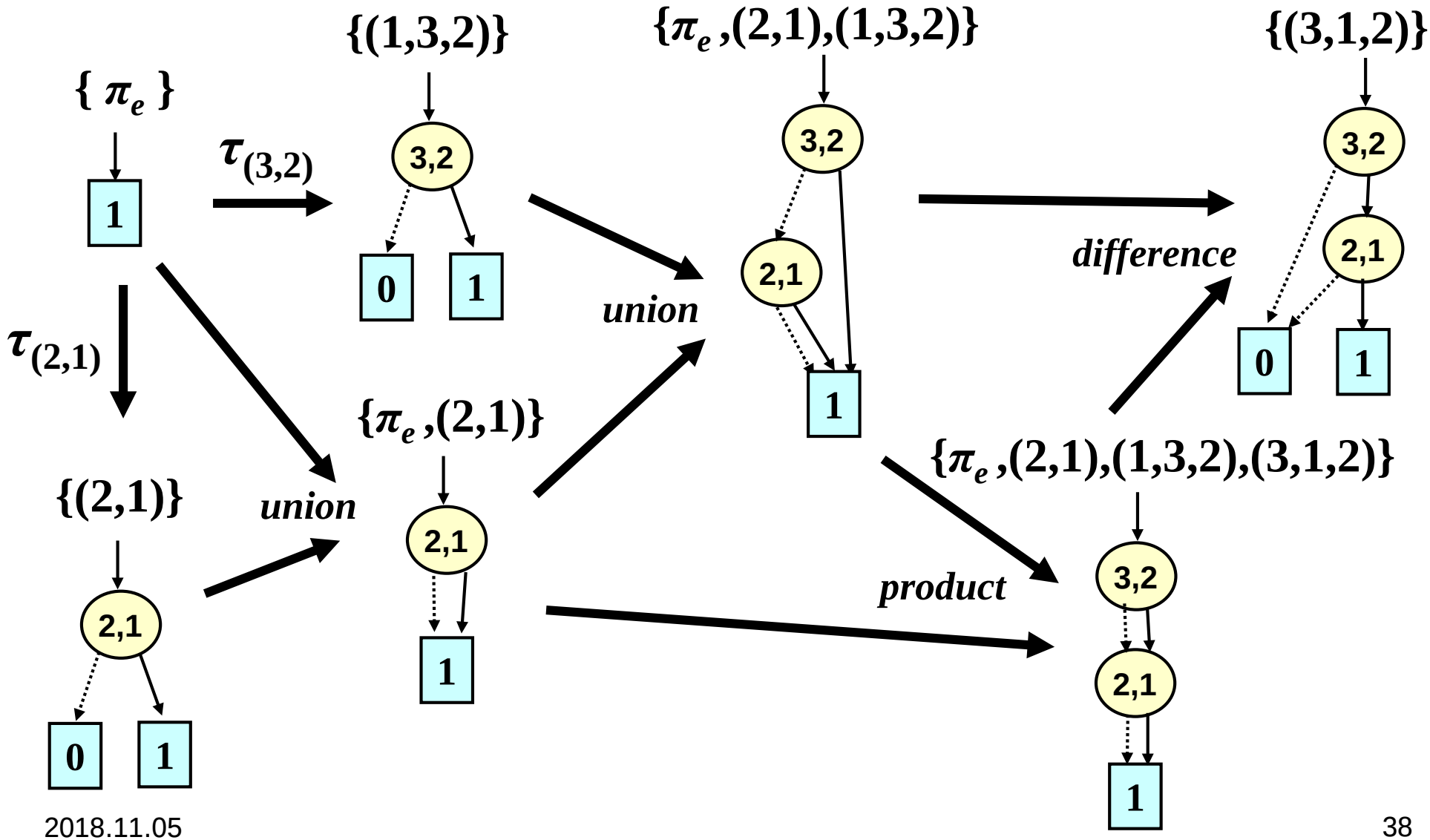


- ルービックキューブ:
  $P = \{ \pi \mid$ 基本操作による順列$\}$
  - P は18通り (= 3通り×6面) の順列の集合
  - 直積演算$P \times P$ は、2回の基本操作で得られるパタンの集合を表す。
  - P^{20} で原理的には全ての可能なパタンの集合が得られる。
    （実際は場合の数が多すぎてメモリあふれが起きる）
- 15パズル・トランプ等も順列集合で表現される
  - 最適な操作系列の解析

(© Wikipedia)

- あみだくじ・ソーティングネットワーク
  - 1対1のマッチング問題に適用可能
  - 順列は単射写像に対応。群論と深い関係がある。
- 情報無損失符号の設計・暗号理論への応用
  - Reversible logic設計問題への応用（量子計算とも関連）

順列集合を表すπDDの構築(基本演算の適用)



BDD/ZDDから高度なDDへ

現在の
ZDD

基本的
成果

0,1に関して非対称な世界への適用拡大

データマイニング 機械学習

高度な検索 etc.

ZDDが有効な応用
分野が、まだまだ多く
残されている。

(組合せ集合モデル)

(より高次のモデル)

- ワード単位データ
- 文字列
- 順序付け
- 分割
- ツリー
- ネットワーク

より高次の成果

高度な
ZDD風の
離散構造

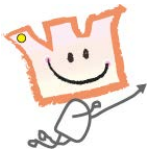
新しい演算
体系の構築

より高次のデータモデルを用いた実用的問題への応用

文字列データ解析

数値データ処理

ツリーや半構造データの解析



ERATO/基盤(S)の活動経過とロードマップ

2009.10 2010.4 2010.10 2011.4 2011.10 2012.4 2012.10 2013.4 2013.10 2014.4 2014.10 2015.4

(0年目) (1年目) (2年目) (3年目) (4年目) (5年目)

採択
GL委嘱
交渉
研究員
スカウト
オフィス選定
内装工事

キックオフWS
秋の合宿
FIT2010
情処企画
企画シンポ
Knuth邸訪問
各種研究会
国際会議共催
民間企業との共同研究
研究員出向交渉

初夏のWS
秋の合宿
講究録出版
通信学会企画
各種研究会シンポ
CMU共催
ALSIP
訪問
ALSIP 2011
電力網報道発表

初夏のWS
秋の合宿
講究録出版
公開シンポ
AI学会特集号
未来館展示
ALSIP 2012
東工大集中講義

初夏のWS
秋の合宿
NII講演
FIT2013
シンポ審査報告会
RM 2013
北大博物館展示
新検定法報道発表

春のWS
秋の合宿
閉所式
NTT研講演
ALSIP特別セミナー
RC 2014
ALSIP 2014
最終報告会
札幌科学館展示
未来館再展示

北大連携講座

後継プロジェクト

2015.4 2015.10 2016.4 2016.10 2017.4 2017.10 2018.4 2018.10 2019.4 2019.10 2020.4

(6年目/1年目) (2年目) (3年目) (4年目) (5年目)

基盤S採択
合同WS
初夏のWS
秋の合宿
専門書出版
未来館再展示
WS
企画シンポ
最終評価
AI学会講演

神田ラボキックオフ
合同WS
初夏のWS
秋の合宿
ISMVL講演
統計応用報道発表
オフィス移転

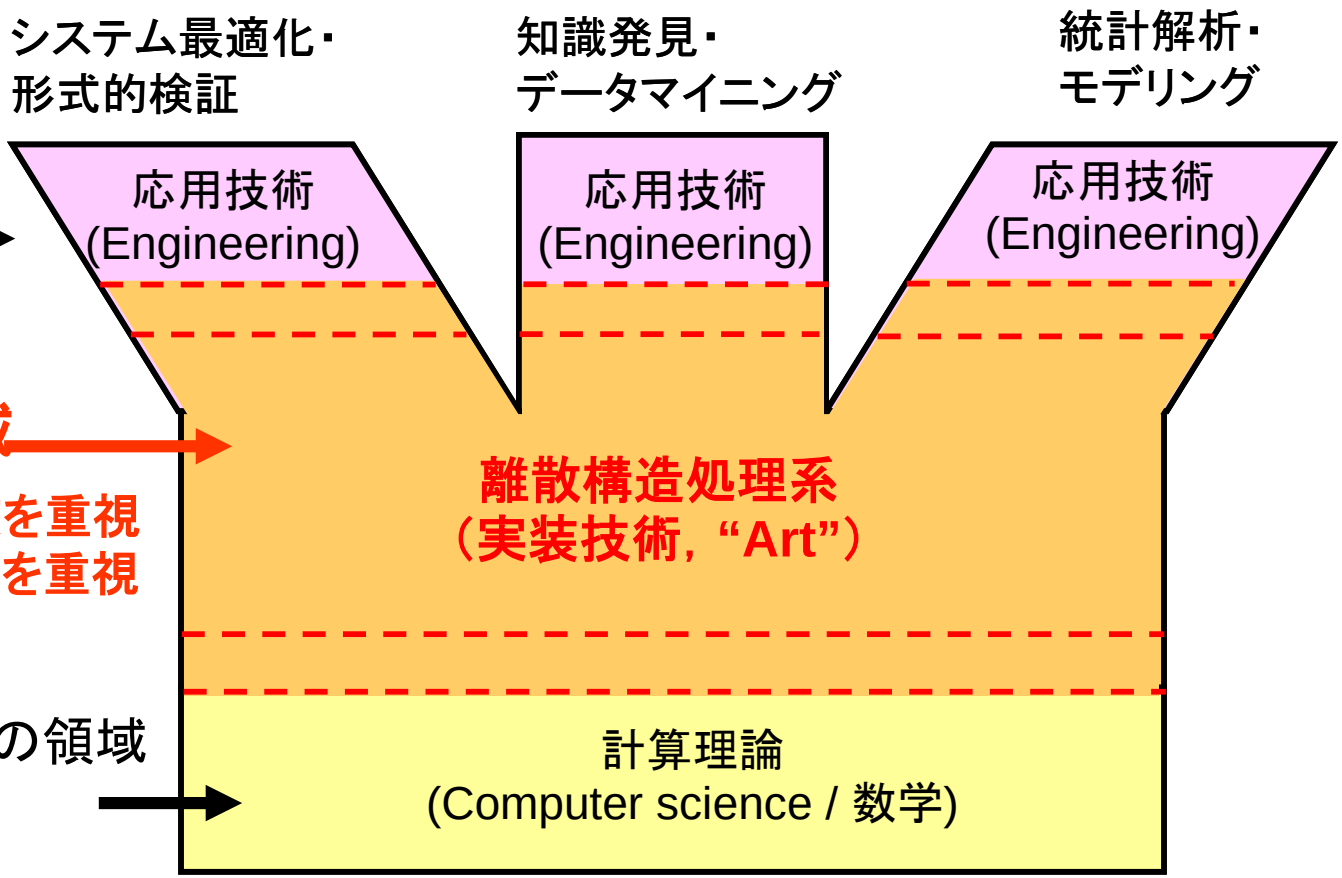
初夏のWS
SoCS講演
秋の合宿
IFCS講演
Dagstuhl講演
FPAI共催
EPFL講演
さきがけ合同WS
合同WS
2018.11.05

研究進捗提出書
JSPS進捗評価
初夏のWS
WEPA共催
JSAI OS共催
SSSW開催
代表者京大移籍
北大WPI
秋の合宿

最終評価
東京五輪

ERATO/基盤(S)「離散構造処理系」の研究構想

Art層の周辺部では後継プロジェクトへの発展に成功しつつある．ボトムアップな科研費で Art層のコア部分に研究者が集まる「場」を継続的に提供し**，競争力の源泉となるアイデアを醸成し続ける**

