BDD/ZDD-Based Enumeration Techniques and Real-Life Applications

Shin-ichi Minato

Hokkaido University, Japan.

Introduction of speaker

- **Shin-ichi Minato,**
  Prof. of Hokkaido Univ., Sapporo, Japan.
  - Worked for NTT Labs. from 1990 to 2004.
- Main research area:
  - 1990's: VLSI CAD (logic design and verification)
  - 2000's: Large-scale combinatorial data processing
    (Data mining, Knowledge indexing, Bayesian networks, etc.)
- 2010~2015: Research Director of "ERATO"
  MINATO Discrete Structure Manipulation Project.
- 2016～2020: PI of JSPS Basic Research Project

Hokkaido University

- Founded in 1876.
  - One of the oldest public university in Japan.
  - **Nobel prize** in Chemistry (Prof. Suzuki) in 2010.
  - Beautiful campus in the center of Sapporo city.

Sapporo city - Japan

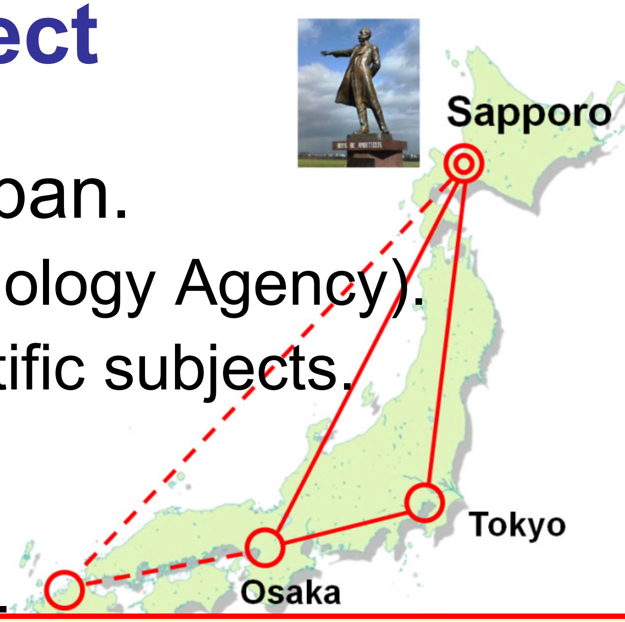
Overview of “ERATO” project

- **Top projects** of scientific research in Japan.
  - Executed by JST (Japan Science and Technology Agency).
  - 5 projects / Year are accepted from all scientific subjects.
    (Computer Science: 0 or 1 project / Year.)
  - 5 year project, total fund: ~10M USD.
    about 10 PD researchers and 3 admin staffs.

2009.10 2010.4 2010.10 2011.4 2011.10 2012.4 2012.10 2013.4 2013.10 2014.4 2014.10

(0th-year)	(1st-year)	(2nd-year)	(3rd-year)	(4th-year)	(5th-year)
acceptance Scouting GLs & PDs Lab-office construction	Kick-off WS Fall WS FIT2010 sympo. Special sympo. Visiting Knuth's home AMBN 2010 Negotiation to collab with company	Spring WS Fall WS Special sympo. Visiting CMU ALSIP 2011	Spring WS Fall WS JSAI Journal Special issue Miraikan exhibition Power NW Press rel. ALSIP 2012	Spring WS Fall WS FIT2013 sympo. Project report HU-museum exhibition RM 2013 “LAMP” Press rel.	Spring WS Fall WS RC 2013 ALSIP 2014 Next Project Started

Overview of “ERATO” project



- **Top projects** of scientific research in Japan.
  - Executed by JST (Japan Science and Technology Agency).
  - 5 projects / Year are accepted from all scientific subjects.
    (Computer Science: 0 or 1 project / Year.)
  - 5 year project, total fund: ~10M USD.
    about 10 PD researchers and 3 admin staffs.

2009.10 2010.4 2010.10 2011.4 20

(0th-year) (1st-year) (2nd

acceptance

Scouting GLs & PDs

Lab-office construction

Kick-off WS Fall WS

FIT2010 Special sympo. sympo.

Visiting Knuth's home

AMBN 2010

Negotiation to collab with company

Spring WS

Visiting CMU

ALSIP 2011

WS

Special sympo.

WS WS

JSAI Journal Special issue

Miraikan exhibition

Power NW Press rel.

ALSIP 2012

WS

FIT2013 sympo.

Project report

HU-museum exhibition

RM 2013

“LAMP” Press rel.

WS

RC 2013

ALSIP 2014

Next Project Started

JSPS Basic Research Project:
“Discrete Structure Manipulation System”
2015～2020, total 1M USD

Discrete structures and applications

Many problems solved by computers can be decomposed as a type of **discrete structures** using simple primitive operations.
→ Often needs **a huge amount of enumerative operations**.

design automation
data mining / knowledge discovery
fault analysis
bio informatics
machine learning / classification
constraint satisfaction problem
web data analysis

So many applications
→ Important for the society.

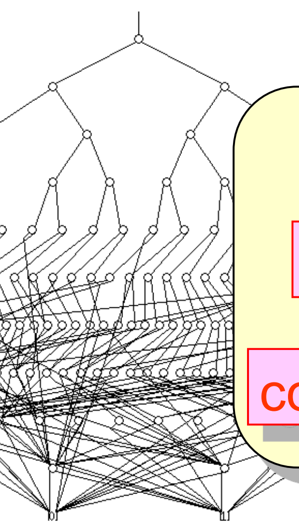
Our main subject

Discrete structure manipulation system

Performance Improvement (10–100x)

set theory
symbolic logic
inductive proof
combinatorics
graph theory
probability theory

Foundational materials for C.S. and math.



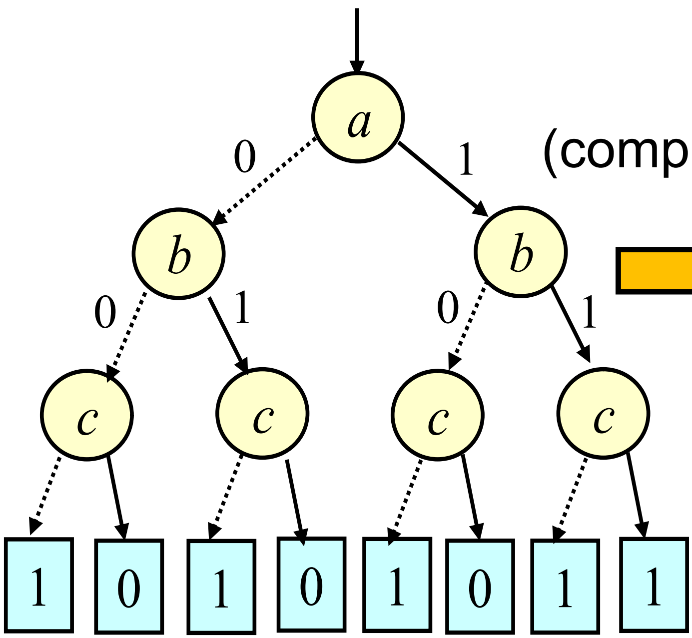
Contents:

- Brief review of BDD/ZDD
- Graph Enumeration Problems
- Real-Life Applications

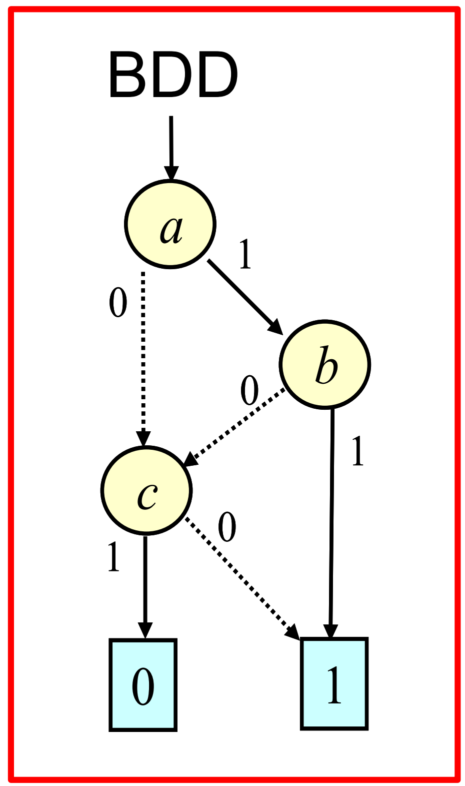
BDD (Binary Decision Diagram)

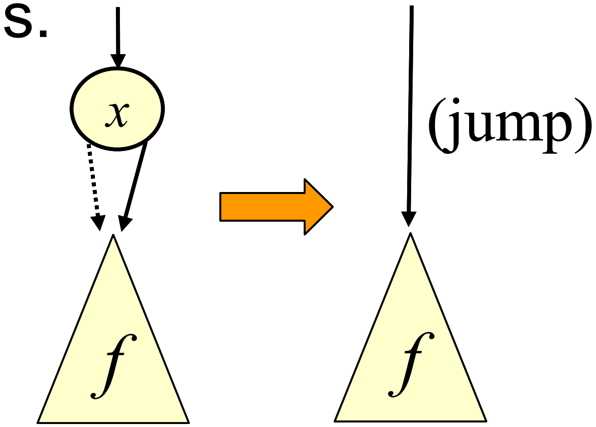
- Developed in VLSI CAD area mainly in 1990's.

(ordered)
Binary Decision Tree

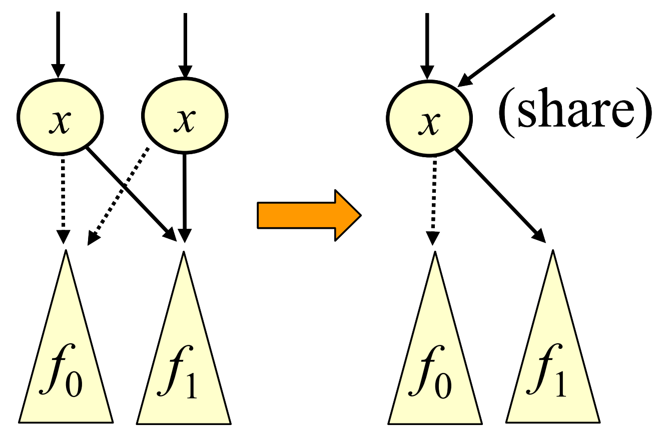


(compress)





Node elimination rule

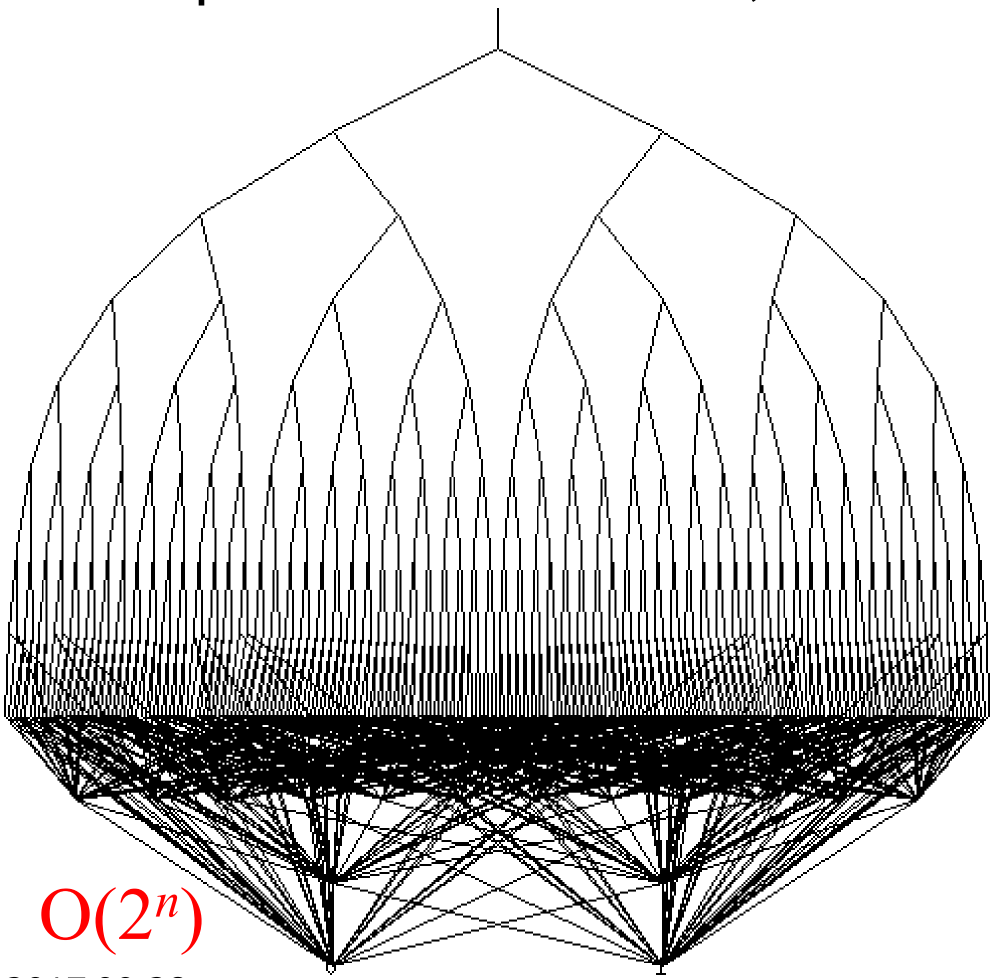


Node sharing rule

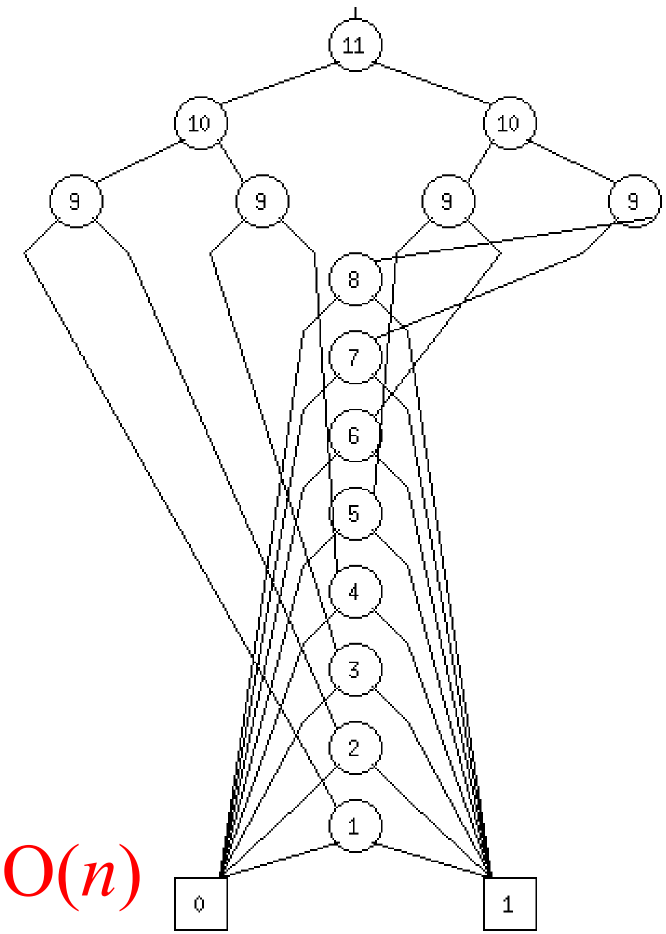
Canonical form for given Boolean functions
under a fixed variable ordering.