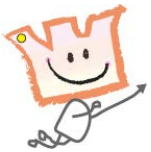


個別の工学的応用に特化した技術領域 →

本研究が扱う技術領域 →

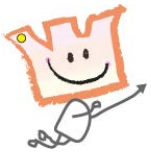
- 概念・理論だけでなく処理系実装を重視
- 技術基盤としての簡潔さ・汎用性を重視

分野横断的な計算理論の領域（概念的・理論的） →



本講演の内容

- 離散構造処理とは
- Decision Diagramによる論理や集合の表現と演算
  - BDDとZDDの基本概念と技術発展の流れ
  - DDの様々な派生形
- DDによるグラフ部分構造の列挙と解析
  - フロンティア法の基本的考え方
  - 社会的に重要な問題への応用
- 最近の研究トピックから
  - 列挙・数え上げの技法に基づく
    確率・統計・データサイエンス分野への応用



未来館での成果展示「フカシギの数え方」

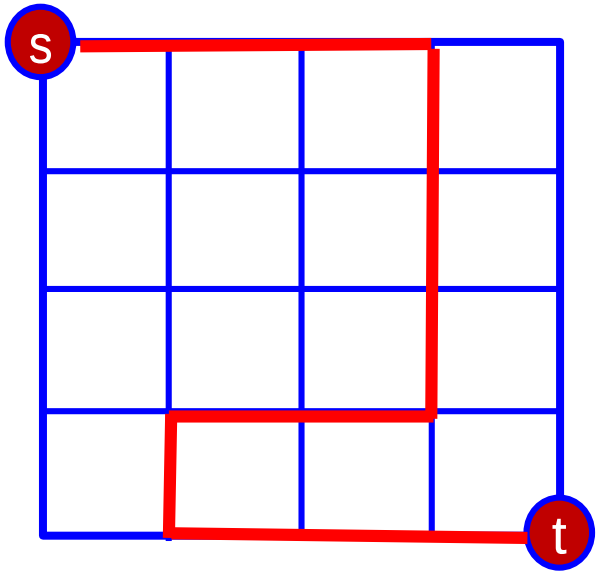
ERATOプロジェクトの
研究成果展示（2012年）
（日本科学未来館で開催）

展示作品の1つとして制作した
アニメーション動画がブレイク

YouTube未来館チャンネルで
現在**230万 ビューを突破**

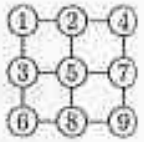
“seif-avoiding walks”の数え上げ

- 最短経路の数え上げは簡単
(→ ${}_{2n}C_n$；高校で習う問題)
- 最短でない経路を許すと突然難しくなる。
（計算式や漸化式は見つかっていない）
  - おねえさんが25万年かかった結果がアニメの中に表示されているため、「計算式教えて」というコメント多数。
  - 残念ながら計算式は知られていない。効率良く数え上げるしかない。



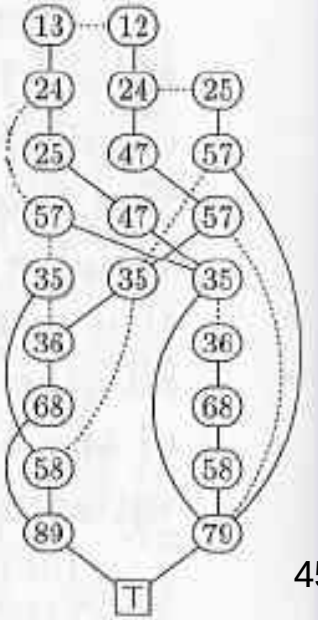
Knuthの教科書の記述

We can also use ZDDs to represent simple paths in an *undirected* graph. For example, there are 12 ways to go from the upper left corner of a 3×3 grid to the lower right corner, without visiting any point twice:



(132)

These paths can be represented by the ZDD shown at the right, which characterizes all sets of suitable edges. For example, we get the first path by taking the HI branches at (13), (36), (68), and (89) of the ZDD. (As in Fig. 28, this diagram has been simplified by omitting all of the uninteresting LO branches that merely go to $\boxed{\perp}$.) Of course this ZDD isn't a truly great way to represent (132), because that family of paths has only 12 members. But on the larger grid $P_8 \square P_8$, the number of simple paths from corner to corner turns out to be 789,360,053,252; and they can all be represented by a ZDD that has at most 33580 nodes. Exercise 225 explains how to construct such a ZDD quickly.

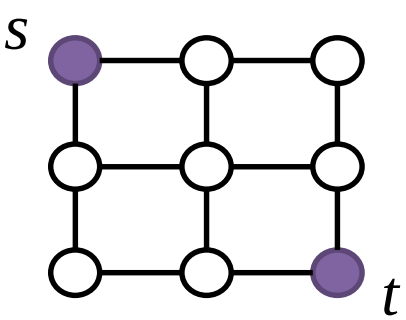


A similar algorithm, discussed in exercise 226, constructs a ZDD that represents all *cycles* of a given graph. With a ZDD of size 22275, we can deduce that $P_8 \square P_8$ has exactly 603,841,648,931 simple cycles.

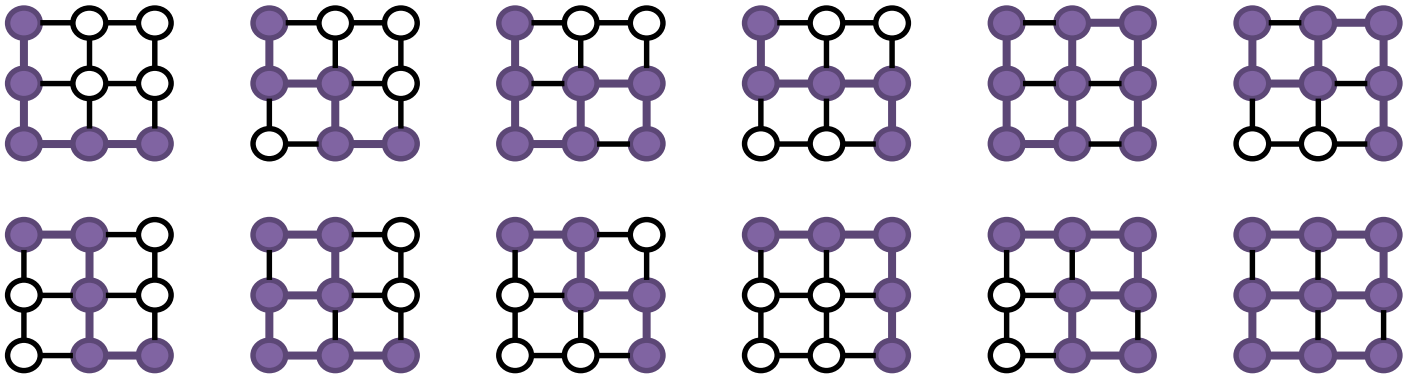
ZDDを用いたパスの列挙 [Knuth2009]

– Knuth本のZDDの項目に演習問題として記載

- 入力：グラフ$G=(V, E)$，2頂点 s, t
- 出力：同じ頂点を2度通らないs からt までのパス（枝の組合せ）を列挙したZDD



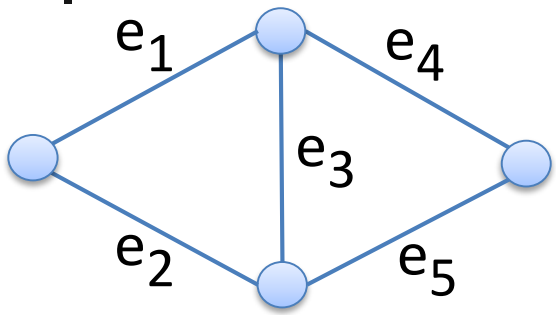
3×3格子グラフの例

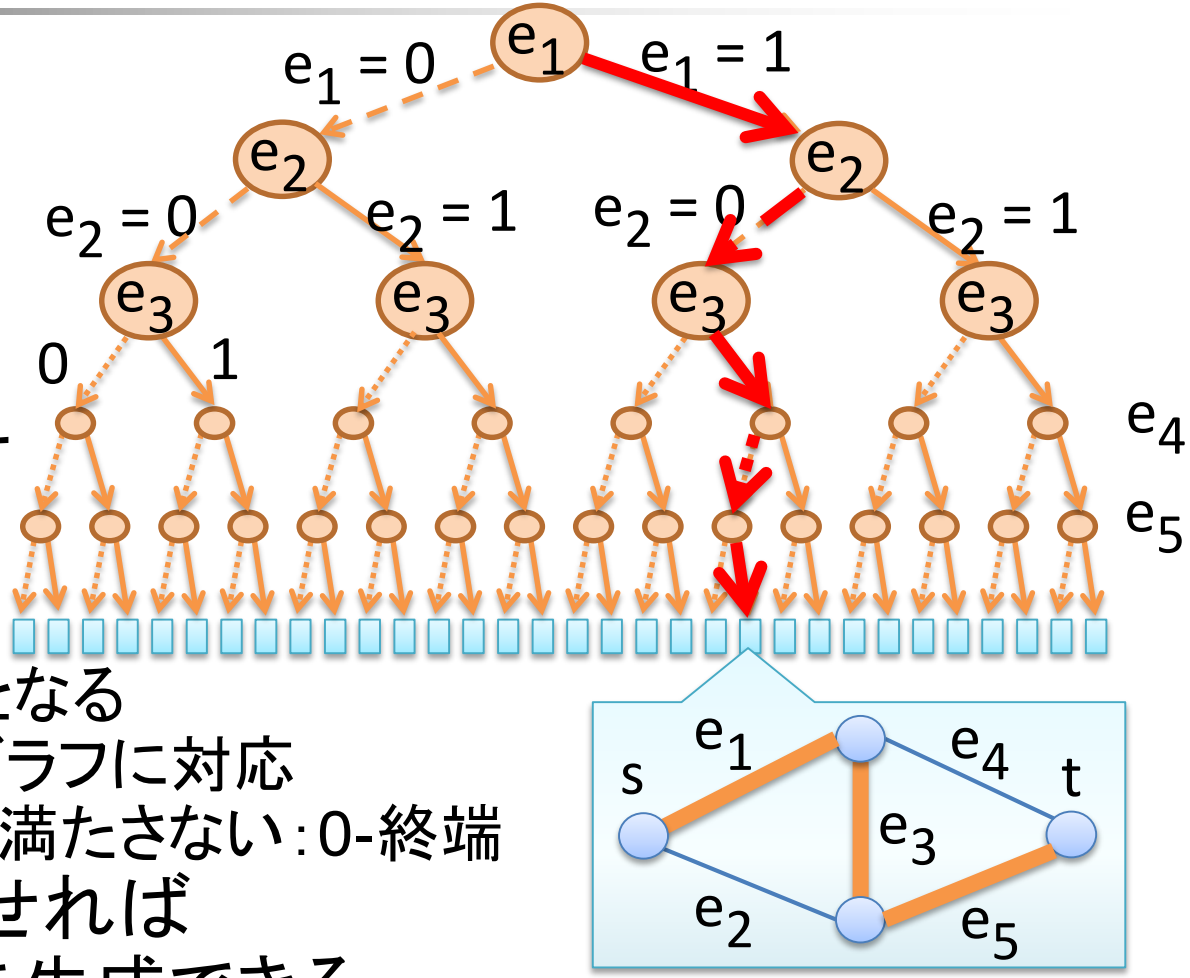


応用は極めて多い

地理情報処理、ネットワークや論理回路の信頼性評価
電力網制御（スマートグリッド等）、ソフトウェア解析、etc.

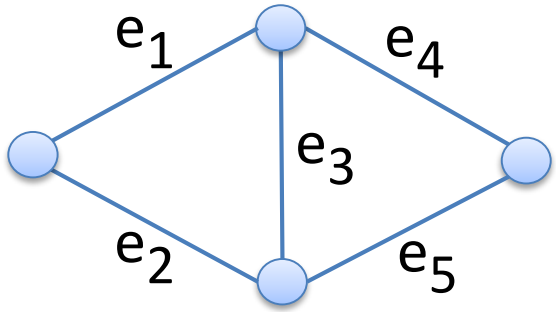
BDD/ZDDによるグラフの列挙

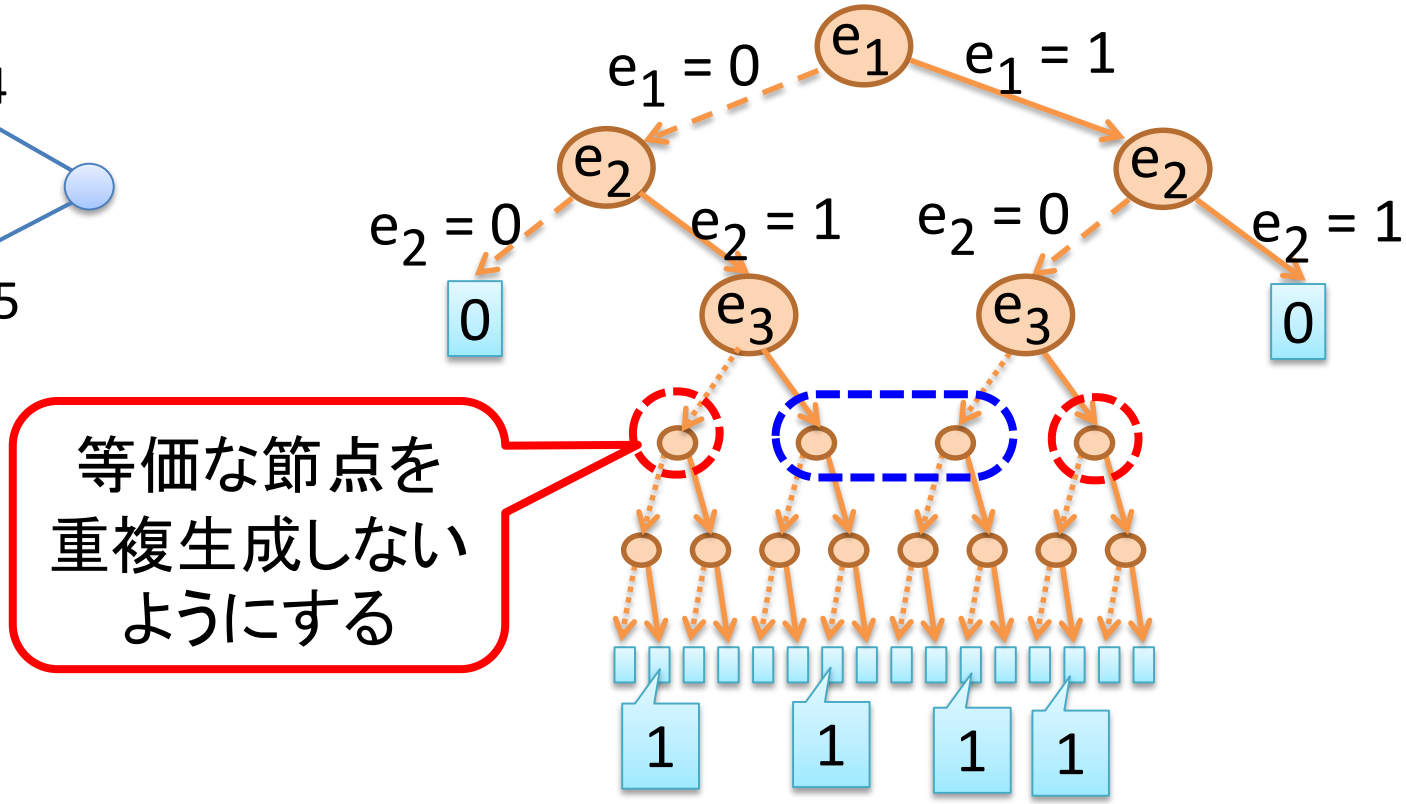




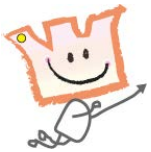
- 与えられた制約を満たすサブグラフ(枝の部分集合)を列挙する問題
  - 各枝の有無が入力変数となる
  - ZDD上での経路がサブグラフに対応
  - 制約を満たす:1-終端、 満たさない:0-終端
- 制約条件を論理式で表せればApply演算でBDD/ZDDを生成できる
  - 部分的に類似する組合せが多数発生する場合、圧縮率が高くなる
  - ある枝を使うと決めたら別の枝が使えなくなることが多い問題ではBDDよりも**ZDDが有利**

アルゴリズムの概略

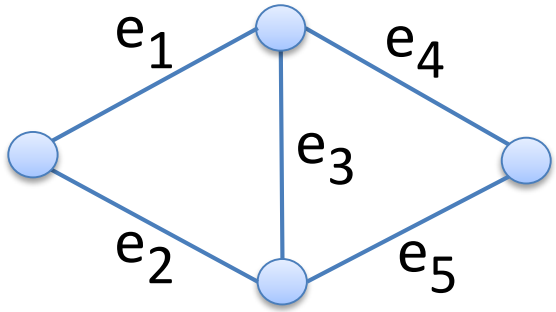


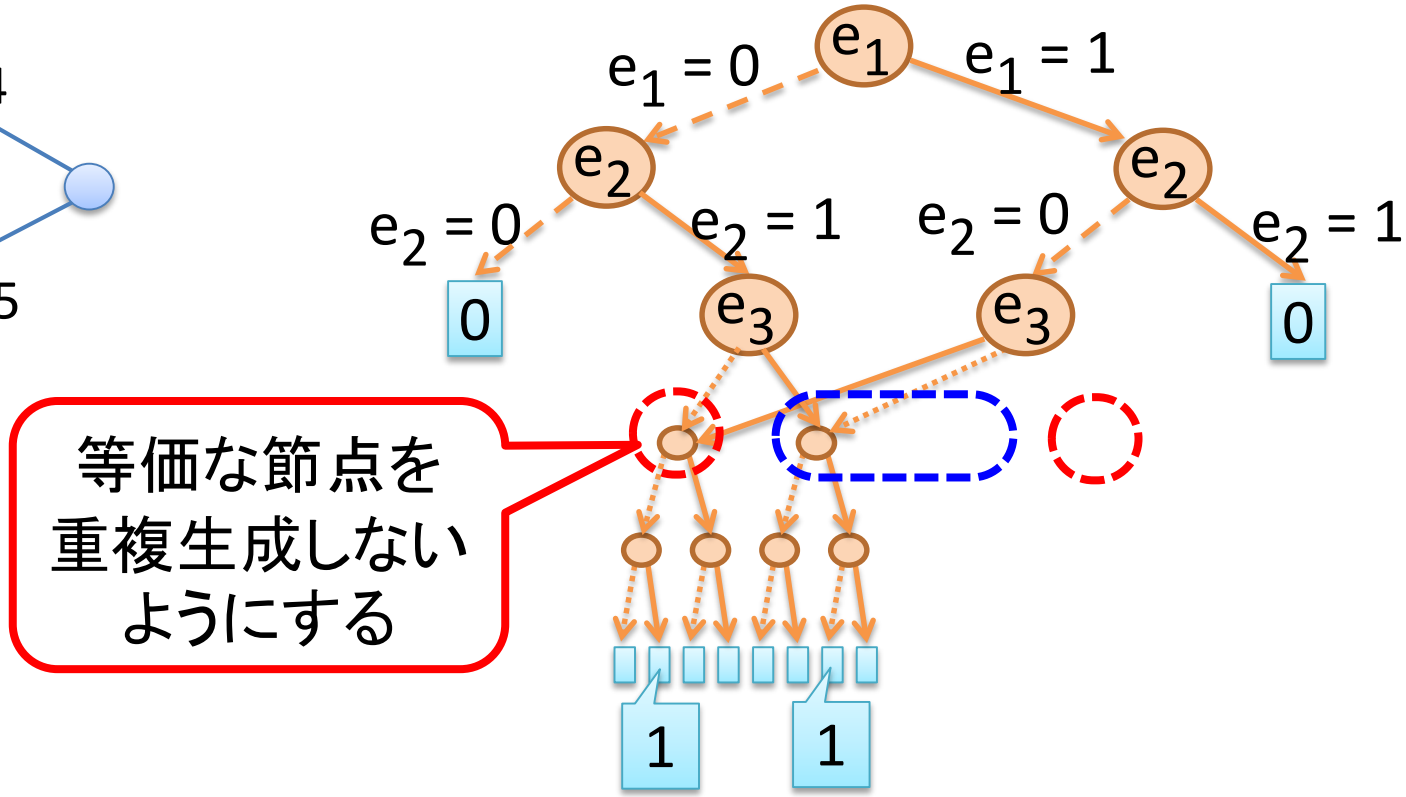


- 上位の枝から順に、枝の有無で場合分けしながら、幅優先で二分木を構築して行く
  - 途中で制約違反が見つかれば0で終端し、以降は分岐させない
  - 最後まで枝を決めて制約を満たすならば、1で終端

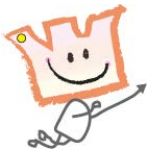


アルゴリズムの概略



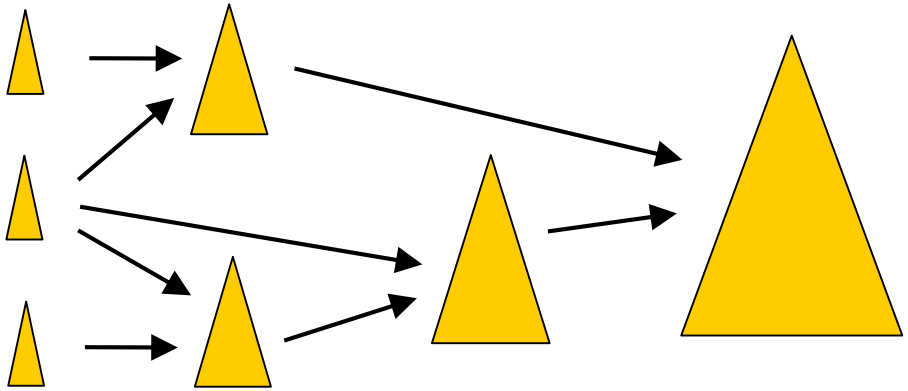


- 上位の枝から順に、枝の有無で場合分けしながら、幅優先で二分木を構築して行く
  - 途中で制約違反が見つかれば0で終端し、以降は分岐させない
  - 最後まで枝を決めて制約を満たすならば、1で終端