Effect of BDD reduction rules

- Exponential advantage can be seen in extreme cases.
  - Depends on instances, but effective for many practical ones.



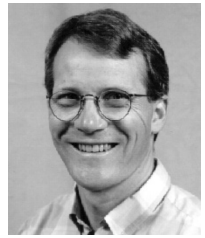
$O(2^n)$



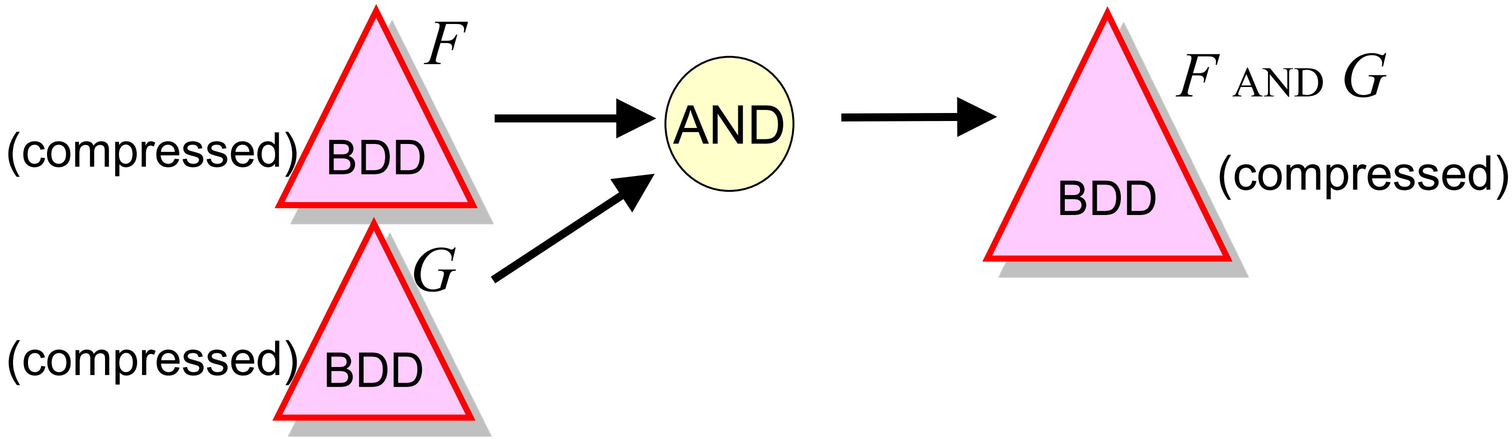
$O(n)$

BDD-based logic operation algorithm

- If the BDD starting from the binary tree: always requires exponential time & space.
- **Innovative BDD synthesis algorithm**
  - **Proposed by R. Bryant in 1986.**
  - **Best cited paper for many years in all EE&CS areas.**



R. Bryant (CMU)



A BDD can be constructed from the two operands of BDDs. (Computation time is almost linear for BDD size.)

Boolean functions and sets of combinations

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	

Boolean function:

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

Set of combinations:

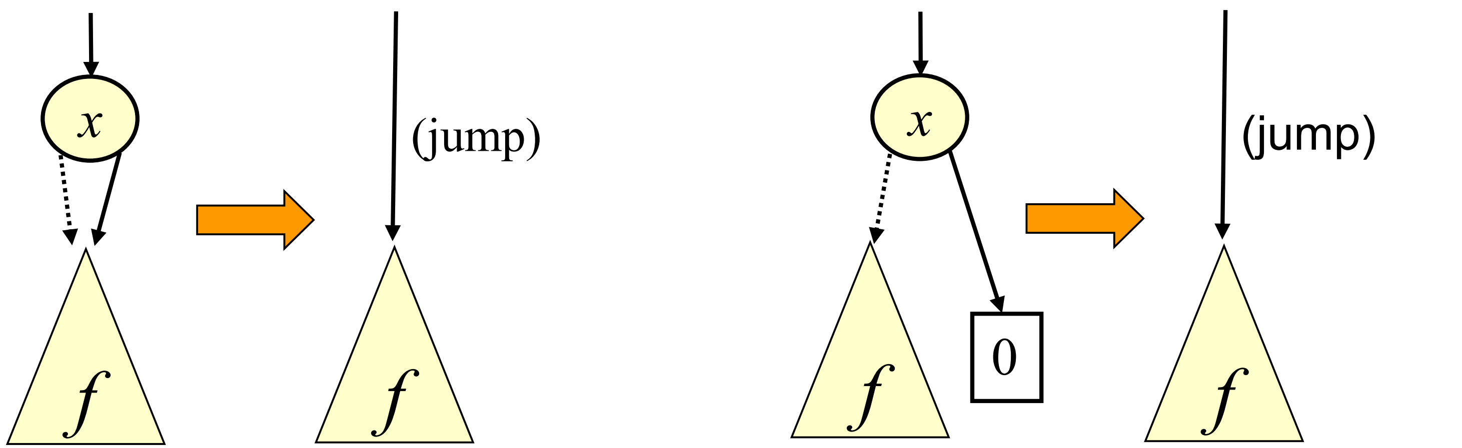
$$F = \{ab, ac, c\}$$

(customer's choice)

- Operations of combinatorial itemsets can be done by BDD-based logic operations.
  - Union of sets → logical OR
  - Intersection of sets → logical AND
  - Complement set → logical NOT

Zero-suppressed BDD (ZDD) [Minato93]

- A variant of BDDs for **sets of combinations.**
- Uses a new reduction rule different from ordinary BDDs.
  - Eliminate all nodes whose “1-edge” directly points to 0-terminal.
  - Share equivalent nodes as well as ordinary BDDs.
- If an item x does not appear in any itemset, the ZDD node of x is automatically eliminated.
  - When average occurence ratio of each item is 1%, ZDDs are more compact than ordinary BDDs, up to 100 times.

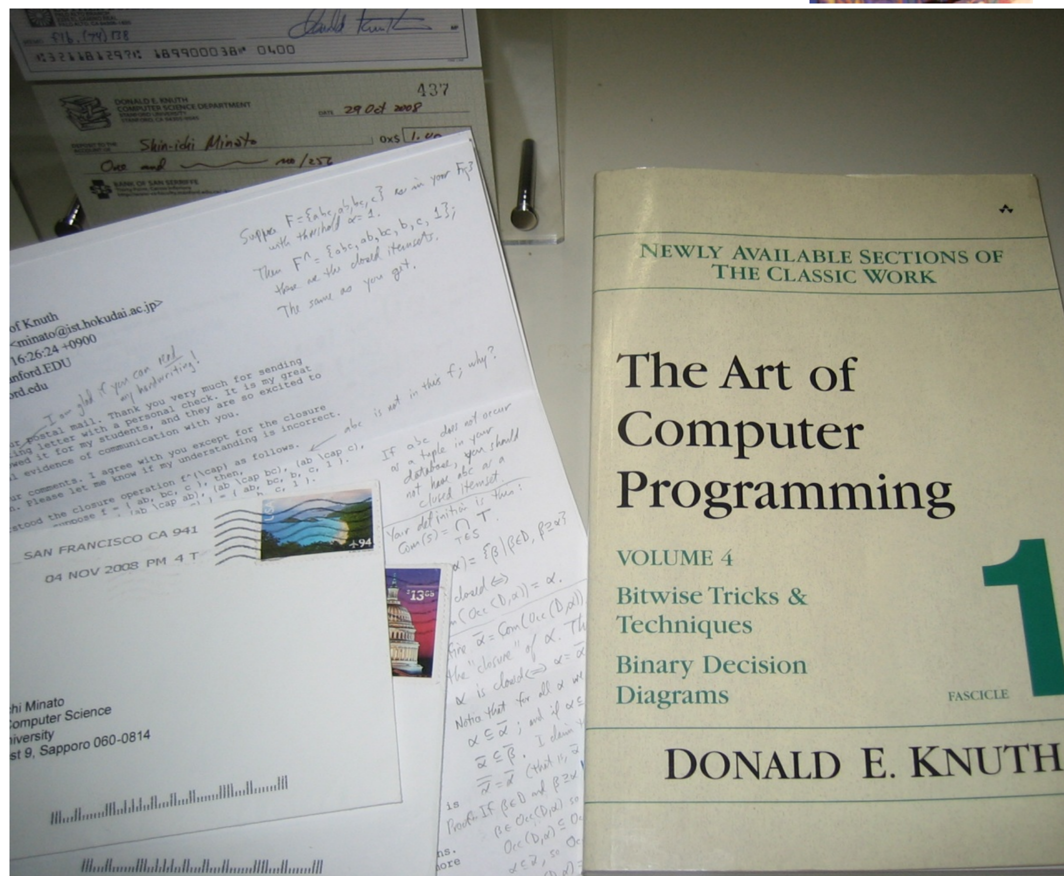


Ordinary BDD reduction

Zero-suppressed reduction

BDDs/ZDDs in the Knuth's book

- The latest Knuth's book fascicle (Vol. 4-1) includes a BDD section with 140 pages and 236 exercises.
- In this section, Knuth used 30 pages for ZDDs, including more than 70 exercises.
  - I honored to serve proofreading of the draft version of his article.
  - Knuth recommended to use "ZDD" instead of "ZBDD."
  - He reorganized ZDD operations and named "Family Algebra."
  - 2010/05, I visited Knuth's home and discussed the direction of future work.



Algebraic operations for ZDDs

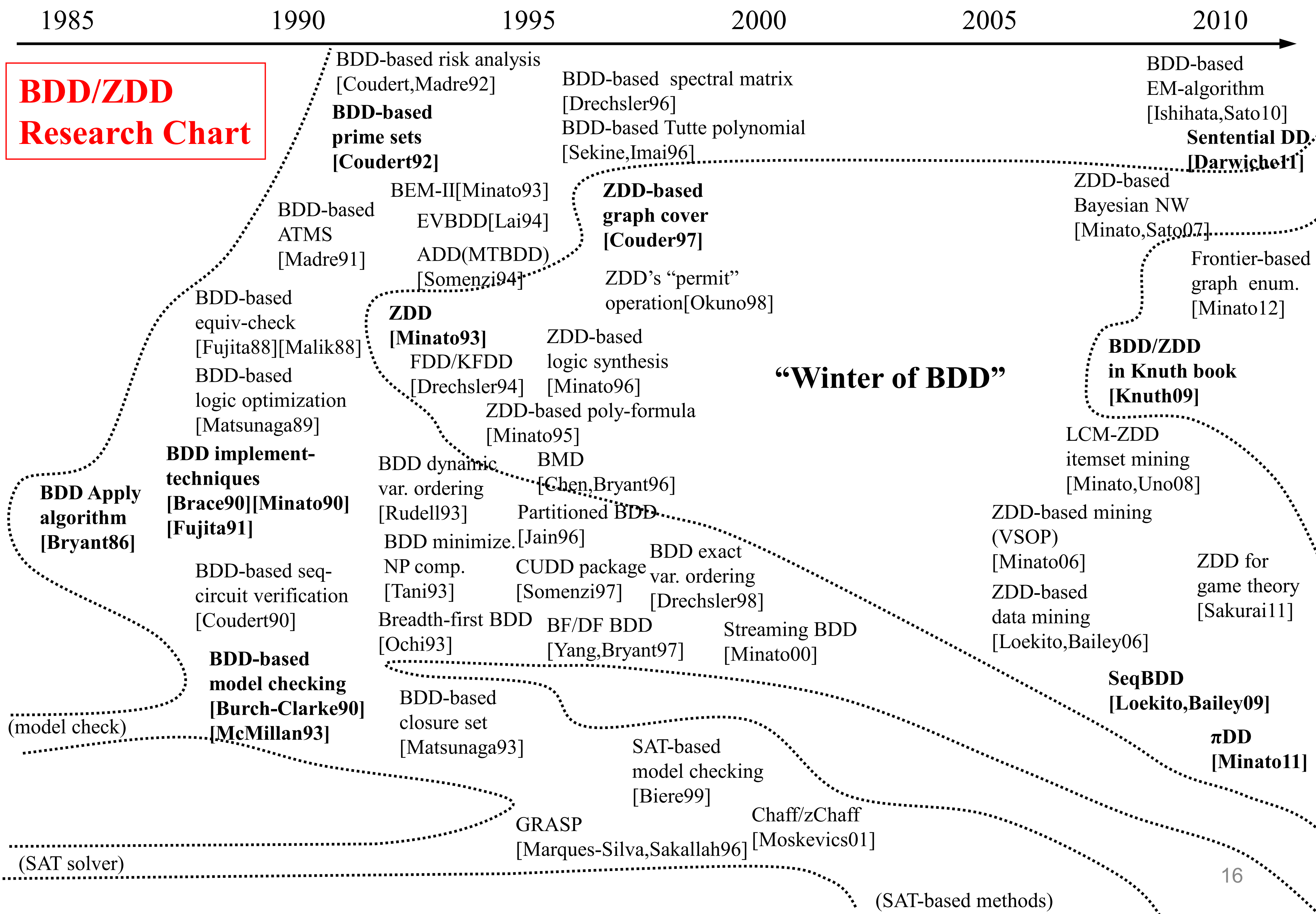
- Knuth evaluated not only the data structure of ZDDs, but more interested in **the algebra on ZDDs**.

φ, {1}	***Empty*** and ***singleton set***. (0/1-terminal)	Basic operations (Corresponds to Boolean algebra)
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.	
$P * Q$	***Cartesian product set*** of *P* and *Q*.	**New operations introduced by Minato.**
P / Q	***Quotient set*** of *P* divided by *Q*.	
$P \% Q$	***Remainder set*** of *P* divided by *Q*.	

Useful for many practical applications.

Formerly I called this "unate cube set algebra," but Knuth reorganized as **"Family algebra."**

BDD/ZDD Research Chart



Applications of BDDs/ZDDs

- BDD-based algorithms have been developed mainly in VLSI logic design area. (since early 1990's.)
  - Equivalence checking for combinational circuits.
  - Symbolic model checking for logic / behavioral designs.
  - Logic synthesis / optimization.
  - Test pattern generation.
- Recently, BDDs/ZDDs are applied for not only VLSI design but also for more general purposes.
  - Data mining (Fast frequent itemset mining) [Minato2008]
  - Probabilistic Modeling (SDD, etc.) [Darwiche2011]
  - Graph enumeration problems [Knuth2009]

Frequent itemset mining

- Basic and well-known problem in database analysis.

Record ID	Tuple
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*

Frequency threshold = 10 → $\{ b \}$

Frequency threshold = 8 → $\{ ab, a, b, c \}$

Frequency threshold = 7 → $\{ ab, bc, a, b, c \}$

Frequency threshold = 5 → $\{abc, ab, bc, ac, a, b, c \}$

Frequency threshold = 1 → $\{abc, ab, bc, ac, a, b, c \}$

"LCM over ZDDs" [Minato et al. 2008]

- ***LCM:*** *[Uno2003]*
  Output-linear time algorithm of frequent itemset mining.
- ***ZDD:*** *[Minato93]*
  A compact graph-based representation for large-scale sets of combinations.

Combination of the two techniques

Generates large-scale frequent itemsets on the main memory, with a very small overhead from the original LCM.

(→ Sub-linear time and space to the number of solutions when ZDD compression works well.)

LCM over ZDDs: An example

- The results of frequent itemsets are obtained as ZDDs on the main memory. (not generating a file.)

Record ID	Tuple
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 over ZDDs

Freq. thres. α = 7

$\longrightarrow$

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

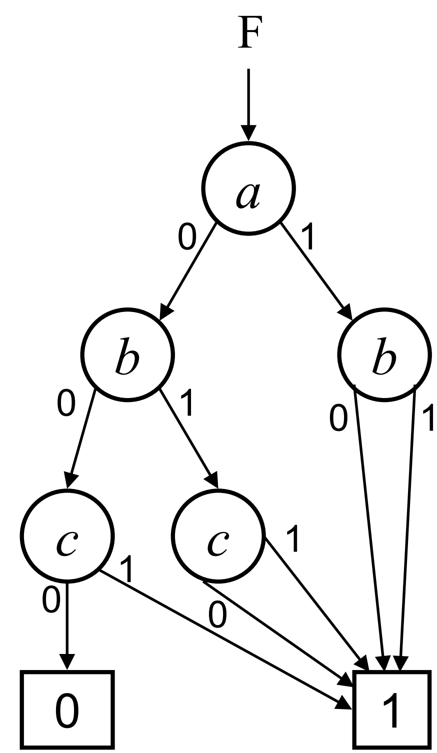


Table 2. Co... CM o... pre...

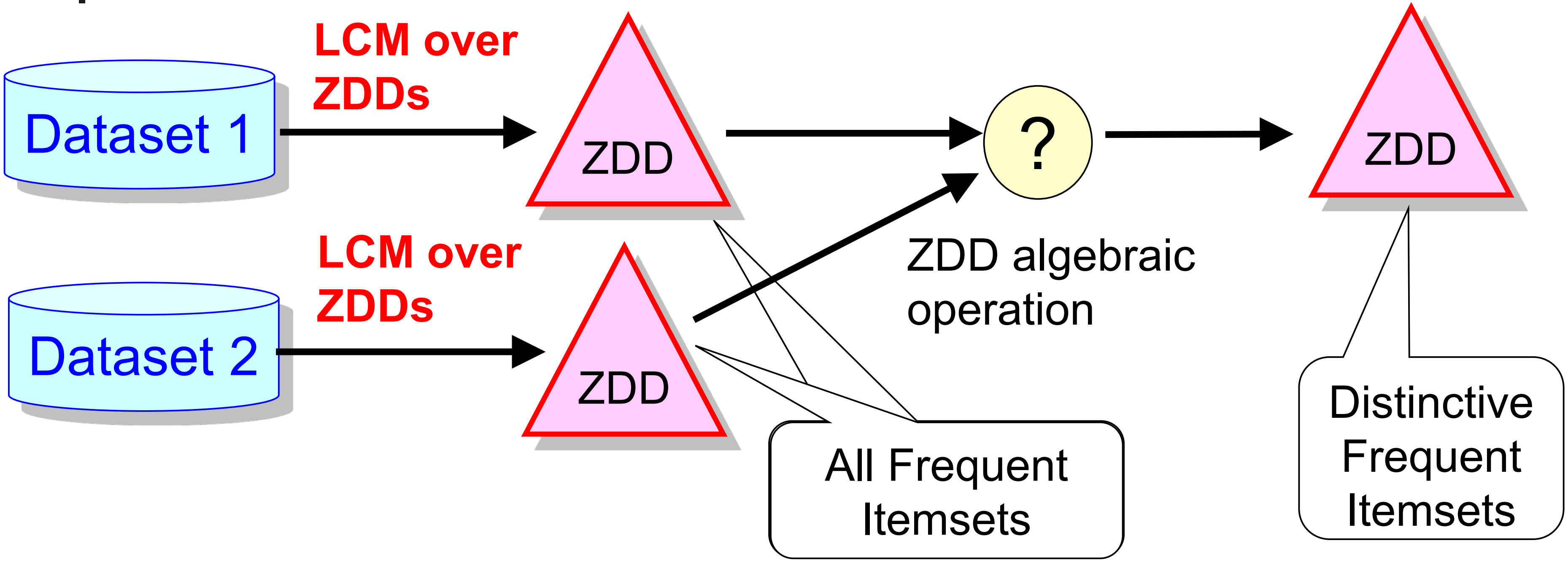
solutions

LCM over ZDDs

Original LCM

Dataset name: min. support	itemsets	CMov... |ZBDD|	Time(s)	...nt Time(s)	LC... Time(s)	...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	118.0[illegible]

Post Processing after LCM over ZDDs

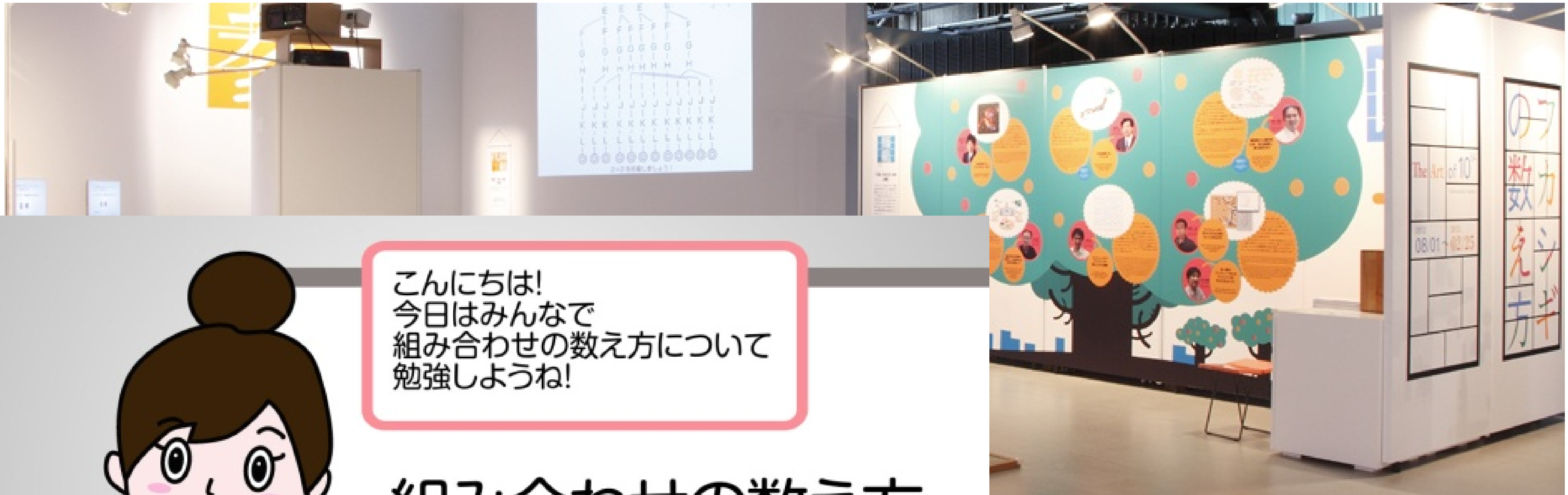


- We can extract distinctive itemsets by comparing frequent itemsets for multiple sets of databases.
  - Various ZDD algebraic operations can be used for the comparison of the huge number of frequent itemsets.

Solving Graph Enumeration Problems Using BDDs/ZDDs

Movie to show “Power of Enumeration”

- Our project supervised exhibition at "Miraikan" (National Future Science Museum of Japan) on 2012.



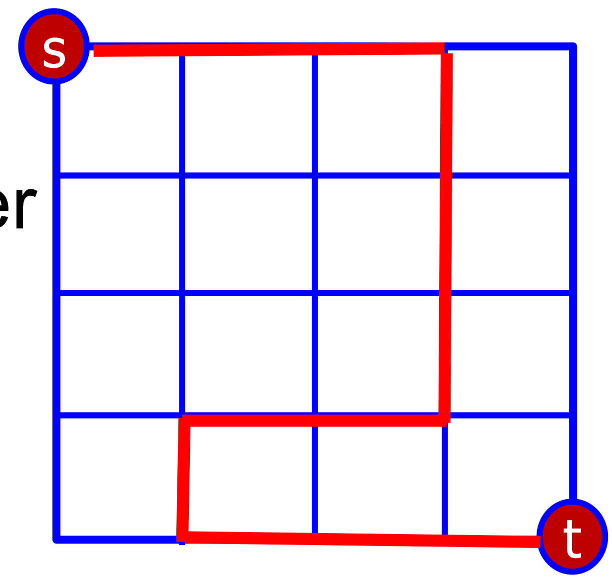
The exhibition video got 1.9 million views ! (still now growing.)

Purpose of the movie

- Shows strong power of combinatorial explosion, and importance of algorithmic techniques.
- Mainly for junior high school to college students
  - Not using any difficult technical terms.
  - Something like a funny science fiction story.
- We used the enumerating problem of "self-avoiding walk" on n x n grid graphs
  - This problem is discussed in the ZDD-section of the Knuth-book, section 7.1.4.
  - We received a letter from Knuth:
    "I enjoyed the You-Tube video about big numbers, and shared it to several friends."

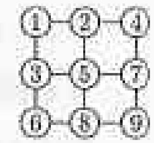
Enumerating “seif-avoiding walks”

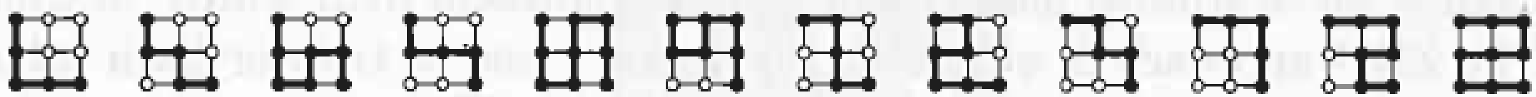
- Counting shortest s-t paths is quite easy. (→ ${}_{2n}C_n$; educated in high school.)
- If allowing non-shortest paths, suddenly difficult. No simple calculation formula has been found.
  - Many people requested the formula because the movie shows a super-big number, which the teacher spent 250,000 years to count.
  - However, no formula exists. Only efficient algorithm!



“simpath” in Knuth-book

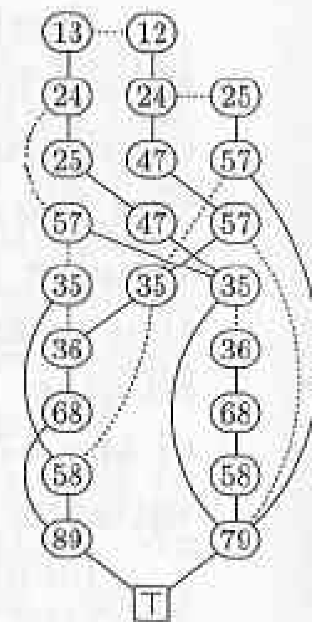
We can also use ZDDs to represent simple paths in an *undirected* graph. For example, there are 12 ways to go from the upper left corner of a 3×3 grid to the lower right corner, without visiting any point twice:





(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 $\boxed{\bot}$.) 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 an m X n square grid.

Integer Sequences: A007764

(n,n), n >= 0, of a (n+1) X (n+1) square

adjacent grid cells, should the definition say "Number of nonintersecting (or self-avoiding) rook paths joining opposite corner cells of an (n+1) X (n+1) grid." (End)

REFERENCES

S. R. Finch, Mathematical Constants, Cambridge, 2003, pp. 331-339.
D. E. Knuth, 'Things A Computer Scientist Rarely Talks About,' CSLI Publications, Stanford, CA, 2001, pages 27-28.
D. E. Knuth, The Art of Computer Programming, Section 7.1.4.
Netnews group rec.puzzles, Frequently Asked Questions (FAQ) file. (Science Section).