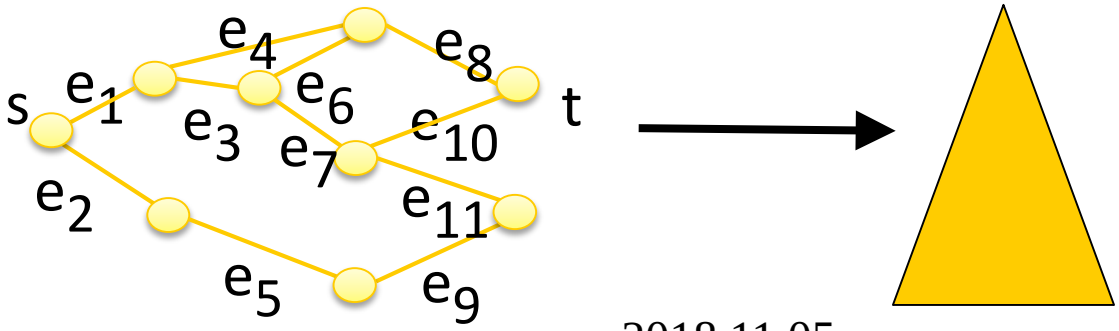


通常のZDD生成とフロンティア法の違い

- 通常の生成法：
  ZDD同士の集合演算を何度も適用し、所望のZDDを生成
  - 1回1回の集合演算は深さ優先順アルゴリズム

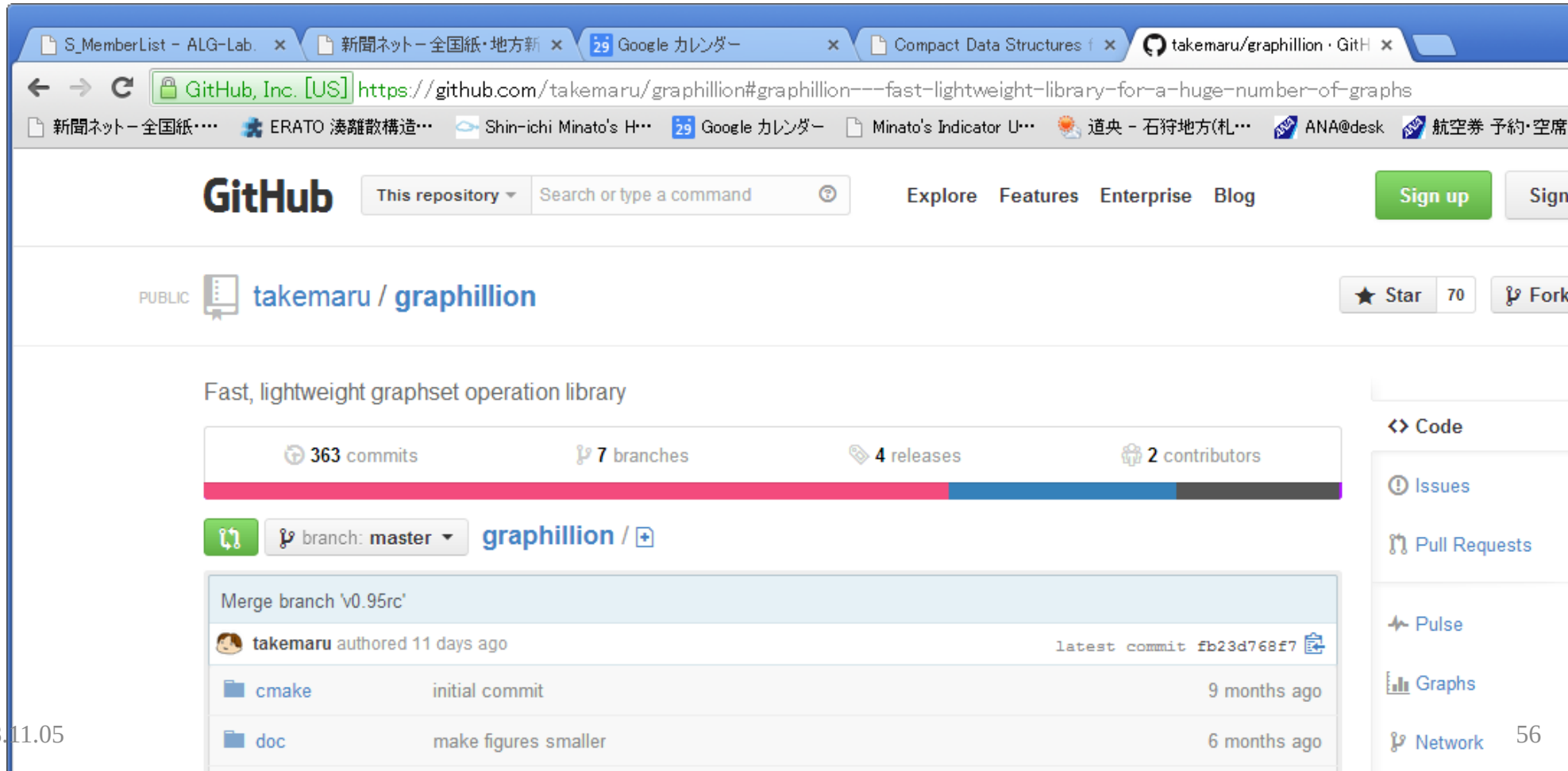


- **フロンティア法（Knuthのsimpath法を一般化・汎用化）**
  問題グラフの各辺を場合分けし、ZDDを一括生成
  - フロンティアを意識した動的計画法（幅優先順アルゴリズム）



オープンソフトウェア: “Graphillion.org”

- ZDDを用いた超高速グラフ列挙のツールボックス.
  - Pythonベースのライブラリとして提供



フロンティア法と電力網への応用

戦略的創造研究推進事業ERATO　湊離散構造処理系プロジェクト

特集2　配電網の最適経路を探索し、配電ロスを最小化へ

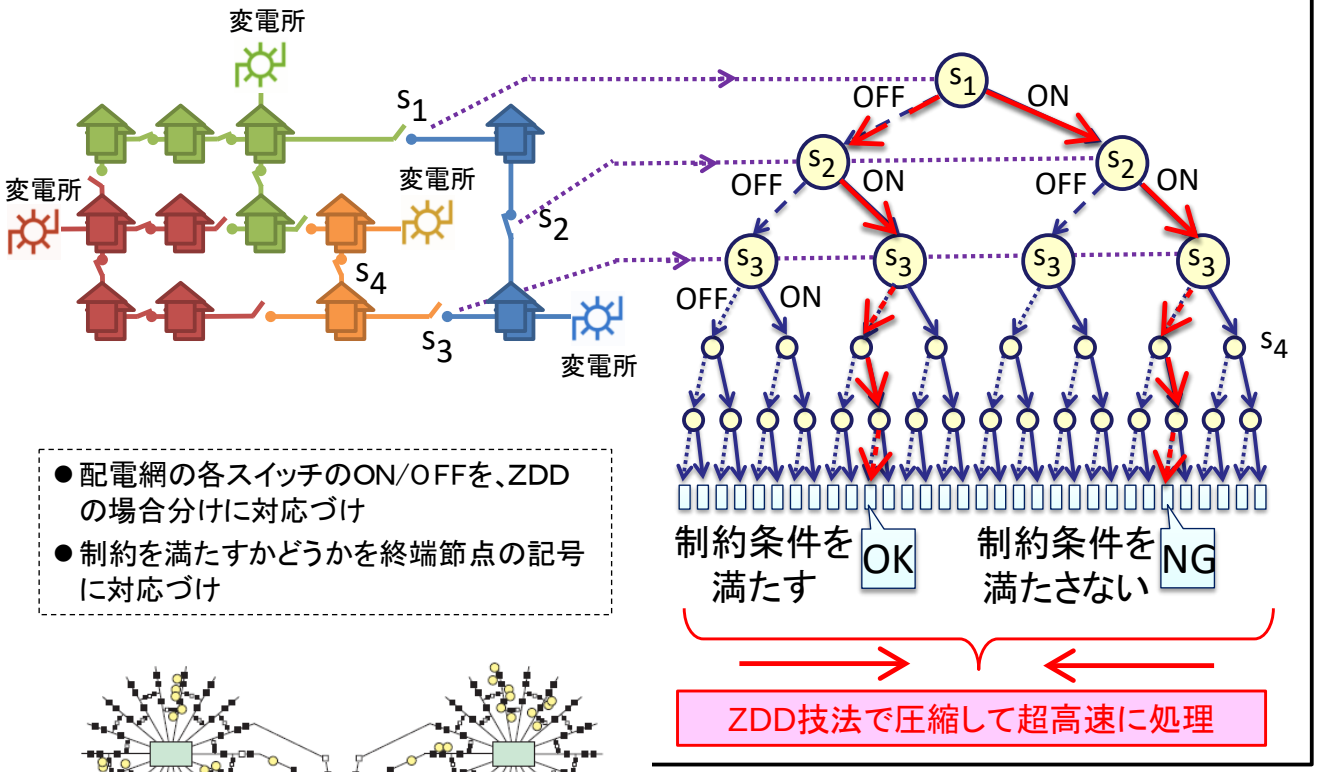
「超高速アルゴリズム」にできること

電気をムダなく効率よく送る「スマートグリッド」の実現には、最適な配電経路を求める技術的基盤も必要だ。ところが、日本の標準的な規模の配電網（人口数十万人規模）では、そのスイッチの組み合わせが10^{140}（10の140乗）通りもあると計算される。この中から最適な経路を見つけ出すのは至難の技だとされてきたが、北海道大学の湊真一教授らが開発した「超高速アルゴリズム」では、その解を得ることができるのだ。

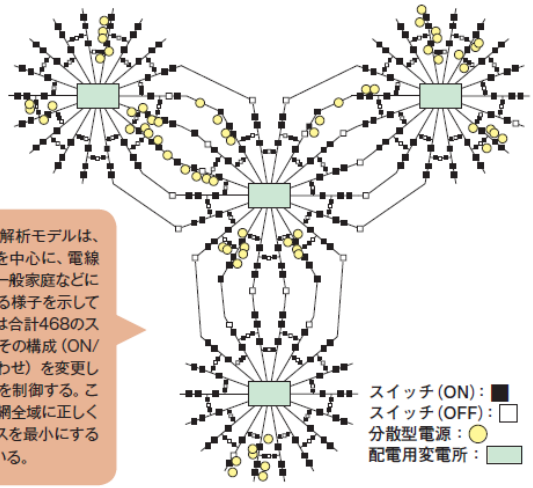
湊　真一
みなと・しんいち
北海道大学大学院情報科学研究科 教授
1965年生まれ。京都大学工学部情報工学科卒業、同大学大学院工学研究科情報工学専攻修了。博士（工学）。日本電信電話株式会社LSI研究所、米国スタンフォード大学客員研究員などを経て、2004年から北海道大学准教授、10年から現職。大規模論理データ処理、知識発引技術の研究に従事。09年からJST ERATO湊離散構造処理系プロジェクト研究総括。

ERATO湊離散構造処理系プロジェクトのサーバ室にて。ZDD、BDDなどの「超高速アルゴリズム技法」を使うには大容量のメモリが必要となるため、1台で1TB（テラバイト）のメモリを集中して搭載するサーバを使用している。今後さらに1TBの追加を計画中。

8 | JST NEWS May 2012



- 配電網の各スイッチのON/OFFを、ZDDの場合分けに対応づけ
- 制約を満たすかどうかを終端節点の記号に対応づけ

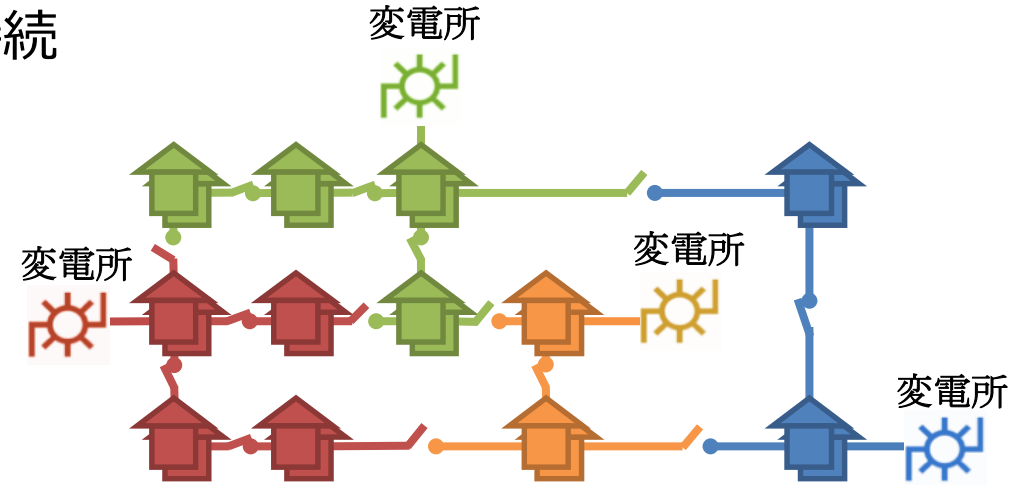


配電網の標準解析モデルは、4つの変電所を中心に、電線に接続された一般家庭などに電力を供給する様子を示している。電線には合計468のスイッチがあり、その構成（ON/OFFの組み合わせ）を変更して電力の流れを制御する。この図では配電網全域に正しく給電、送電ロスを最小にする構成を表している。