LINKS

I. Jensen, H. Iwashita, R. Spaans, Table of n, a(n) for n = 0..26 (I. Jensen computed terms 0 to 19, H. Iwashita computed 20 and 21, R. Spaans computed 22 to 24, and H. Iwashita computed 25 and 26)
M. Bousquet-Melou, A. J. Guttmann and I. Jensen, Self-avoiding walks crossing a square
Doi, Maeda, Nagatomo, Niiyama, Sanson, Suzuki, et al., Time with class! Let's count! [Youtube-animation demonstrating this sequence. In Japanese with English translation]
S. R. Finch, Self-Avoiding-Walk Connective Constants
H. Iwashita, J. Kawahara, and S. Minato, ZDD-Based Computation of the Number of Paths in a Graph
H. Iwashita, Y. Nakazawa, J. Kawahara, T. Uno, and S. Minato, Efficient Computation of the Number of Paths in a Grid Graph with Minimal Perfect Hash Functions
I. Jensen, Series Expansions for Self-Avoiding Walks
OEIS Wiki, Self-avoiding walks
Ruben Grønning Spaans, C program
Eric Weisstein's World of Mathematics, Self-Avoiding Walk

EXAMPLE

Suppose we start at (0,0) and end at (n-1,n-1). Let U, D, L, R denote steps that are up, down, left, right.
a(2) = 2: UR or RU.
a(3) = 12: UURR, UURDRU, UURDDRUU, URUR, URRU, URDRUU and their reflections in the x=y line.

CROSSREFS

Sequence in context: A243807 A006023 A039748 * A015195 A051421 A182162
Adjacent sequences: A007761 A007762 A007763 * A007765 A007766 A007767

KEYWORD

nonn,walk,hard,nice

AUTHOR

David Radcliffe and Don Knuth

EXTENSIONS

Computed to n=11 by John Van Rosendale in 1981, extended to n=12 by D. E. Knuth, Dec 07 1995.

Number of paths for *n* x *n* grid graphs

0	1
1	2
2	12
3	184
4	8512
5	1262816
6	575780564
7	789360053252
8	3266598486981642
9	41044208702632496804
10	1568758030464750013214100
11	182413291514248049241470885236
12	64528039343270018963357185158482118
13	69450664761521361664274701548907358996488
14	227449714676812739631826459327989863387613323440
15	2266745568862672746374567396713098934866324885408319028
16	68745445609149931587631563132489232824587945968099457285419306
17	6344814611237963971310297540795524400449443986866480693646369387855336
18	178211284084206512989338494665232527516783806570476765593145247460582669278253 2
19	1523344971704879993080742810319229690899454255323294555776029866737355060592877569255844
20	3962892199823037560207299517133362502106339705739463771515237113377010682364035706704472064940398
21	31374751050137102720420538137382214513103312193698723653061351991346433379389385793965576992246021316463868
22	7559702866673453396615191233152226193531037320724094811673914104795179257927436312349870388833176349872711714044 39792
23	55435429355237477009914318489061437930690379970964331332556958646484008407334885544566386924020875711242060085408513482933945720
24	12371712231207064758338744862673570832373041989012943539678727080484951695515930485641394550792153037191858028212512280926600304581386791094
25	8402974857881133471007083745436809127296054293775383549824742623937028497898215256929178577083970960121625602506027316549718402106494049978375604247408
26	17369931586279272931175440421236498900372229588288140604663703720910342413276134762789218193498006107082296223143380491348290026721931129627708738890853908108906396

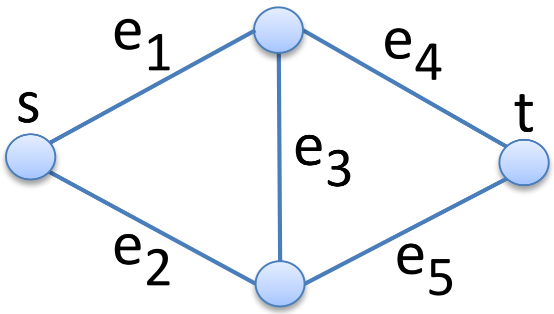
12 x 12: [Knuth 1995]

19 x 19:[B.-Melou 2009]

26 x 26: Our record in Nov. 2013
(1404 edges included in the graph.)

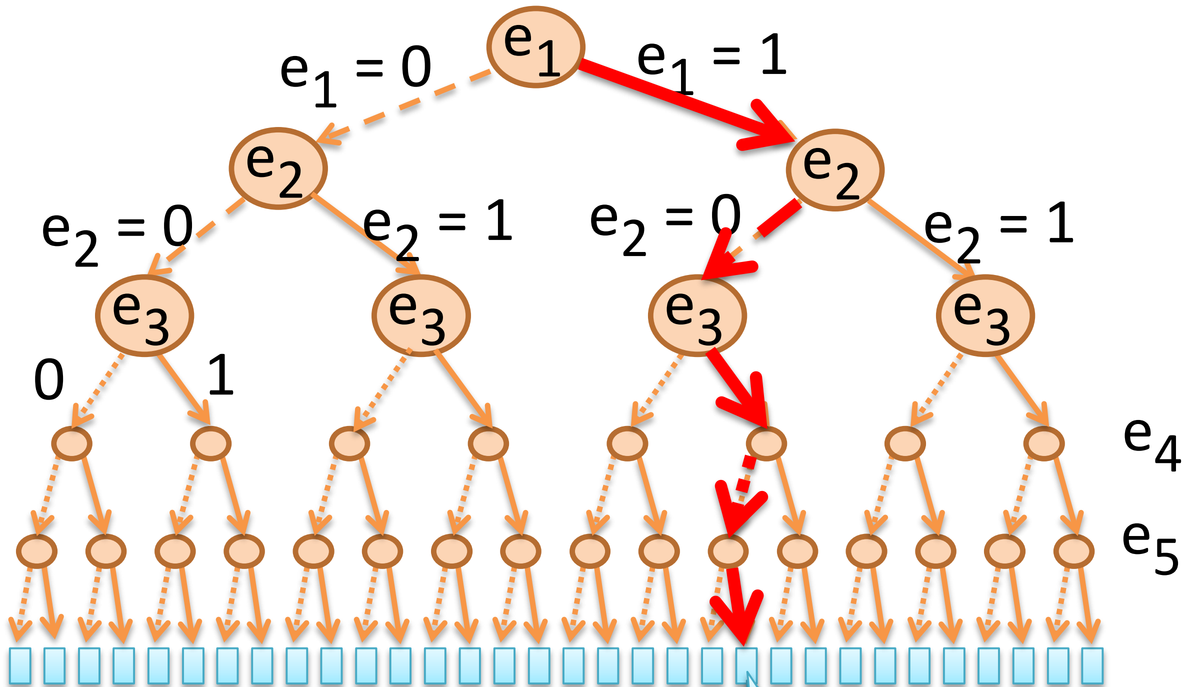
- up to 18 × 18, we can generate a ZDD to keep all solutions.
- from 19 × 19, we just count the number of solutions.
- from 22 × 22, we only consider the n × n grid graphs.

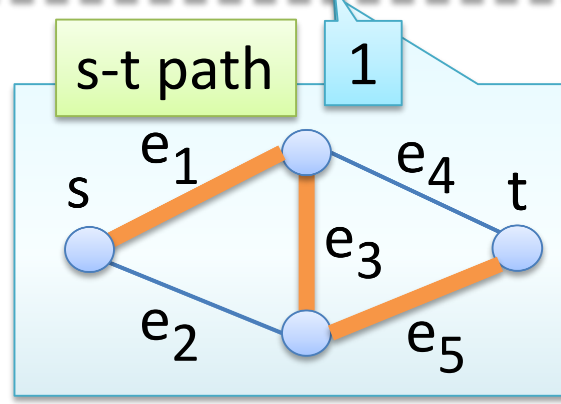
Knuth's algorithm "Simpath"



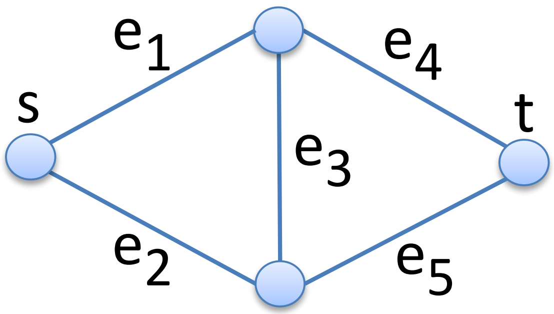
1. Assign a full order to all edges from e_1 to e_n.

2. Constructing a binary decision tree by case-splitting on each edges from e_1 to e_n.



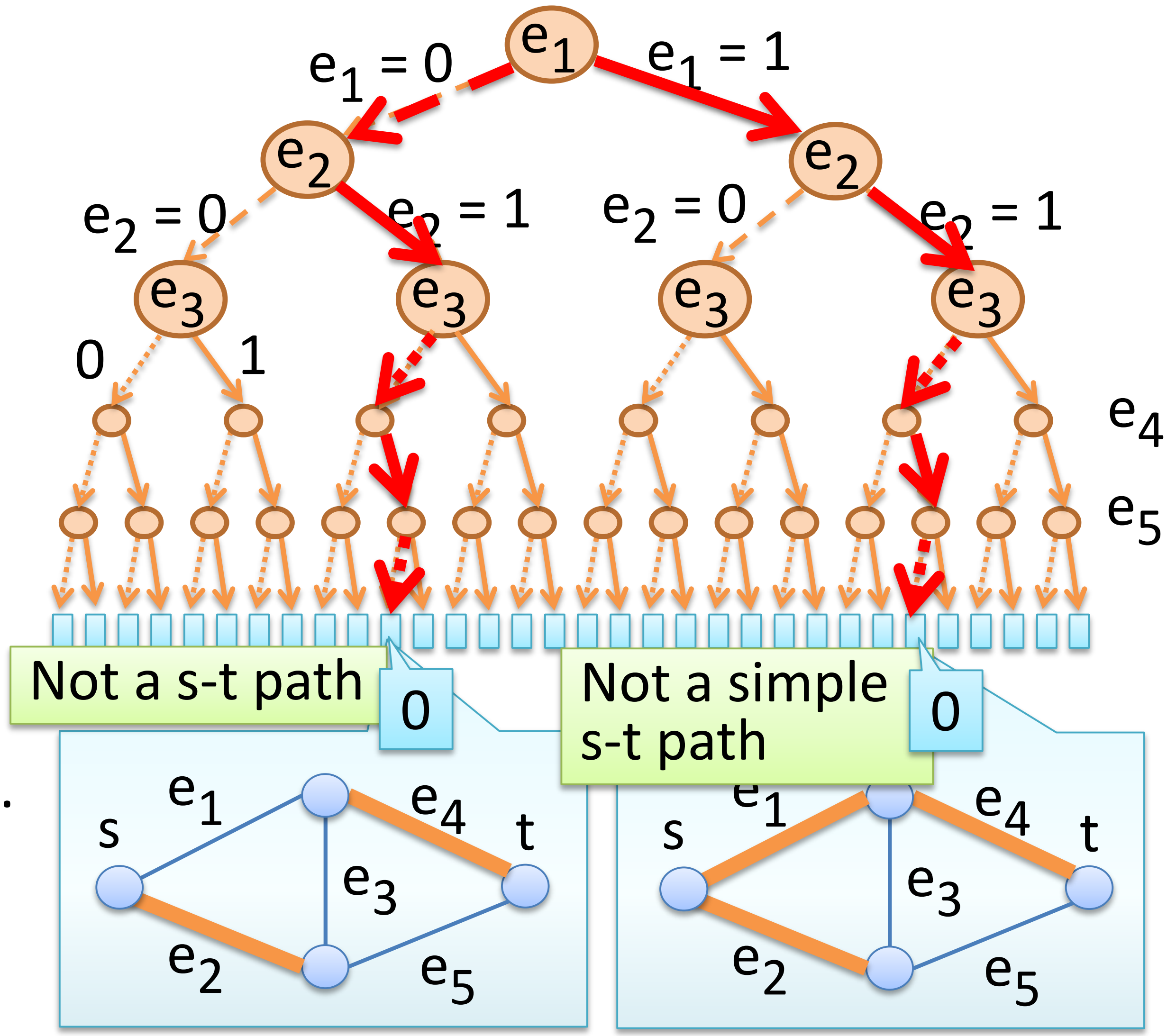


Knuth's algorithm "Simpath"