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

電力網のスイッチ制御

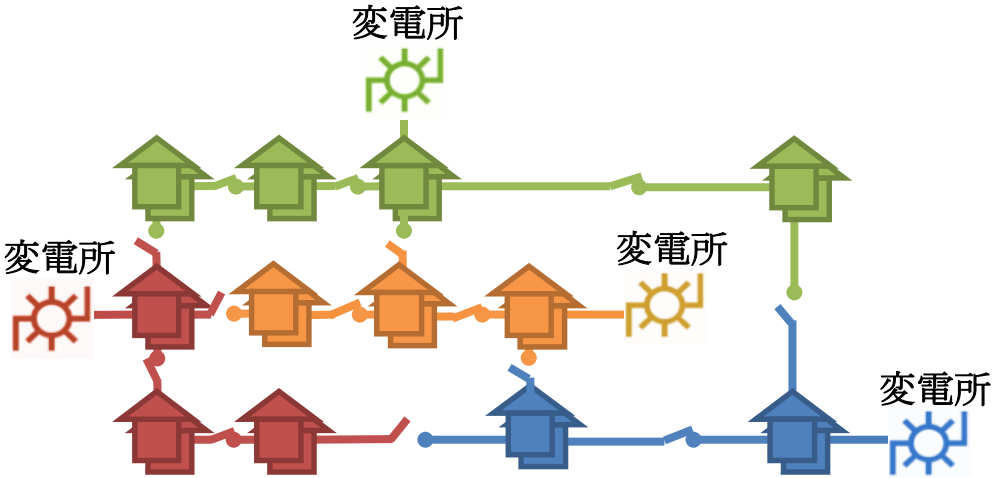
・各区域が必ず１ヶ所の変電所設備に接続
・停電しない
・異なる変電所系統をつながない
・電流が多過ぎると電線が焼ける
・遠くに伝えると電圧が下がる



膨大な数の構成が存在

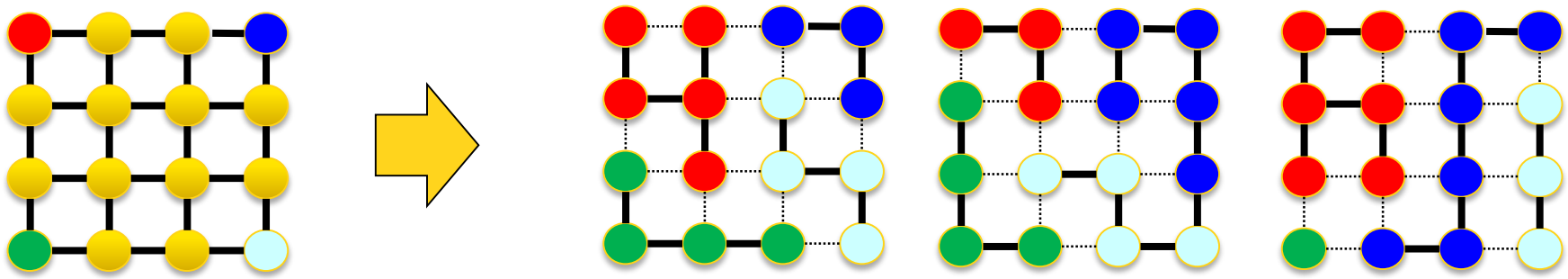
右のような14スイッチの小さな例題でも
16384通りの組合せの中から
210通りを探し出す問題

標準的な配電網の例では
468個のスイッチがあり、
10の140乗の組合せから
探し出す問題に



電力網のスイッチ制御

・ZDDを使った新しい高速アルゴリズムを開発。
・標準的な電力網モデル（スイッチ468個）でグラフ的制約と電気的制約を共に満たす解を全て求め、圧縮して表現することに成功
・電力の損失を最小にする組合せを求めることにも成功



圧縮データ:約110万個（779MB）　　実行時間:数十分

解の個数:約10^{63}**（2136那由他8201阿僧祇3834恒河沙8532極9116載8261正2214澗8049溝560穣9817杼8392垓4438京5235兆3981億8952万1540）通り**

電力網のほかにも様々な応用がある

・避難所の配置問題　・道路、鉄道、ガス、水道、通信
・選挙の区割り問題　・建物のフロアプラン

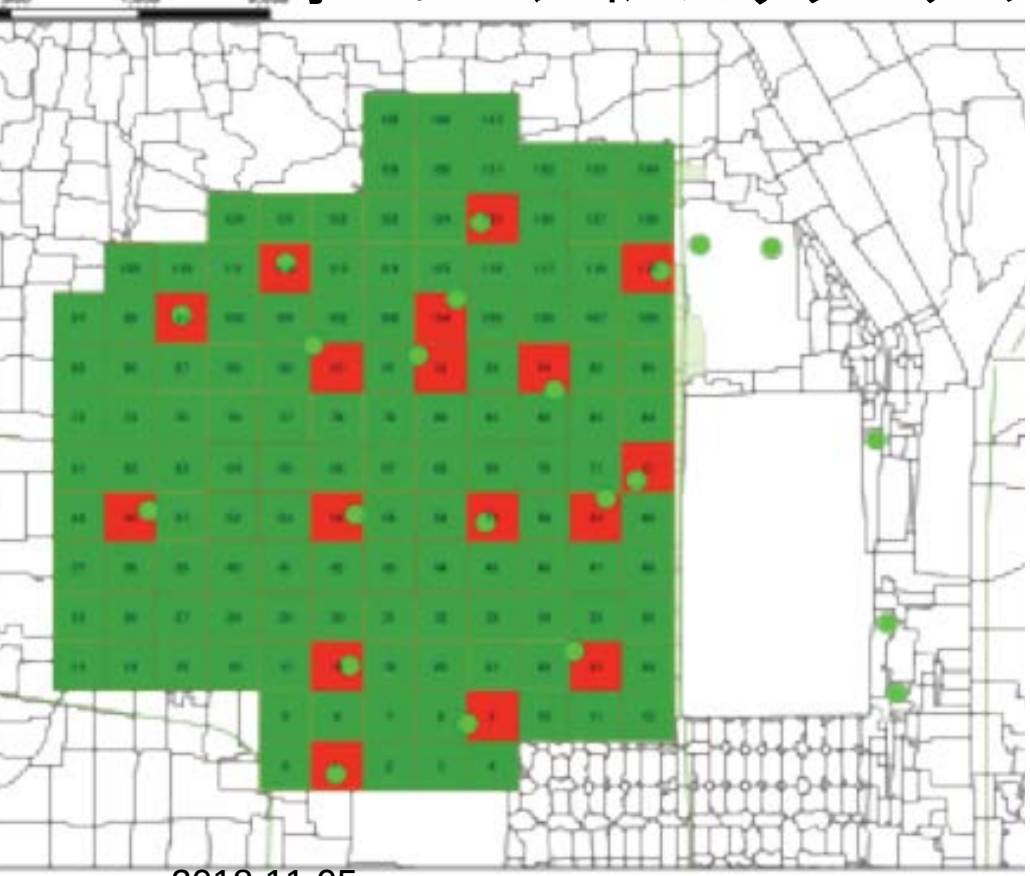
超高速グラフ列挙技術の実問題への応用

社会的に重要な様々な実問題に関係

- 有向/無向グラフのパス列挙:
  - 地理情報システム
  - 大規模システムの依存関係の解析、フローチャートの解析
  - ナンバーリンク、スリザーリンク等のパズル
  - 文字列の連接可能性の列挙
- グラフk分割問題:
  - 電力網の配電区割りの列挙(早大・林先生との共同研究)
  - 避難所の配置問題(京大・加藤先生との共同研究)
  - 建物フロアプランへの応用(大阪市大・瀧澤先生との共同研究)
  - 選挙区割り問題(文教大・堀田先生との共同研究)
- 地理情報、電力網、物流網のような社会インフラの構造は,平面グラフや格子グラフに近い形であることが多い。
  → ZDDの圧縮効果が極めて高くなる傾向がある。

避難所割り当て問題

- 京大建築系・加藤直樹先生他との共同研究
  - 京都市上京区の避難所割り当て問題に適用
  - オペレーションズリサーチの国際会議（ISORA 2013）で発表



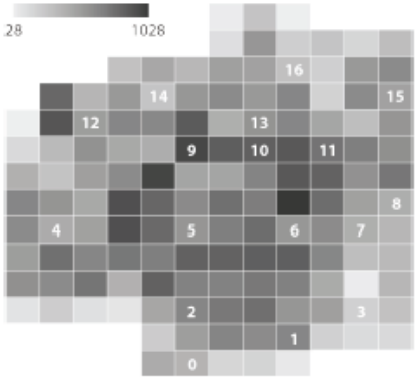


Figure 13: Number of population in each cell.

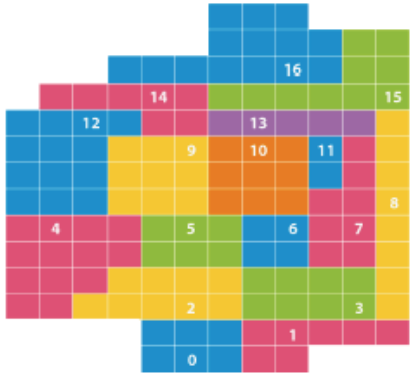


Figure 14: The optimal pattern with respect to *Criterion 1*.

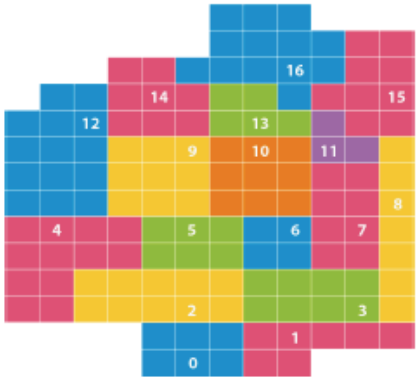


Figure 15: The optimal pattern with respect to *Criterion 2*.