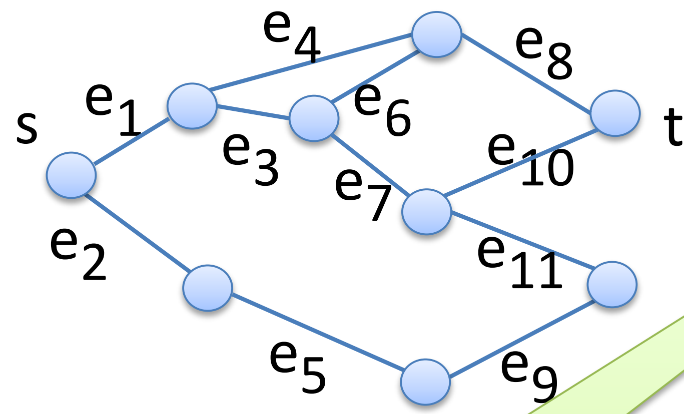


1. Assign a full order to all edges from e_1 to e_n.

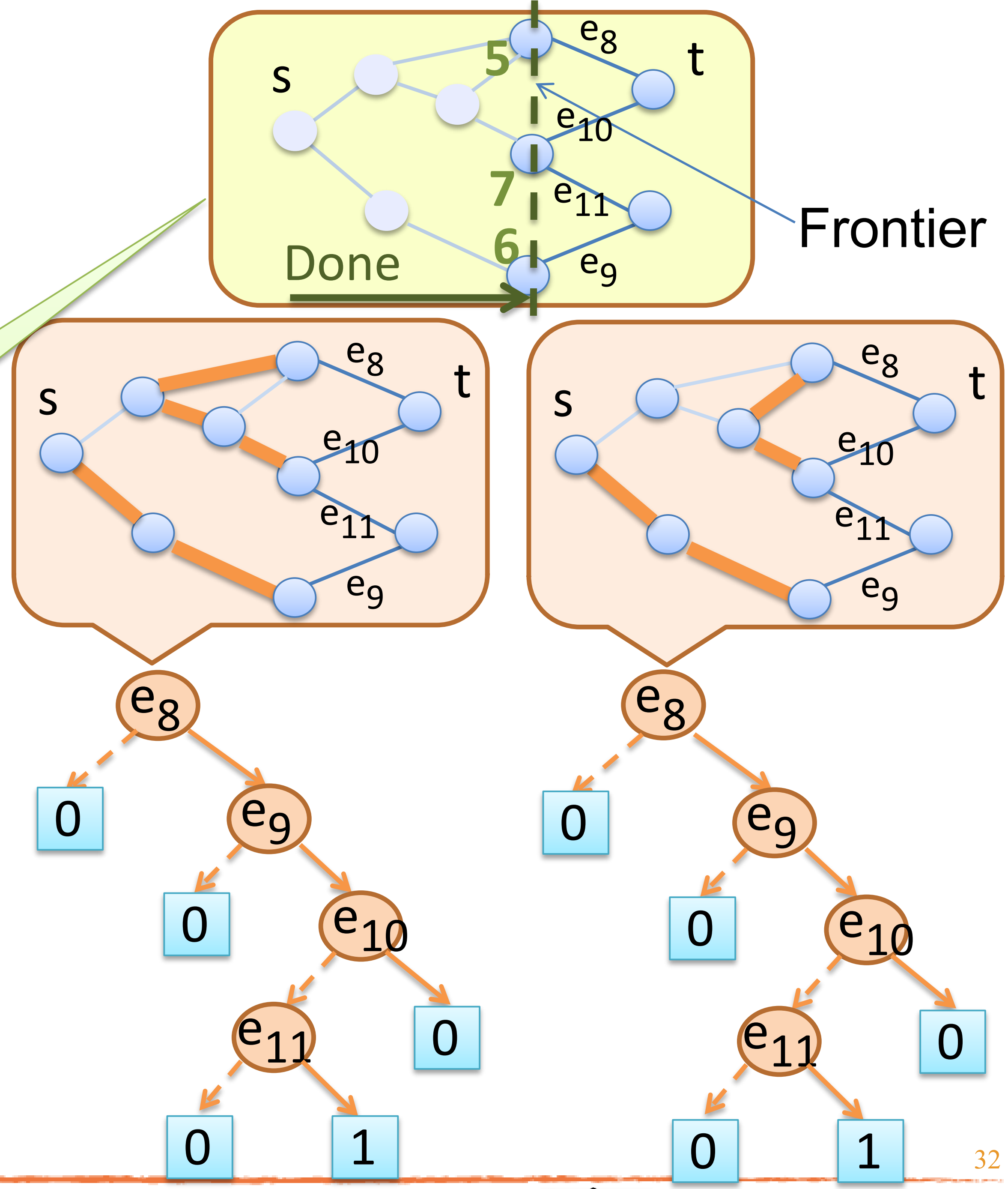
2. Constructing a binary decision tree by case-splitting on each edges from e_1 to e_n.



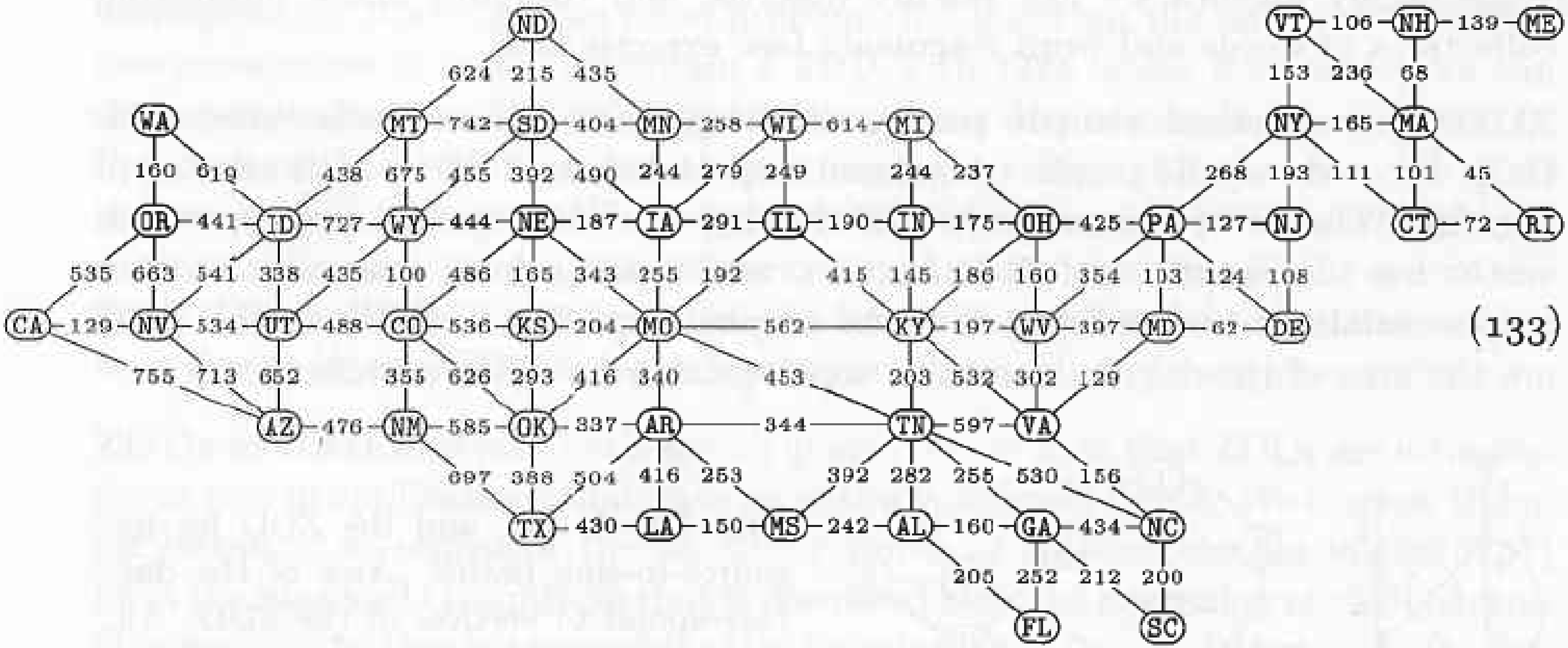
Knuth's algorithm Simpath



6 connecting to s
5 connecting to 7



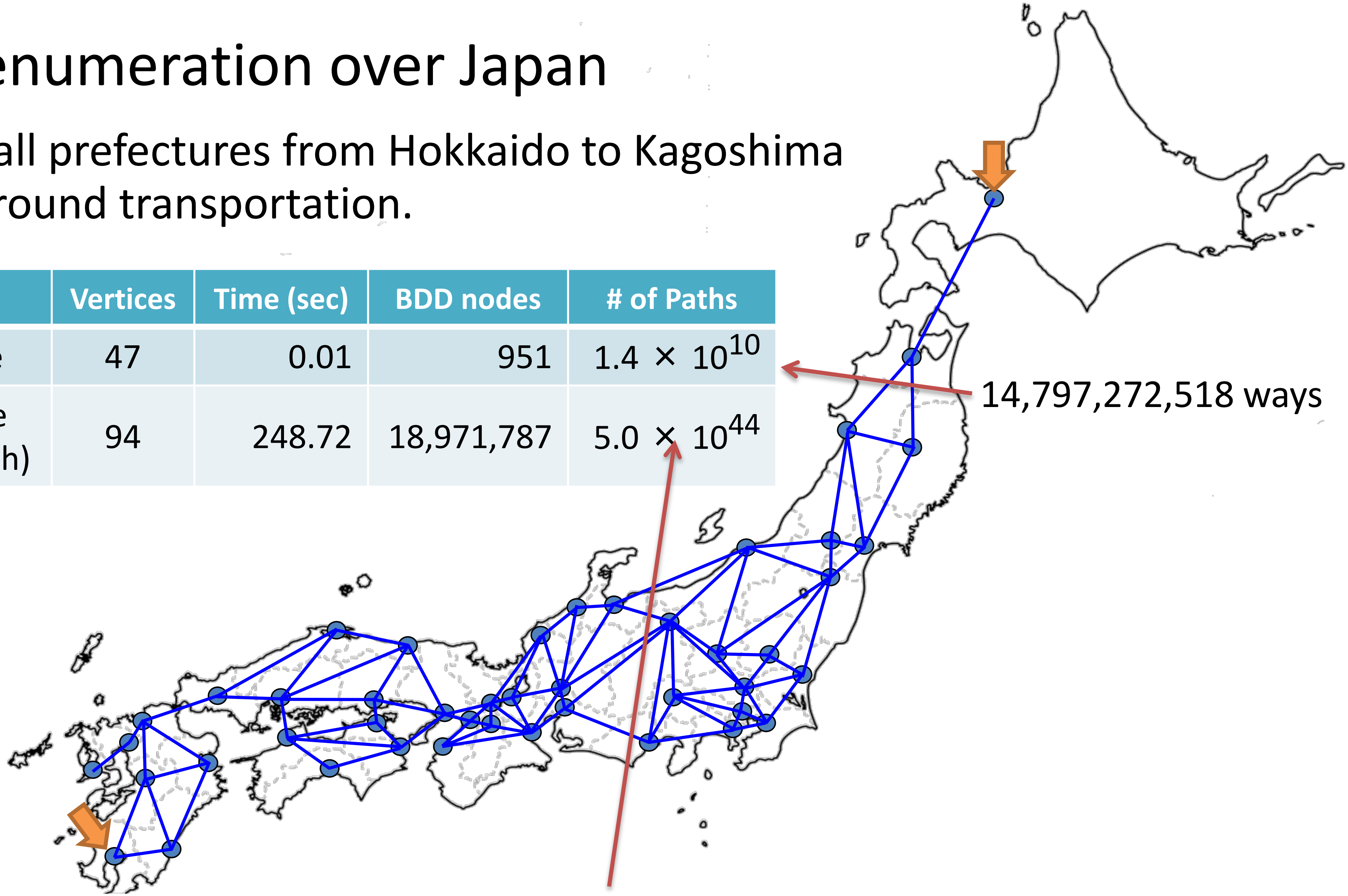
"simpath" for US map in Knuth-book



Path enumeration over Japan

Trip all prefectures from Hokkaido to Kagoshima by ground transportation.

	Vertices	Time (sec)	BDD nodes	# of Paths
Up to once	47	0.01	951	1.4×10^{10}
Up to twice (2-layer graph)	94	248.72	18,971,787	5.0×10^{44}



14,797,272,518 ways

5,039,760,385,115,189,594,214,594,926,092,397,238,616,064 ways

（= 503正9760澗3851溝1518穣9594杼2145垓9492京6092兆3972億3861万6064）

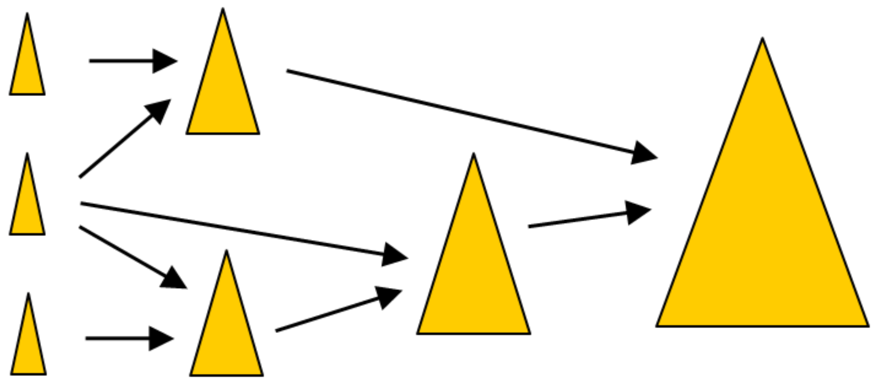
Frontier-based method (generalization of simpath)

- Variation of s-t path problem
  - s-t paths → Hamilton paths (exercise in Knuth-book)
  - paths→ cycles (also in Knuth-book)
  - Non-directed graphs → directed graphs
  - → Multiple s-t pairs (non crossing routing problem)
- Other various graph enumeration problems
  - Subtrees / spanning trees, forests, cutsets, k-partitions, connection probability, (perfect) matching, etc.
- Generating BDDs for Tutte polynomials (graph invariant)
  - We found that Sekine-Imai's idea in 1995 was in principle similar to Knuth simpath algorithm.
  - They used BDDs instead of ZDDs.
  - Enumerating connective subgraphs, not paths.

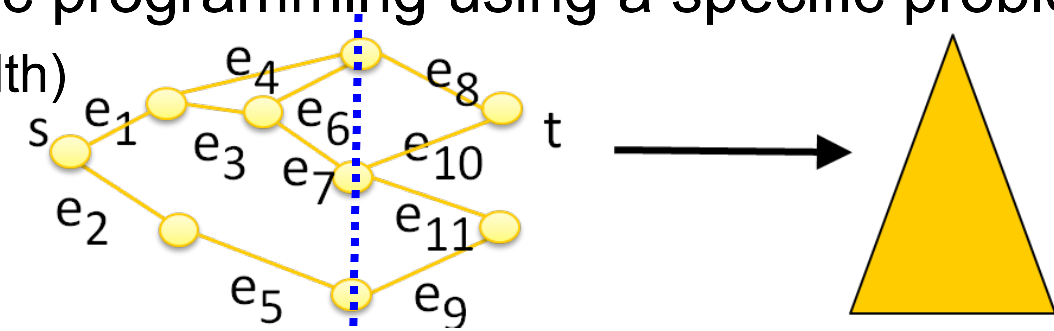
$$T(x,y) = \sum_{A \subseteq E} (x-1)^{\rho(E)-\rho(A)} (y-1)^{|A|-\rho(A)}$$

Comparison with conventional ZDD generation

- Conventional method:
  Repeating logic/set operations between two ZDDs.
  - Based on Bryant's "Apply" algorithm



- Frontier-based method:
  Direct ZDD generation by traversing a given graph.
  - Dynamic programming using a specific problem property.
    (path-width)



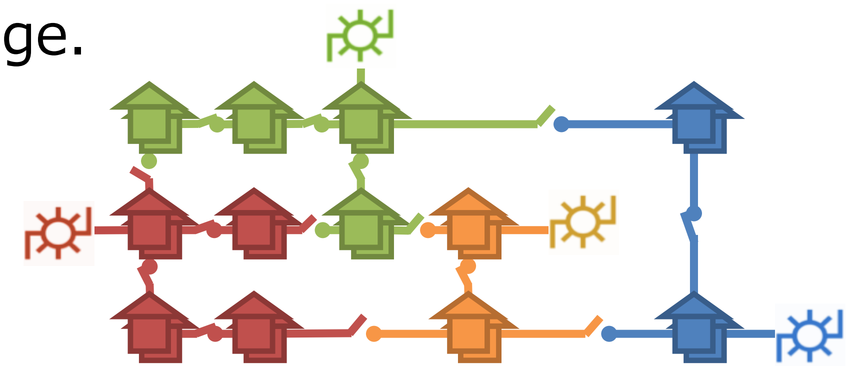
Real-life applications of BDD/ZDD-based techniques

After the Big Earthquake in Japan

- Collaboration with Prof. Hayashi at Waseda Univ.
  - A leader of smart grid technology in power electric community. He receives much more attention after the earthquake.
  - Control of electricity distribution networks are so important after the nuclear plant accident, since solar and wind power generators are not stable.
  - We truly want to contribute something to the society as leading researchers of information technology.
    → We accelerate our collaborative work after the earthquake.

Switching power supply networks:

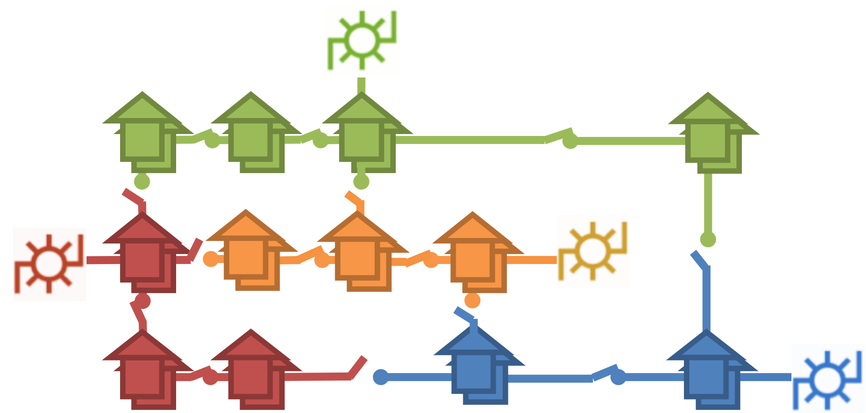
- Each district must connects to a power source (no black out).
- Two power sources must not directly connected.
- Too much currency may burn a line.
- Too long line may cause voltage shortage.



Huge number of patterns.

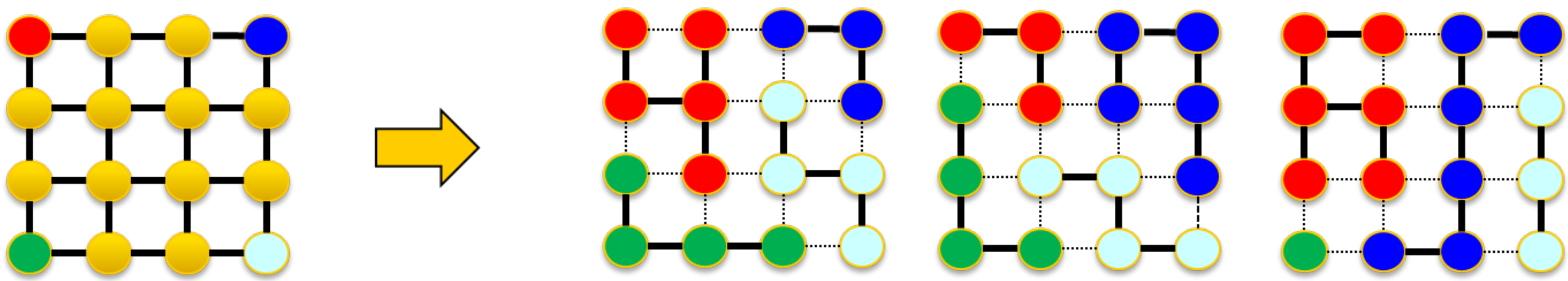
This trivial example has 14 switches.
There are 210 feasible patterns
out of 16384 combinations.

A typical real-life NW has 468 switches.
We have to search the patterns
out of 10^{140} combinations.



Application to power supply networks

- Graph k partition problem (k-cut set enumeration)
  - Enumerate all partitions s.t. given k-vertices not together.
    → Every area is supplied from one power feeder.
  - Frontier-based method is effective.

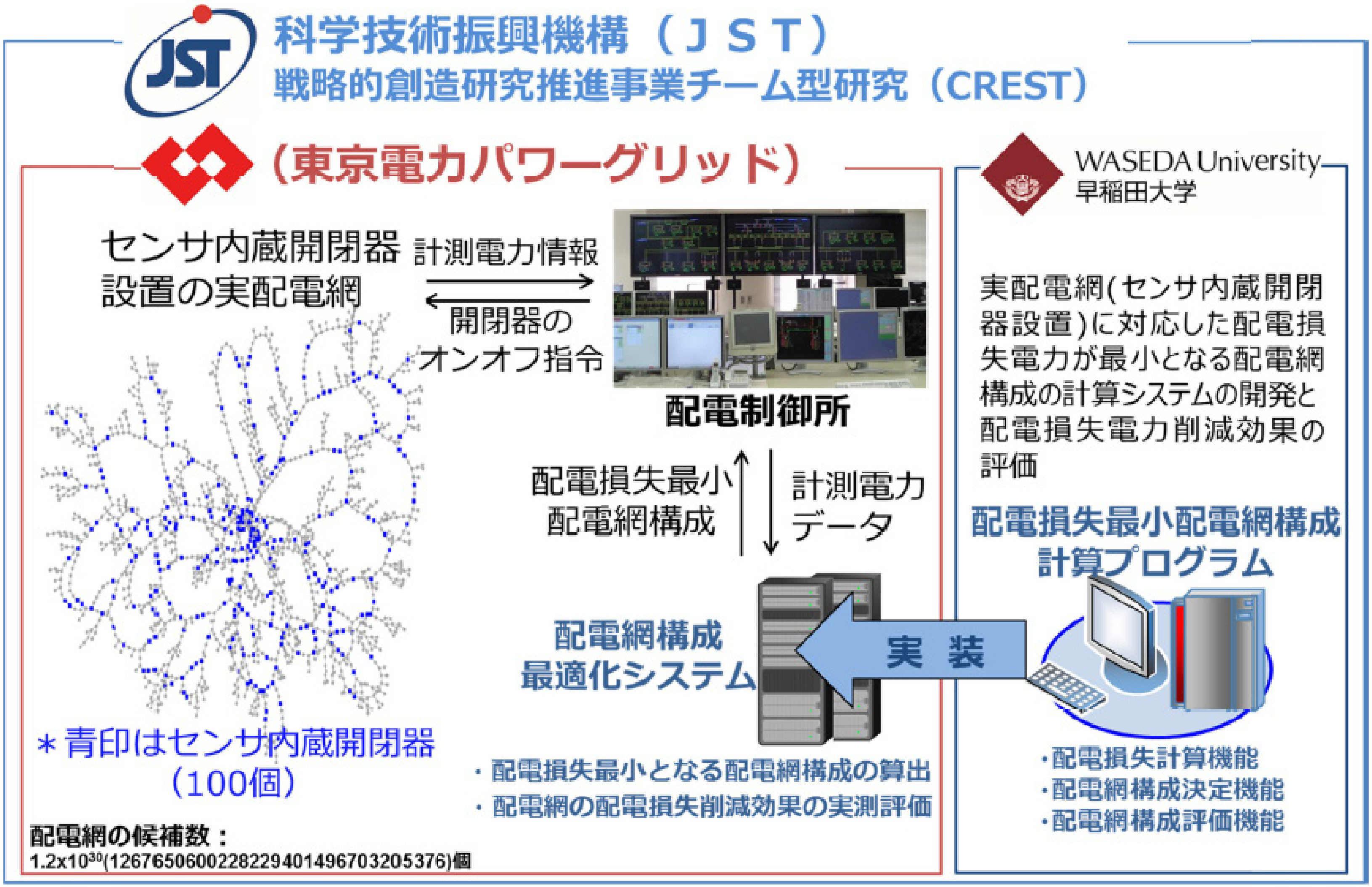


- We succeeded in generating a ZDD of all solutions for a realistic benchmark with 468 control switches.

ZDD nodes: 1.1 million nodes（779MB), CPU time: ~20 min.
Number of solutions: 10^{63}
（2136820138348532911682612214804956098178392443852353981895215 40）

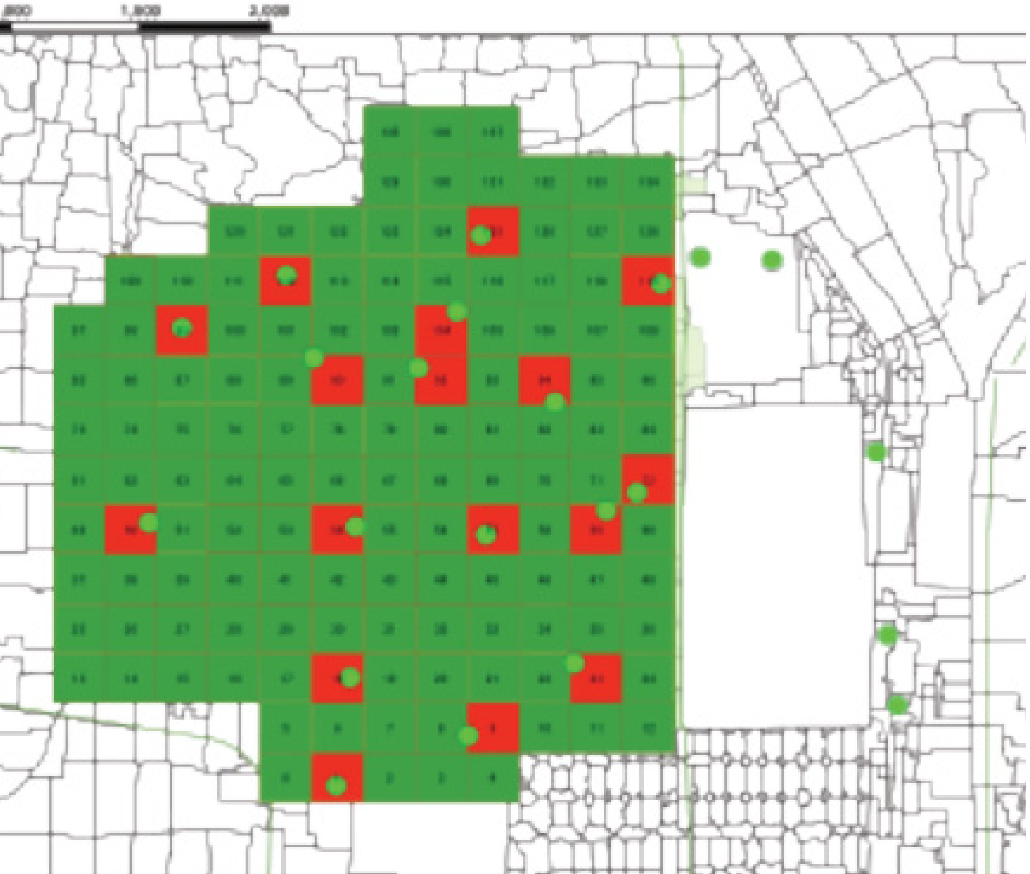
Collaboration with Electric Power Company

- Press-release from TEPCO (Tokyo Electric) on April 2016
  - Experiment on the real network for minimizing energy loss.



Layout of refuge places in Kyoto city

- Very similar algorithm as the power distribution problem.
  - Collaboration with Prof. Naoki Kato at Kyoto Univ.
  - Presented at ISORA 2013 (Int'l Conf. on OR)



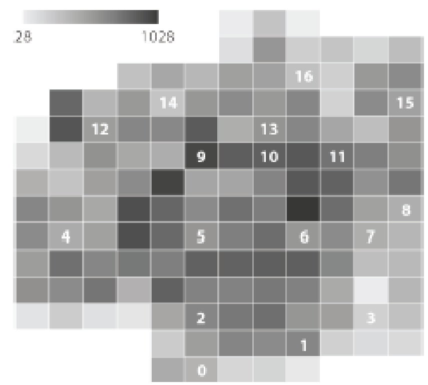


Figure 13: Number of population in each cell.

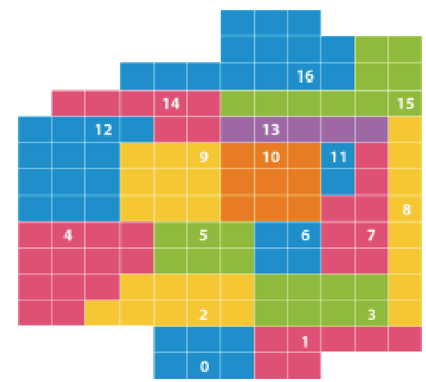


Figure 14: The optimal pattern with respect to *Criterion 1*.

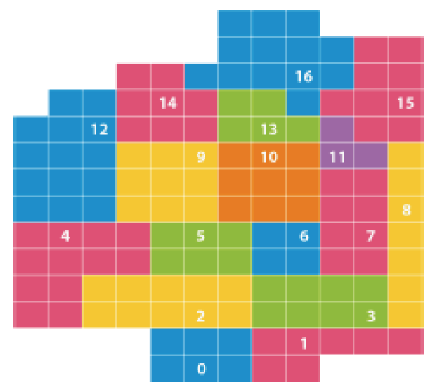


Figure 15: The optimal pattern with respect to *Criterion 2*.