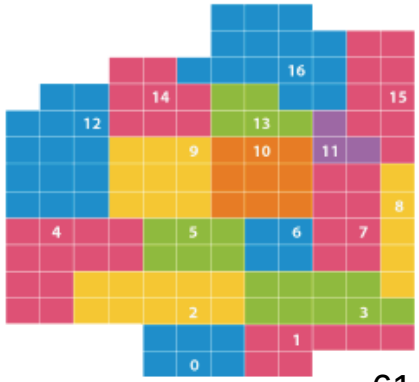
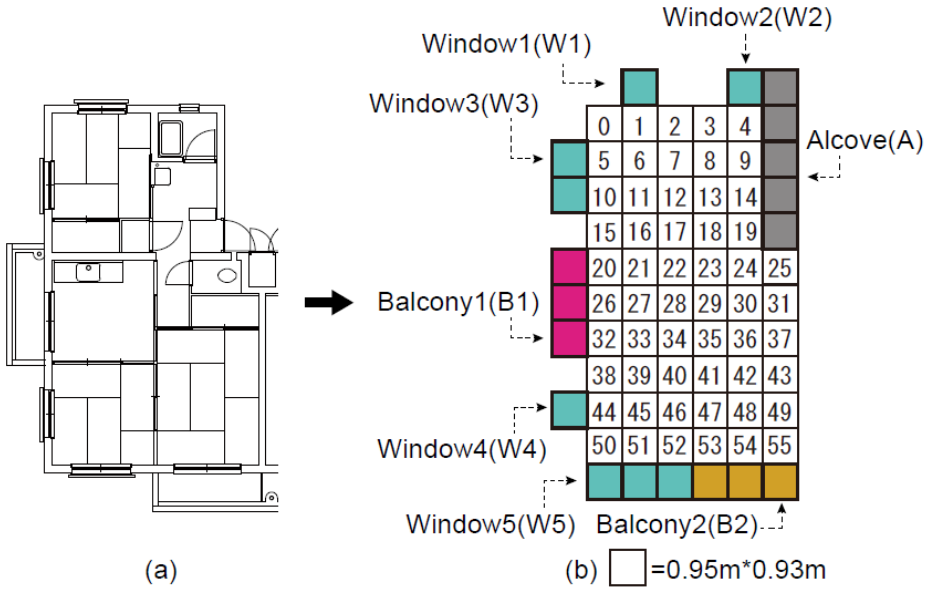
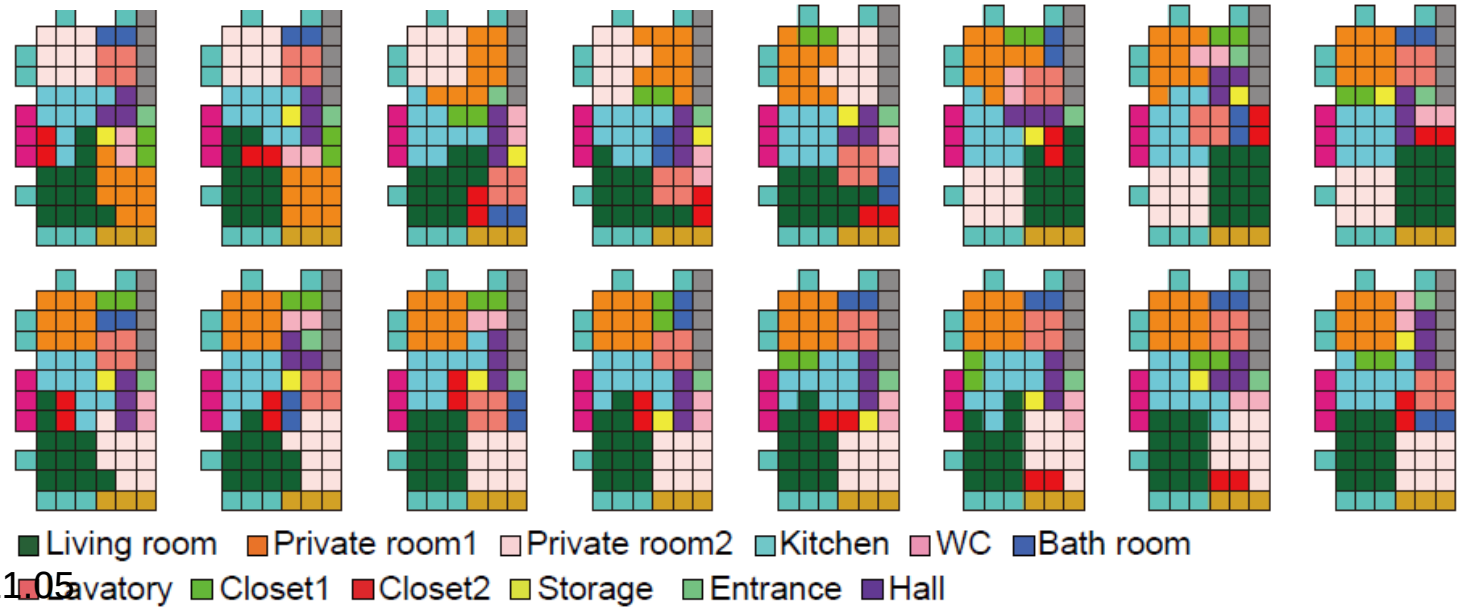


Figure 16: One of the pareto

建物フロアプランへの応用



- 大阪市大･瀧澤先生との共同研究
- 建築分野の国際会議CAADRIA2014でBest paper awardを受賞



選挙区割りへの応用

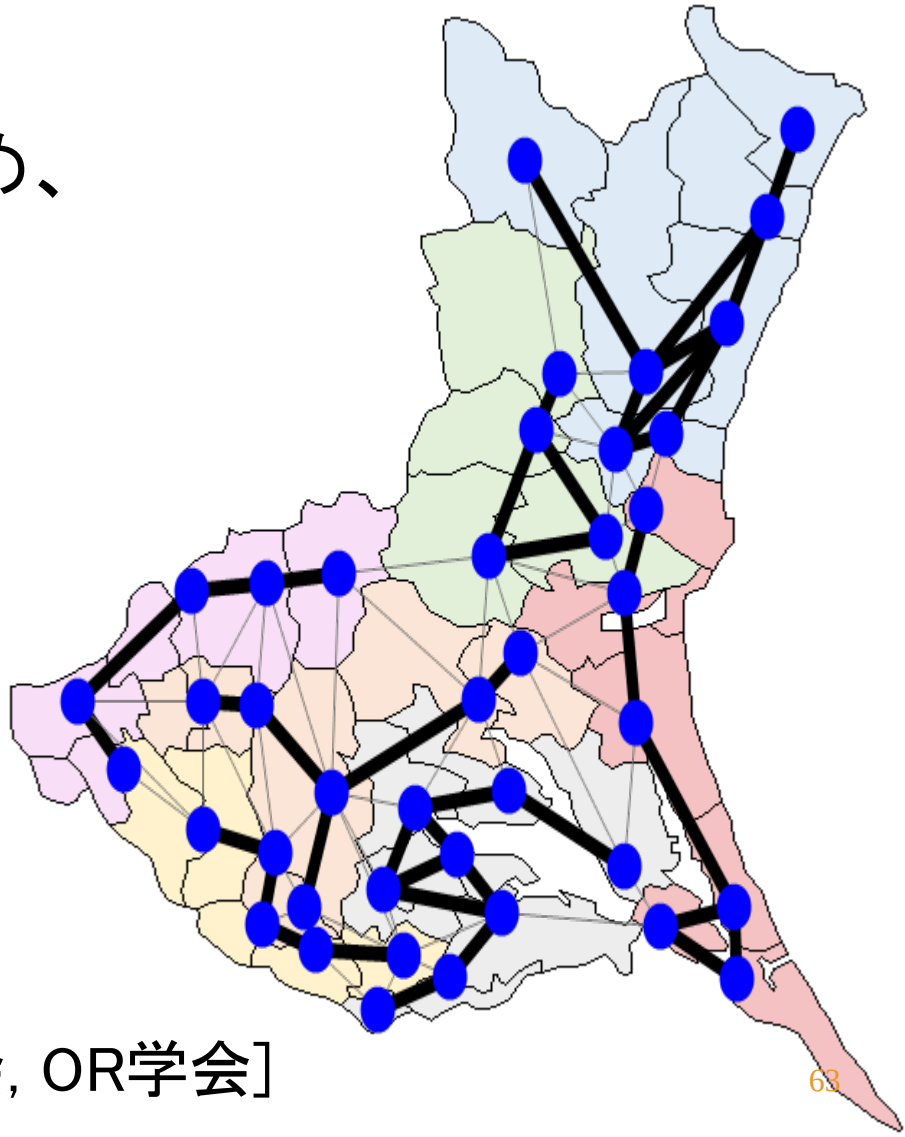
川原先生（NAIST）、堀田先生（文教大）他との共同研究

一票の格差最小だけではなく
地理的・社会的制約条件があるため、
列挙して解の分布を示すことに
社会的意味がある

茨城県の例：
頂点数41、辺数87、分割数7
（41市区町村,7選挙区）

解の個数は**11,893,998,242,846個**

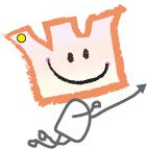
その中から格差が1.4倍以下のものを
抜き出すと **25,730,669個**
（1925.21秒で計算）



[2015選挙学会, OR学会]

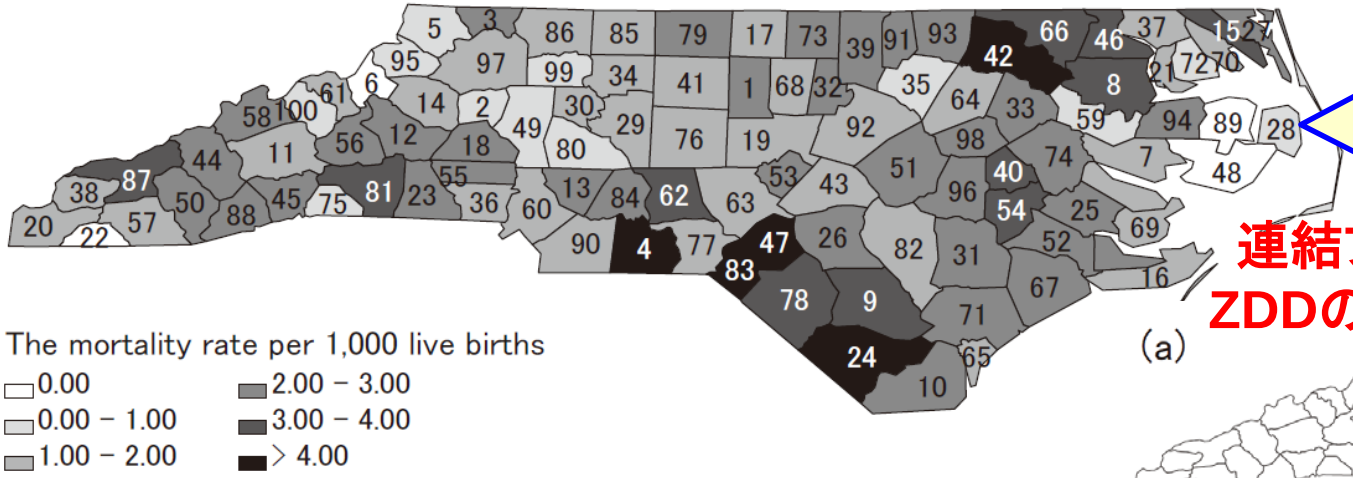
本講演の内容

- 離散構造処理とは
- Decision Diagramによる論理や集合の表現と演算
  - BDDとZDDの基本概念と技術発展の流れ
  - DDの様々な派生形
- DDによるグラフ部分構造の列挙と解析
  - フロンティア法の基本的考え方
  - 社会的に重要な問題への応用
- 最近の研究トピックから
  - 列挙・数え上げの技法に基づく
    確率・統計・データサイエンス分野への応用