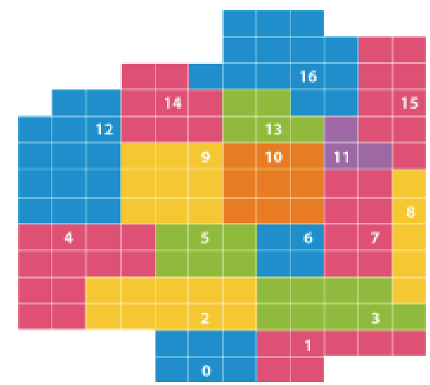
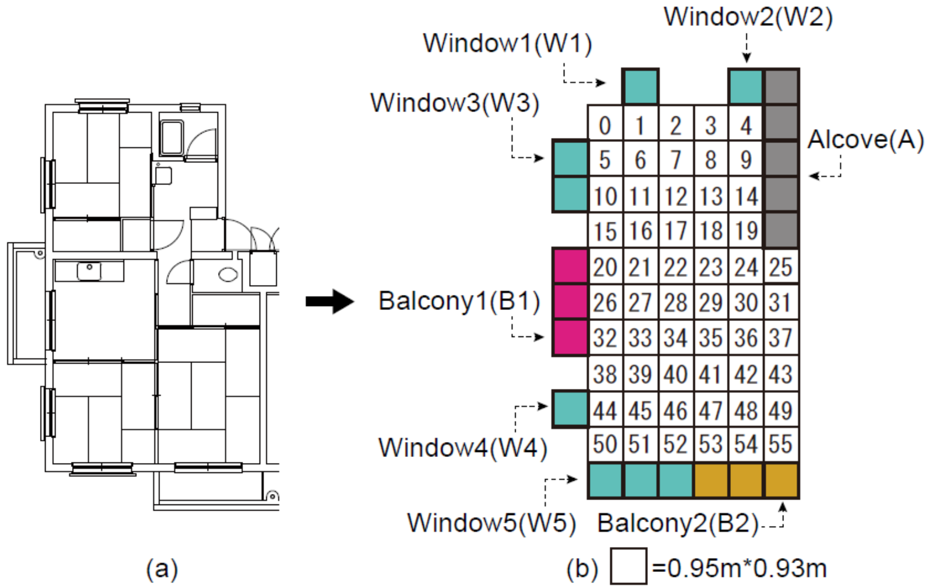
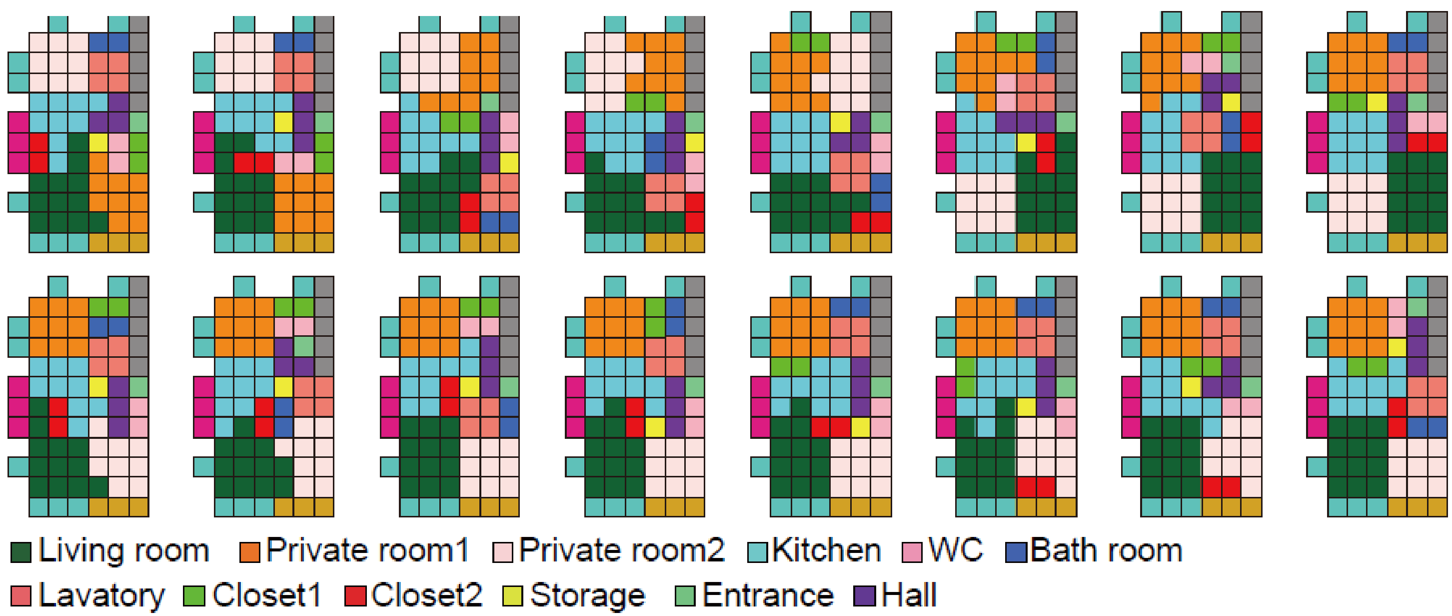


Figure 16: One of the pareto

House floor planning



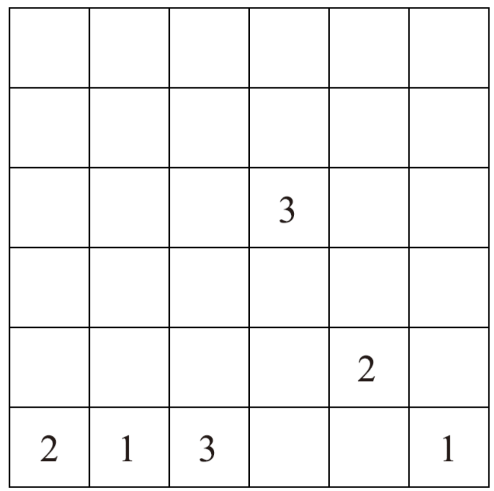
- Collaborative work with Prof. Takizawa at Osaka City Univ.
- Best paper award in CAADRIA2014, an int'l conf. on building architecture.

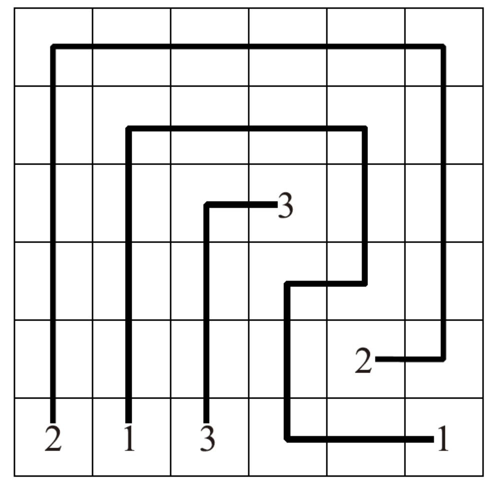


Application to Pencil Puzzles

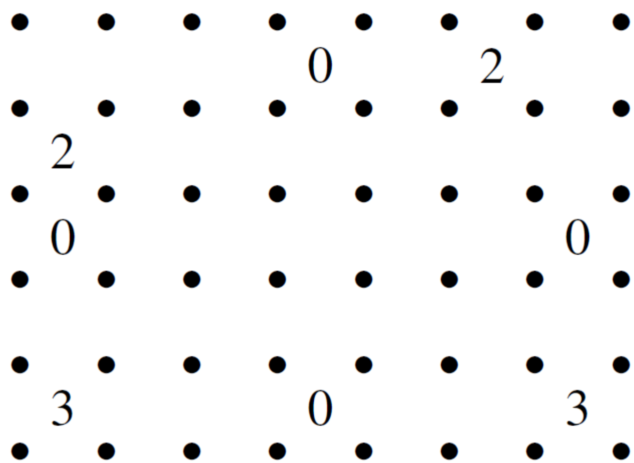
- "Finding All Solutions and Instances of Numberlink and Slitherlink by ZDDs" [Yoshinaka et al. 2012]

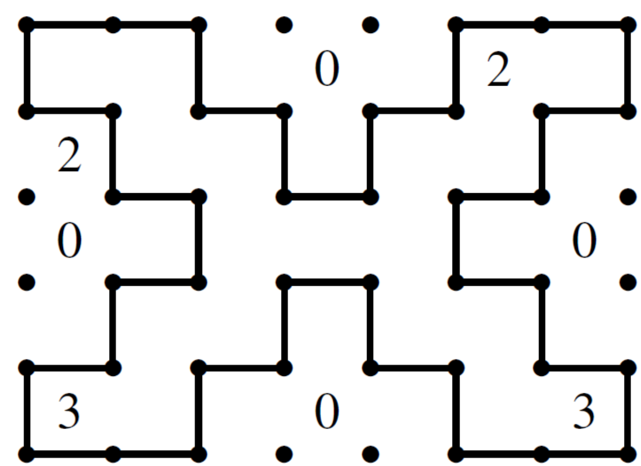
Numberlink:





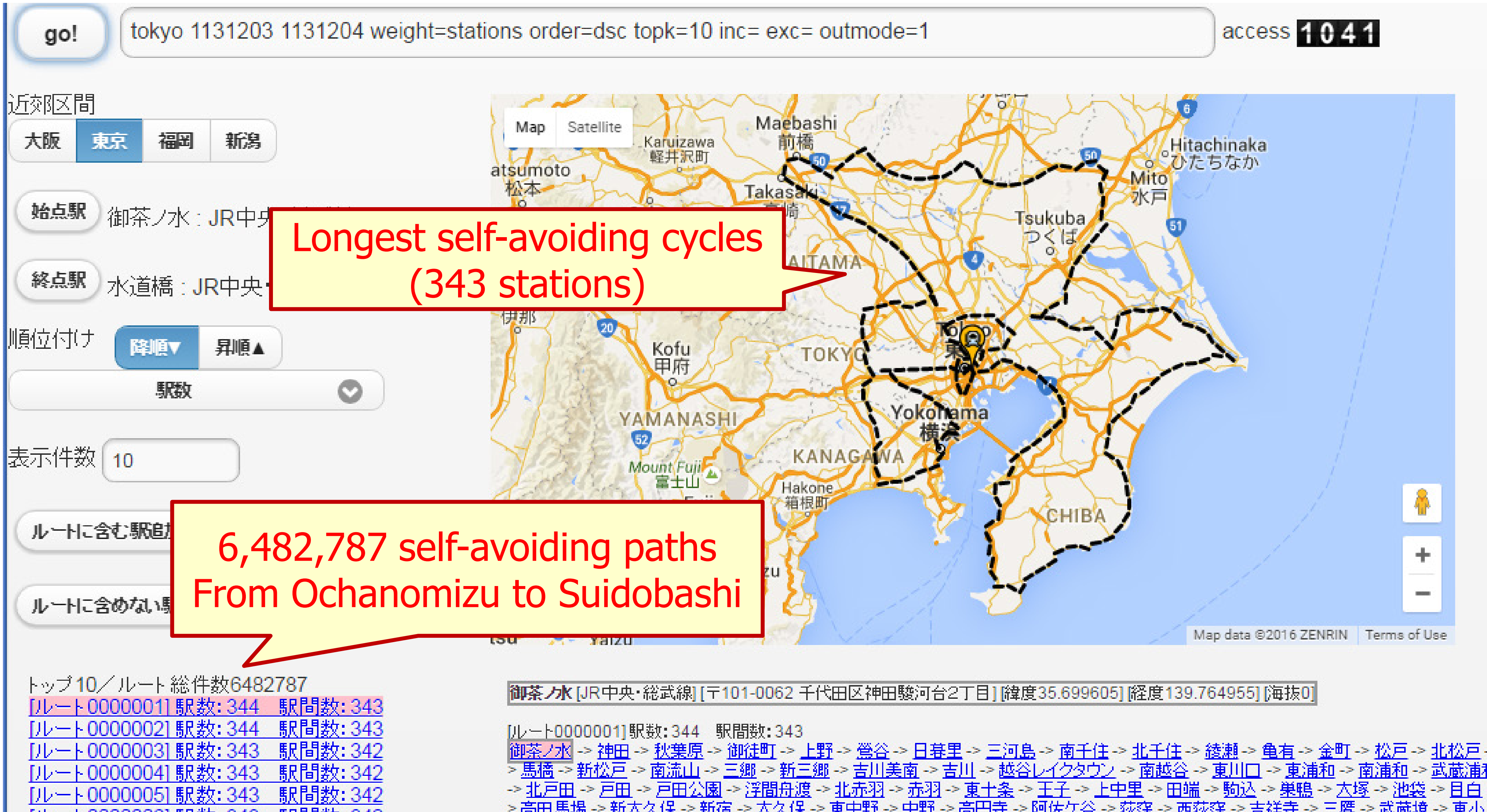
Slitherlink:





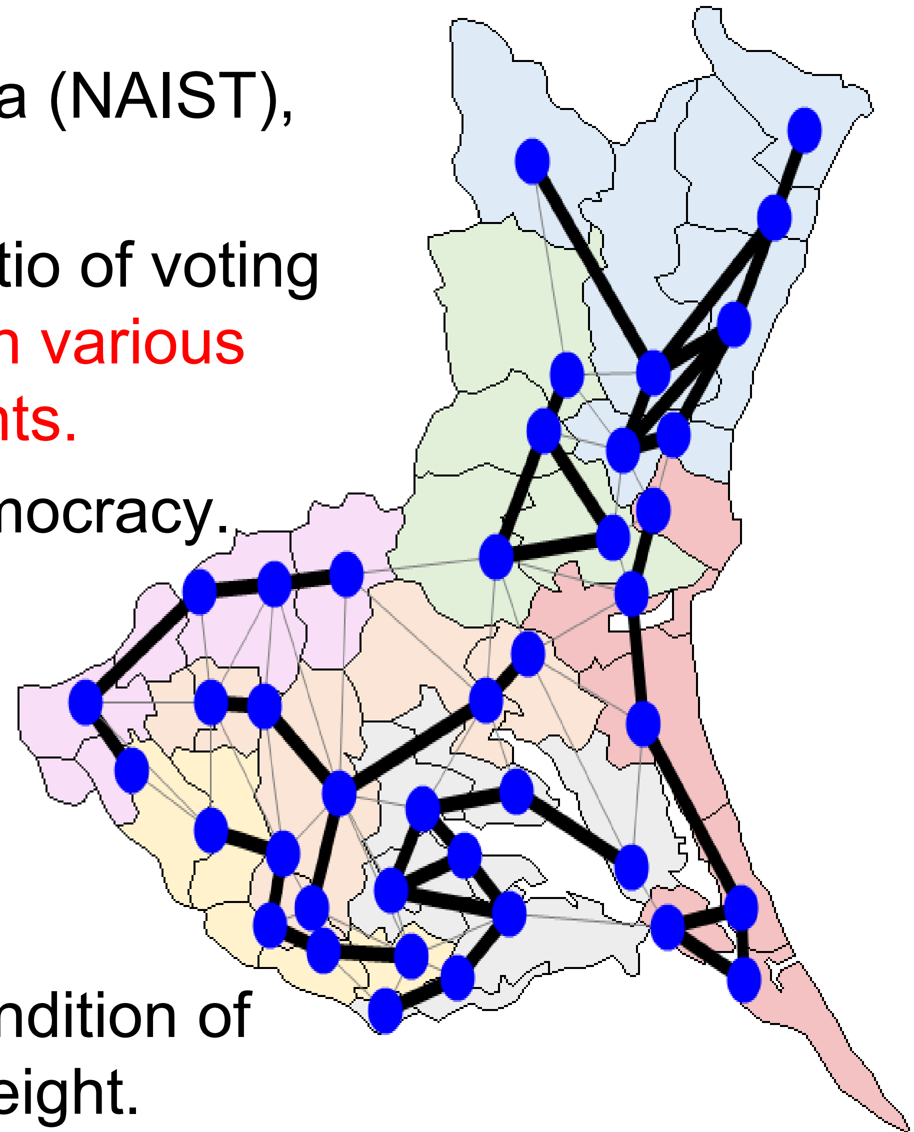
Railway route search and path enumeration

- Enumerating all self-avoiding paths in Tokyo area.