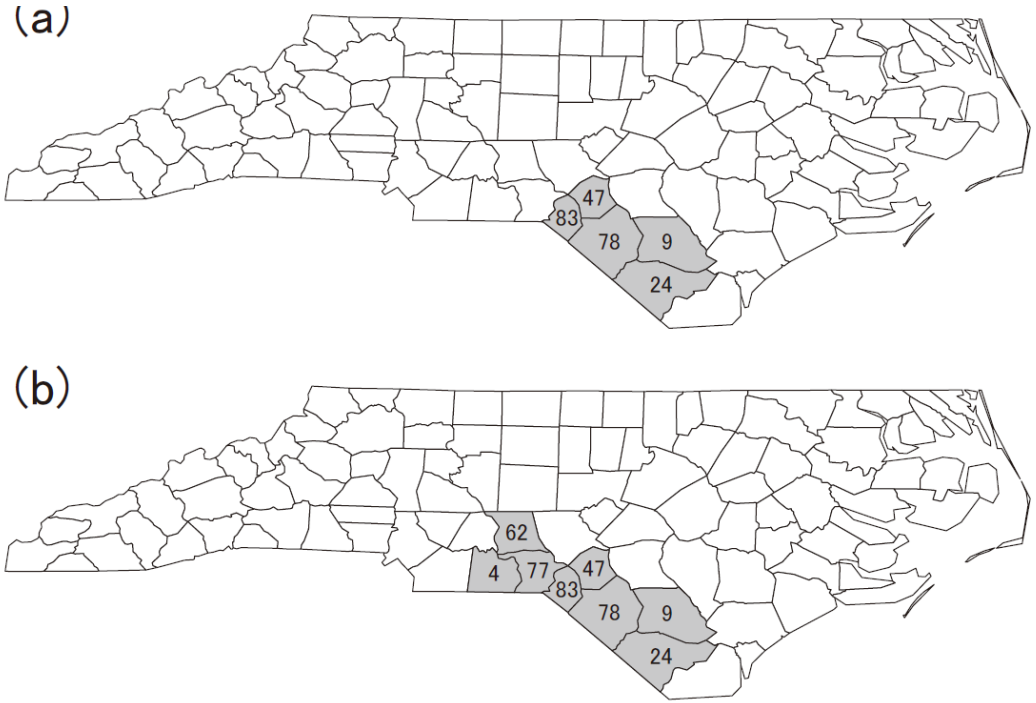


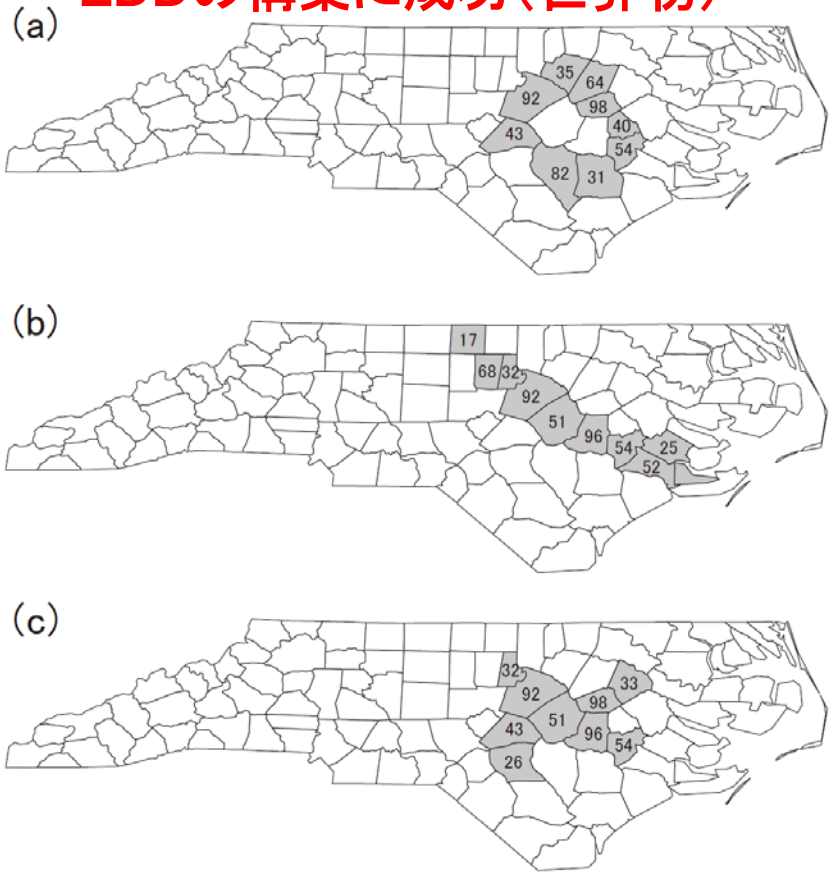
国際的に有名なホットスポット例題への適用

米国N.Carorina
州内100郡市の
乳児突然死亡率

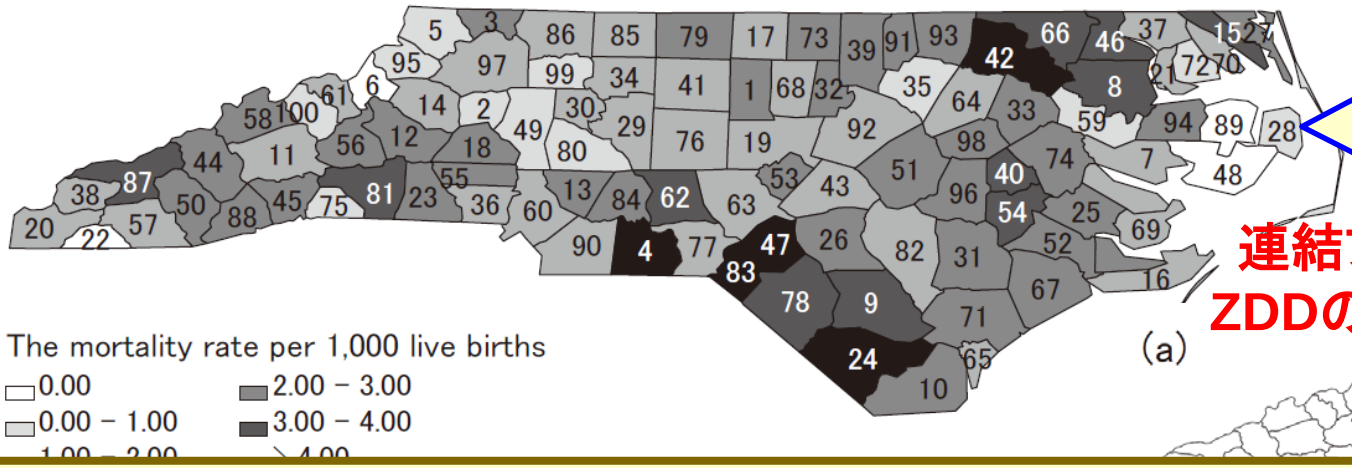


連結ブロックを全列挙する
ZDDの構築に成功（世界初）





国際的に有名なホットスポット例題への適用



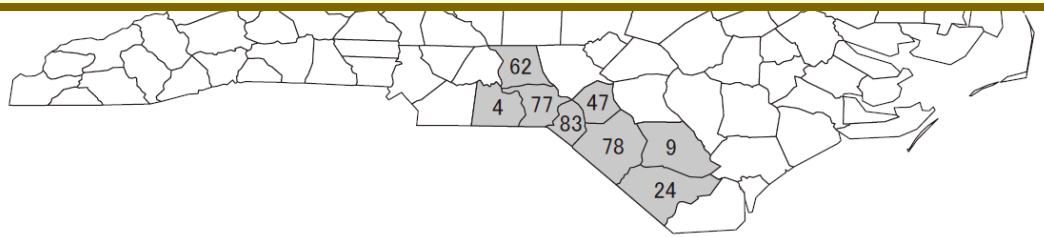
米国N.Carorina
州内100郡市の
乳児突然死亡率

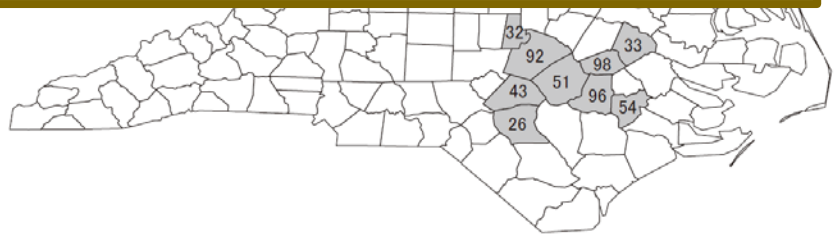
連結ブロックを全列挙する
ZDDの構築に成功（世界初）

ホットスポット＝内部と外部の尤度比最大の隣接ブロック

尤度比がある閾値αを超える隣接ブロックの全列挙：
→「ホットスポット」列挙問題（全列挙、極大のみ列挙）

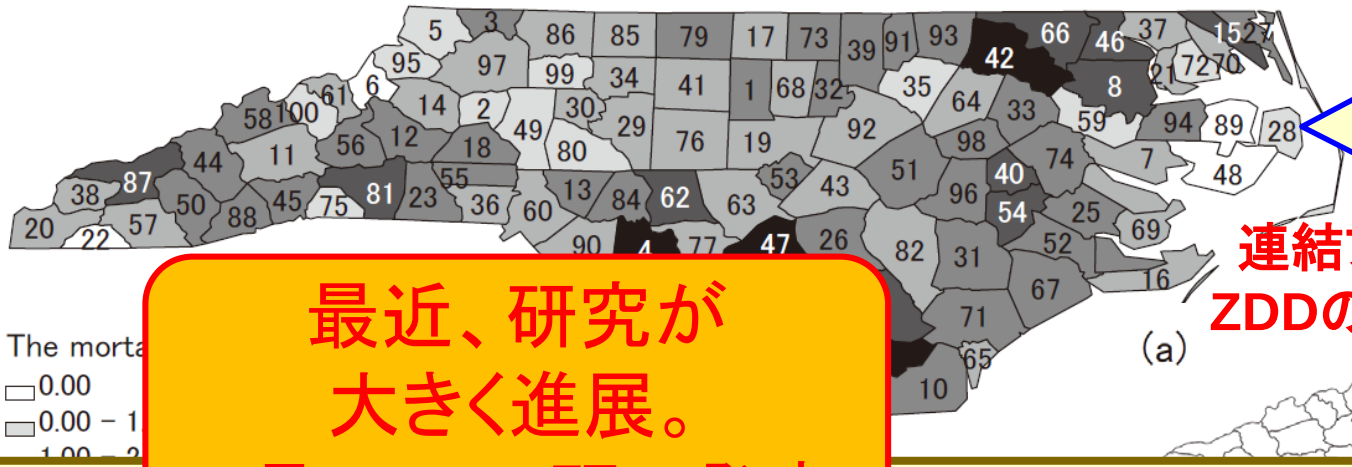
日毎、月毎データベースに対するホット/コールドの変化を抽出
→ 時系列的特徴を持つホットスポット列挙問題





国際的に有名なホットスポット例題への適用

米国N.Carorina
州内100郡市の
乳児突然死亡率



連結ブロックを全列挙する
ZDDの構築に成功（世界初）

最近、研究が
大きく進展。
９月のALG研で発表

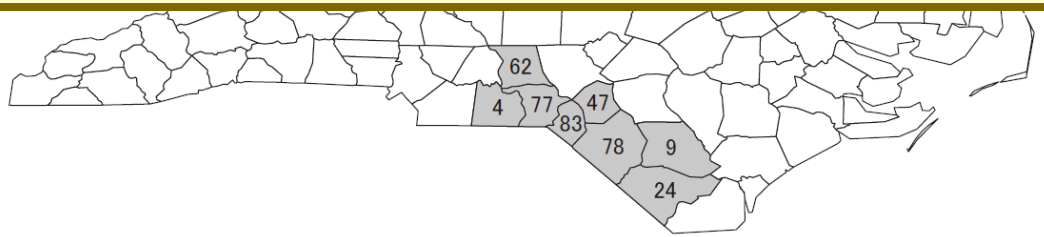
ホットスポット：内部と外部の尤度比最大の隣接ブロック

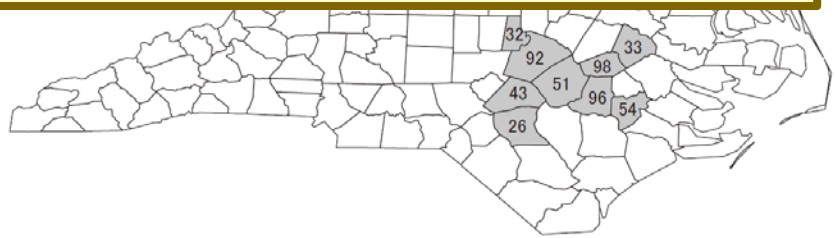
尤度比がある閾値αを超える隣接ブロックの全列挙：

→「ホットスポット」列挙問題（全列挙、極大のみ列挙）

日毎、月毎データベースに対するホット/コールドの変化を抽出

→ 時系列的特徴を持つホットスポット列挙問題





スキャン統計量に基づくホットスポット抽出

- ホットスポット抽出の既存研究では、集中度の指標として「スキャン統計量」と呼ばれる尤度比が広く用いられている

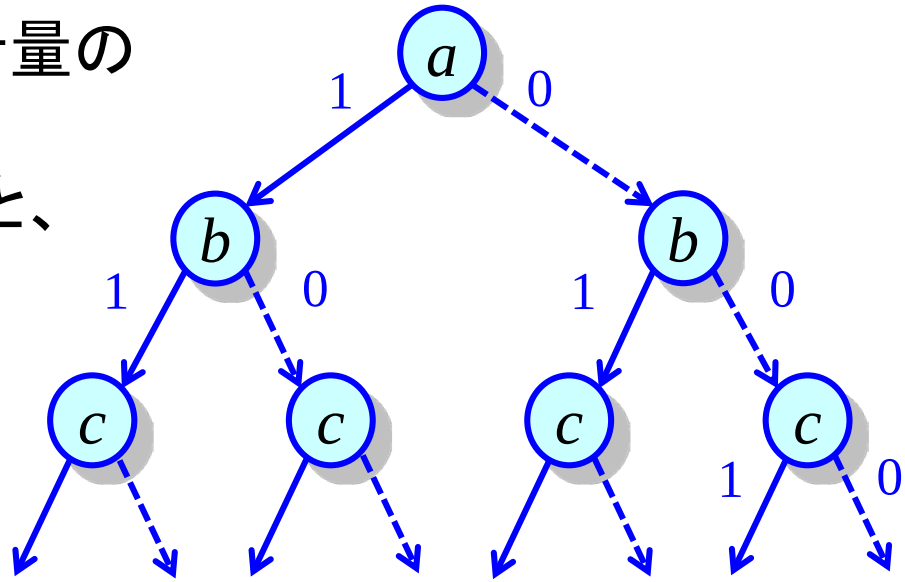
$$\lambda(Z) = \begin{cases} \dfrac{\left(\frac{c(Z)}{n(Z)}\right)^{c(Z)} \left(\frac{c(Z^c)}{n(Z^c)}\right)^{c(Z^c)}}{\left(\frac{c(G)}{n(G)}\right)^{c(G)}} & \text{if } \dfrac{c(Z)}{n(Z)} > \dfrac{c(Z^c)}{n(Z^c)} \\ 1 & \text{otherwise} \end{cases}$$

Z : 対象領域　Zc: 非対象領域　G: 全体領域
$n(Z)$: 対象領域の総人数　$c(Z)$: 対象領域の正例数

- スキャン統計量は単純な線形和ではないので、部分問題に分解することが簡単ではなく高速に探索する良い方法が見つかっていなかった。

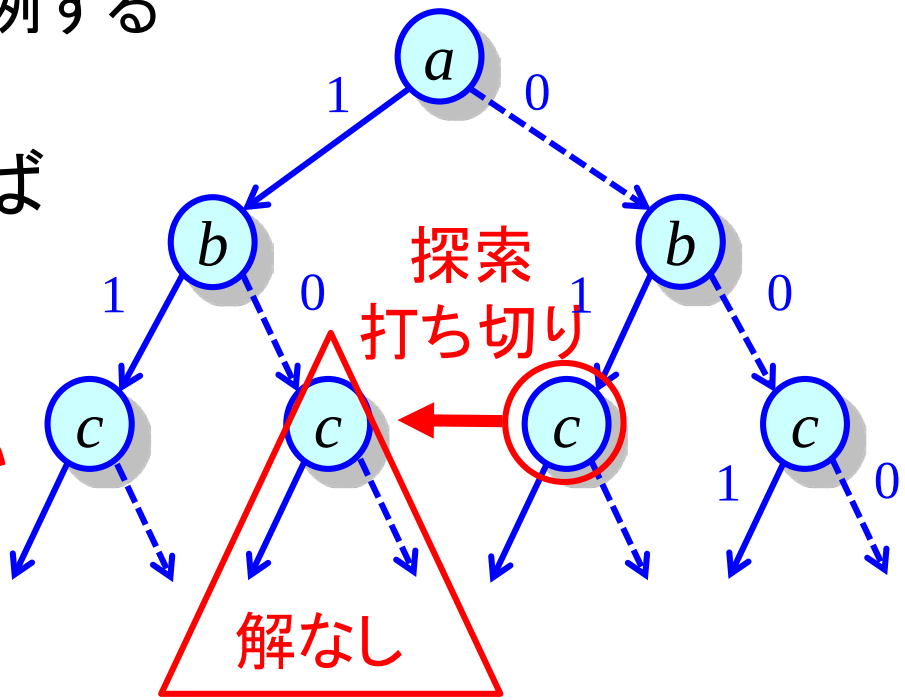
スキャン統計量を閾値として与える列挙法

- 全てのブロックを「使う/使わない」に場合分けして、網羅的にスキャン統計量を計算し、指定した閾値を上回るブロック組合せを全て出力する。
  - 統計量の最大値よりも大きな閾値を与えると「解なし」となる（探索の過程で、統計量最大となる組合せを知ることができる）
  - 閾値を小さくして行くと解の個数は指数関数的に増大
  - まず十分大きな閾値で探索して統計量の最大値を求めておいて、それより少し小さな閾値で探索すると、現実的な個数の解が生成される。（頻出パタンマイニングと類似）
- **大きい閾値で解なしの場合でも愚直に探索すると指数時間かかる**



枝刈りによる高速化のアイデア

- 探索途中に確定した変数で統計量を計算し、過去に同様の状況で解なしだった結果があれば、探索を打ち切って解なしとする
  - うまくいけば、解の個数にほぼ比例する計算時間で探索が終了する
- 統計量が単純な線形和であれば途中の統計量を見るだけで容易に枝刈りができるが、スキャン統計量は簡単ではない



$$\lambda(Z) = \begin{cases} \dfrac{\left(\frac{c(Z)}{n(Z)}\right)^{c(Z)} \left(\frac{c(Z^c)}{n(Z^c)}\right)^{c(Z^c)}}{\left(\frac{c(G)}{n(G)}\right)^{c(G)}} \\ 1 \qquad \text{otherwise} \end{cases}$$

スキャン統計量の部分的単調性（今回の主結果）

- スキャン統計量には部分的な単調性が存在
  - $c(Z)=c(Z')$ と固定したとき、$n(Z) > n(Z')$ ならば $\lambda(Z) < \lambda(Z')$
  - $n(Z)=n(Z')$ と固定したとき、$c(Z) < c(Z')$ ならば $\lambda(Z) < \lambda(Z')$
- 上記の2つの性質より下記の定理を証明

（主定理） **$n(Z) > n(Z')$ かつ $c(Z) < c(Z')$ ならば $\lambda(Z) < \lambda(Z')$**

→ 探索の各ステップで $n(Z)$と$c(Z)$を記録しておき、
両者の数値を参照することで、安全な枝刈りが可能に

$$\lambda(Z) = \begin{cases} \dfrac{\left(\frac{c(Z)}{n(Z)}\right)^{c(Z)} \left(\frac{c(Z^c)}{n(Z^c)}\right)^{c(Z^c)}}{\left(\frac{c(G)}{n(G)}\right)^{c(G)}} & \text{if } \dfrac{c(Z)}{n(Z)} > \dfrac{c(Z^c)}{n(Z^c)} \\ 1 & \text{otherwise} \end{cases}$$

尤度比最大のホットスポット探索の高速化

- 47都道府県のH28年自殺者数データでの実験結果

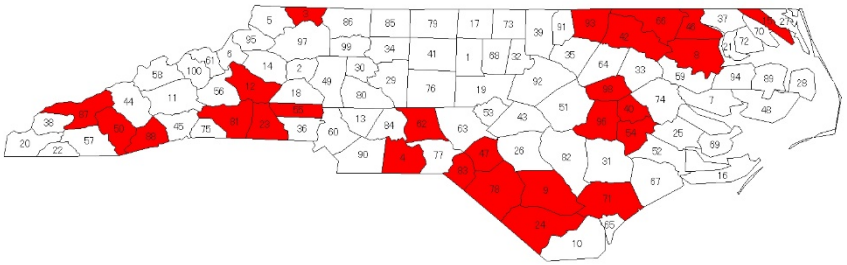
m	素朴な探索法		提案法 prop od	
	total calls	time(sec)	total calls	time(sec)
20	1,048,575	0.07	1,230	0.04
25	33,554,431	0.6	1,820	0.04
30	1,073,741,823	17.9	2,717	0.05
35	34,359,738,367	560.0	5,209	0.05
40	–	time out	7,103	0.05
45	–	time out	9,059	0.05
47	–	time out	9,847	0.05

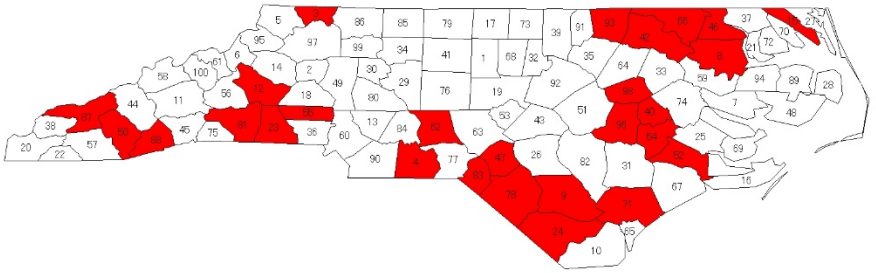
- 統計量の部分的単調性を用いた安全な枝刈り法を開発し、超高速に厳密最適解を探索することに成功
- 素朴な探索を行うと40日程度かかる計算を0.05秒に削減
  → 100万倍以上の高速化を達成

国際会議投稿中

実験結果（NC州100郡データ）

- ノースカロライナ100郡の乳児突然死データで実験
  - 対数尤度比閾値 67.7 で 2件ヒット。計算時間0.51秒





  - 対数尤度比の閾値を 67.0 に下げると 1582件ヒット
    （計算時間はほとんど同じ）
- 100ブロック規模の例題で、世界で初めて現実的な時間で厳密最適解を求められるようになった
  （素朴な計算だと1兆年かかるレベル）

今後の課題

- 今回は非連結なホットスポットもまとめて探索してしまう
  - 連結解だけ欲しくても非連結解が大量にヒット
  - 連結解に限定するためにはBDD/ZDDとの組合せが必要
- 得られたホットスポットの検定をしたい（p値を求めたい）
  - スキャン統計量と同じように部分的単調性がありそう（高いp値の解を高速に列挙できるのでは）
  - 多重検定補正をちゃんとやりたい
- ホットスポット探索の応用
  - グラフの密な部分を抽出するのに使えるのでは？
  - 多腕バンディット問題に使えるのでは？

本講演のまとめ

- あらゆる計算機応用の基盤技術であり、理論と応用の交差点（→Art層）を形成している「離散構造処理系」に焦点を絞り、今後も研究を進める
  - 「最適化と列挙は車の両輪」（©湊）
  - 解のカウントと確率計算は密接な関係がある
  - 列挙アルゴリズムは確率計算や統計検定の問題と相性が良い

  → データサイエンスを支える重要技術に
- ERATOプロジェクトの成果を引継ぎ、理論研究者と応用分野の研究者が集まる「場」をさらに発展強化させる
  - 関連する国家プロジェクトと連携し、基盤(S)成果の発展・実用化を目指す