Partitioning electoral districts

- Collaboration with Prof. Kawahara (NAIST), Prof. Hotta (Bunkyo U.), et al.
- Mal-apportionment (difference-ratio of voting weights) should be minimum, with various geographical and social constraints.
- Important problem to support democracy.



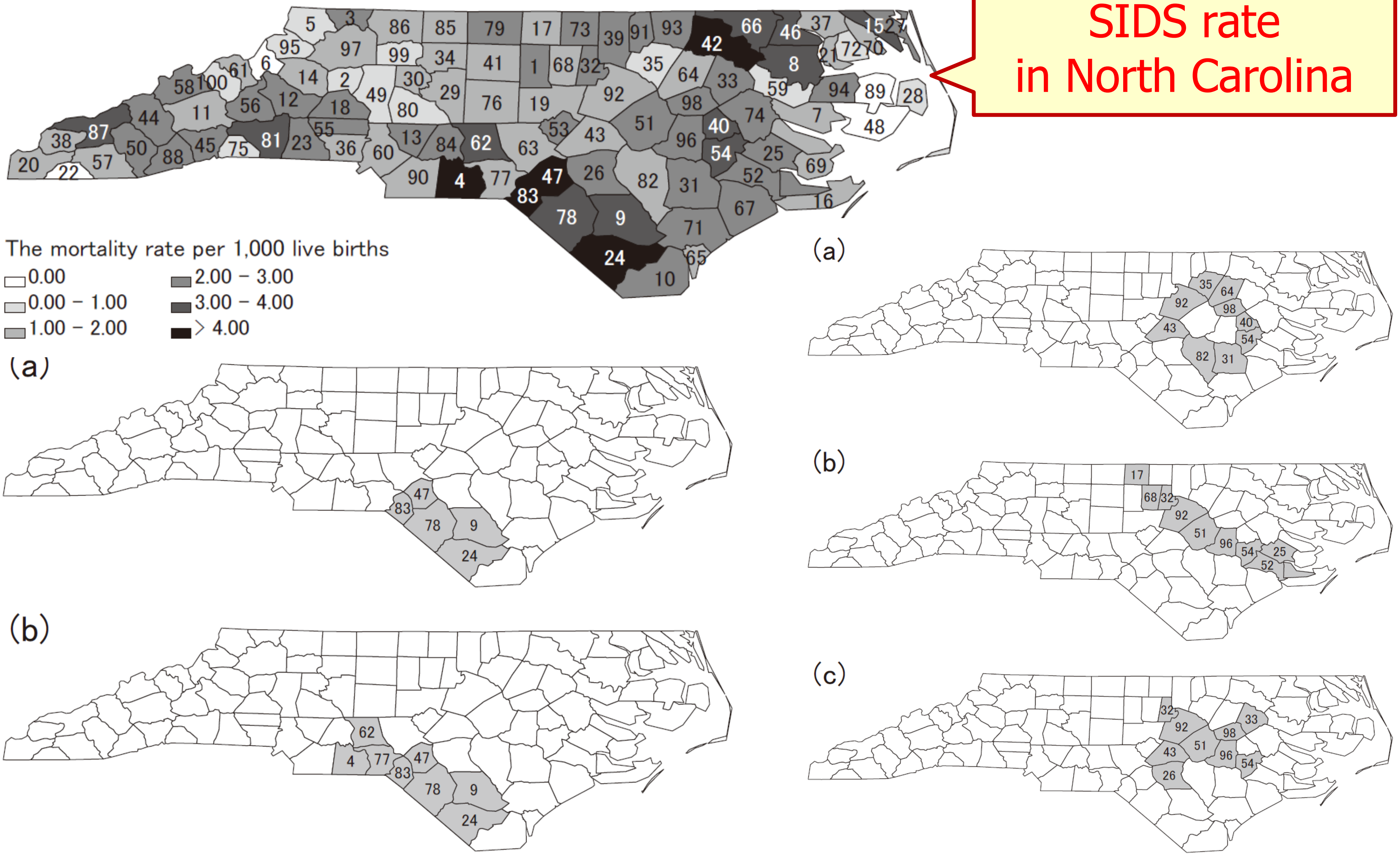
Example on Ibaraki-Pref., Japan:
41 vertices, 87 edges, 7 partitions.
(→ 41 city units into 7 districts)

All solutions: 11,893,998,242,846

25,730,669 solutions satisfying the condition of 1.4 or less difference-ratio of voting weight.
(CPU time: 1925.21 sec)

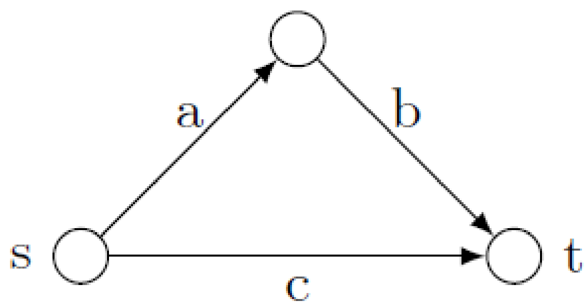
Hotspot extraction from geographical statistics

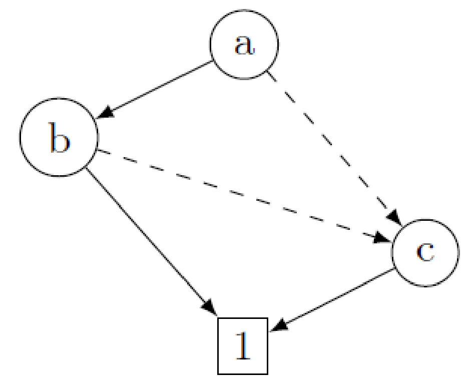
SIDS rate in North Carolina



Statistical data analysis using BDDs

- **Exact Computation of Influence Spread by BDDs**
  Presented at WWW2017 by Maehara et al.





Network	Vertices	Edges	Time [ms]	BDD Size	Shared Size	Cardinality
South-African-Companies	11	26	0.1	12.1	472	2.2e+07
Southern-women-2	20	28	0.3	54.7	2,266	1.3e+08
Taro-exchange	22	78	4.1	1,119.2	277,756	1.6e+23
Zachary-karate-club	34	156	24.9	7,321.8	4,988,148	6.4e+46
Contiguous-USA	49	214	117.9	30,599.8	41,261,047	1.6e+64
American-Revolution	141	320	2.2	120.0	1,530,677	5.7e+95
Southern-women-1	50	178	—	—	—	—
Club-membership	65	190	—	—	—	—
Corporate-Leadership	64	198	—	—	—	—

Statistically significant pattern mining

- **We proposed a novel p-value correction procedure "LAMP." [PNAS: Terada et al. 2013]**
  - Accurate statistical assessment to discover combinatorial factors.
  - Based on frequent itemset enumeration techniques.
- Our result shows that:
  state-of-the-art enumeration techniques can contribute to many kinds of experimental sciences.

PNAS

Statistical significance of combinatorial regulations

Aika Terada[a,b,c], Mariko Okada-Hatakeyama[d], Koji Tsuda[c,e,1], and Jun Sese[a,b,1]

[a]Department of Computer Science and [b]Education Academy of Computational Life Sciences, Tokyo Institute of Technology, Meguro-ku, Tokyo 152-8550, Japan; [c]Minato Discrete Structure Manipulation System Project, Exploratory Research for Advanced Technology, Japan Science and Technology Agency, Sapporo, Hokkaido 060-0814, Japan; [d]Laboratory for Integrated Cellular Systems, RIKEN Center for Integrated Medical Sciences (IMS-RCAI), Yokohama, Kanagawa 230-0045, Japan; and [e]Computational Biology Research Center, National Institute of Advanced Industrial Science and Technology, Koto-ku, Tokyo 135-0064, Japan

Edited by Wing Hung Wong, Stanford University, Stanford, CA, and approved July 3, 2013 (received for review February 4, 2013)

More than three transcription factors often work together to enable cells to respond to various signals. The detection of combinatorial regulation by multiple transcription factors, however, is not only

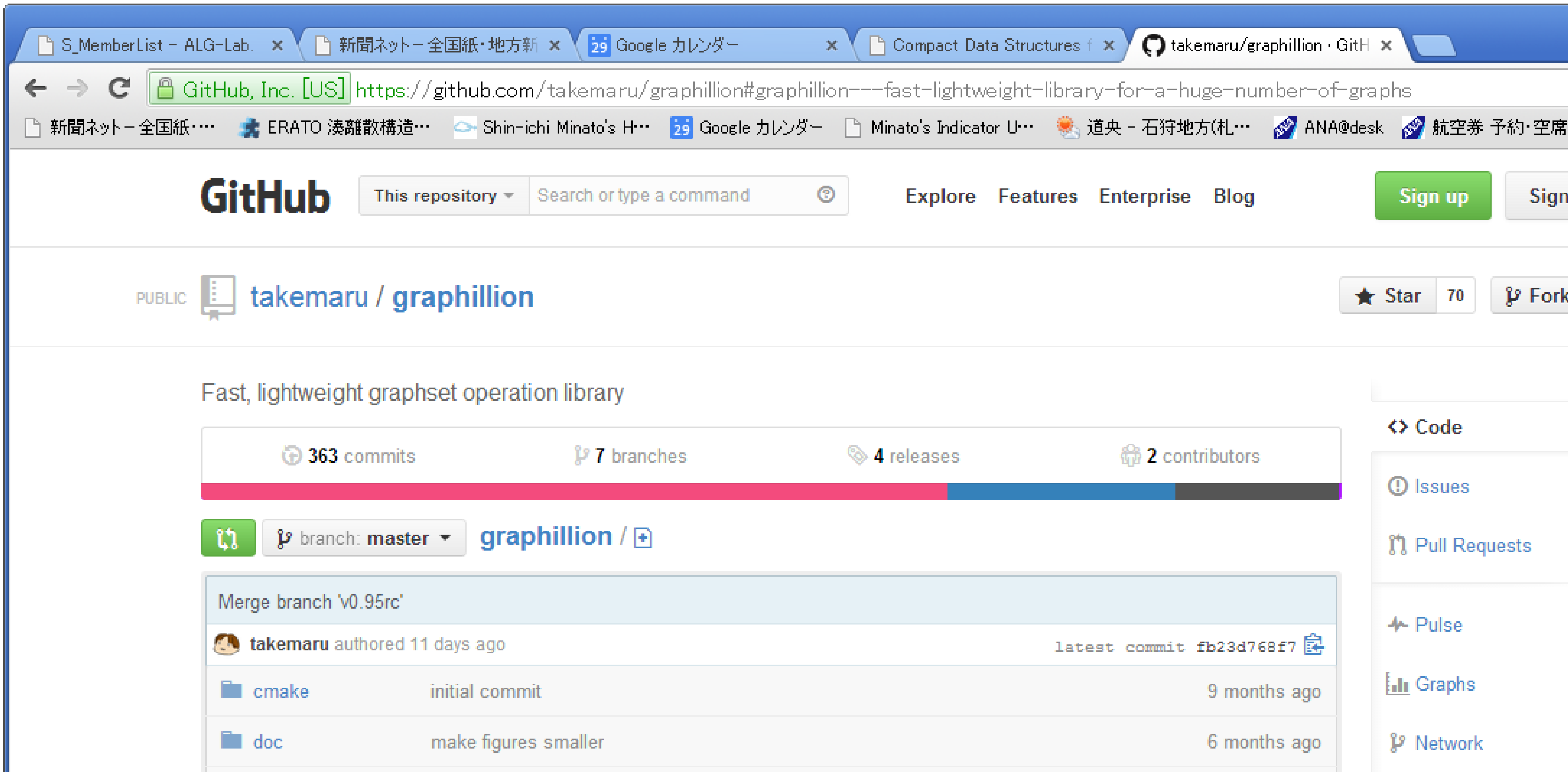
deliberately excluding such tests. Here, we propose an efficient branch-and-bound algorithm, called the "limitless arity multiple-testing procedure" (LAMP). LAMP counts the exact number of

Advantages of generating ZDDs

- Not only enumeration but also giving an index structure.
- Not only indexing but also providing rich operations.
- Well-compressed structure for many practical cases.
- Related to various real-life important problems.
  - GIS (car navigation, railway navigation)
  - Dependency/Fault analysis industrial systems
  - Solving puzzles (Numberlink, Slitherlink, etc.)
  - Enumerating all possible concatenations of substrings
  - Control of electric power distribution networks
  - Layout of refuge shelters for earthquake and tsunami
  - Design of electoral districts for democratic fairness

Open software: “Graphillion.org”

- Toolbox for ZDD-based graph enumeration.
  - Easy interface using Python graph library.



Tutorial video for “Graphillion”

LINE stickers of our YouTube video !

Available from Oct. 2015

Summary

- Focus on BDD/ZDD-based enumeration techniques.
  - Representing “logic” and “set,” primitive models of discrete structures.
  - Efficient algebraic operations without de-compression.
  - Starting from VLSI CAD in 1990s, but now widely used.
- Recent results
  - Demonstration video: **1.9 million views!**
  - Enumerating all solutions for various types of graph problems:
    → “Knowledge Compilation” for finding “good” solutions.
  - Many practical applications.
    - Power distribution network, railways, water/gas supply, etc.
- Visit “Graphillion.org” to see our toolbox.