Towards Non-monotone Dualization in BDD/ZDD

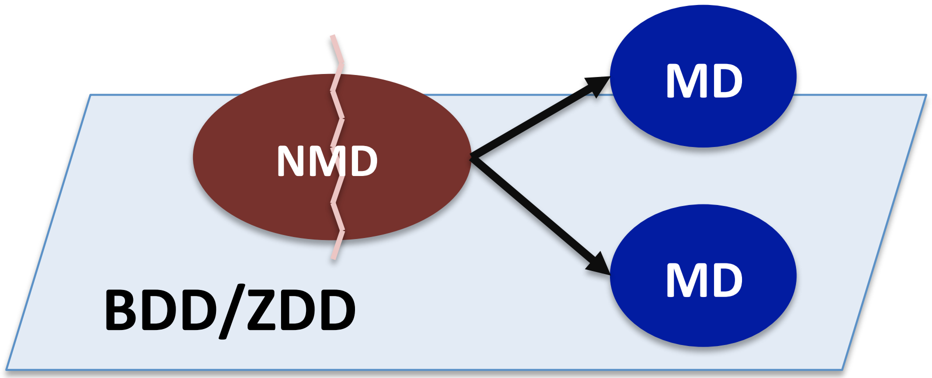
Y. Yamamoto[1], K. Iwanuma[1] and H. Nabeshima[1]

[1]University of Yamanashi

The 3rd CSPSAT2 Meeting

2013/7/26

Contents



- Background
  - Non-monotone Dualization (NMD)
  - Monotone Dualization (MD)
  - Reduction technique from MND to *two* MDs
- State-of-the art MD computation
  - Enumeration-tree based method
  - BDD/ZDD based method
- A preliminary evaluation of them for NMD
- (*Ideas* for performing NMD over ZDD)

Dualization problem

Input: a tautology-free CNF formula φ

Output: an irredundant prime CNF formula ψ s.t.

ψ is logically equivalent to the dual φ^d

$$\varphi = x_2 \wedge (x_1 \vee x_2) \wedge (\neg x_2 \vee x_3)$$

$$\varphi^d = x_2 \vee (x_1 \wedge x_2) \vee (\neg x_2 \wedge x_3) \quad \text{[The dual of } \varphi\text{]}$$

Converting DNF to CNF

$$\psi_1 = (x_2 \vee x_1 \vee \neg x_2) \wedge (x_2 \vee x_1 \vee x_3) \wedge (x_2 \vee \neg x_2) \wedge (x_2 \vee x_3)$$

$$\psi_2 = (x_2 \vee x_1 \vee \neg x_2) \wedge (x_2 \vee x_1 \vee x_3) \wedge (x_2 \vee \neg x_2) \wedge (x_2 \vee x_3)$$

$$\psi_3 = (x_2 \vee x_1 \vee \neg x_2) \wedge (x_2 \vee x_1 \vee x_3) \wedge (x_2 \vee \neg x_2) \wedge (x_2 \vee x_3)$$

Terminologies

– φ is a **prime** CNF (*resp*. DNF) iff

there is no literal l of a clause (*resp*. term) C in φ s.t.

$$\boldsymbol{\varphi \equiv (\varphi - \{C\}) \cup \{C - \{l\}\}.}$$

– φ is an **irredundant** CNF (*resp*. DNF) iff

there is no clause C (*resp*. term) in φ s.t.

$$\boldsymbol{\varphi \equiv \varphi - \{C\}.}$$

- **Example**

  Let φ_1 and φ_2 be the following CNF formulas:

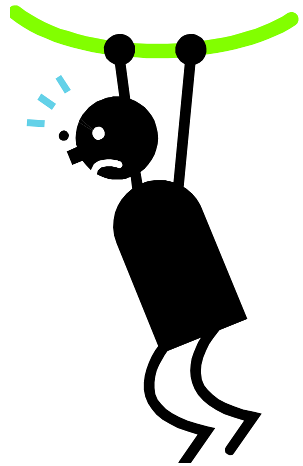
$$\varphi_1 = (a \lor b) \land (c \lor \neg b) \land (a \lor c),$$

$$\varphi_2 = (a \lor b) \land (a \lor \neg b).$$

φ_1 is redundant but prime.

φ_2 is non-prime but irredundant.

Problem difficulty



- Task: DNF-CNF conversion

  Ex) CNF ⇒ a prime irredundant DNF:

  Enumerating the most compact *models*

  – **Harder** than SAT problems

  – Characteristic *sub-classes* have been much focused

- Monotone dualization

  – Targets: ***monotone*** Boolean functions
  (i.e., no negations occur in their formulas)

  – Equivalent to the ***minimal hitting set enumeration*** and the transversal hypergraph computation

  – Tractability w.r.t. polynomial time: ***unknown***

Applications of dualization

- **Learning from interpretations**:
  used for seeking underlying *concept* (CNF) behind the *models* (DNF) [L. DeRaedt, 97]
- **Learning from entailment**:
  used in several procedures based on inverse entailment,
  like CF-induction [K. Inoue, 04; Y. Yamamoto, 11] and
  Residue procedure [A. Yamamoto, 03]

Ex) Suppose that a set of models M are given:

M = { (bird $\wedge$ normal $\wedge$ flies), ($\neg$flies $\wedge$ $\neg$normal), ($\neg$flies $\wedge$ $\neg$bird) }.

By dualizing M, we have the following CNF formula:

H = (bird $\vee$ $\neg$flies) $\wedge$ (normal $\vee$ $\neg$flies) $\wedge$ (flies $\vee$ $\neg$normal $\vee$ $\neg$bird).

MD vs. NMD (1/2)

- *Monotone* dualization (**MD**):
  - Solvable in quasi-polynomial total time [Fredman&Khchiyan 96']
  - Outputs contain no *tautologies* and *resolvents*
- *Non-monotone* dualization (**NMD**):
  - NP-hard
  - Outputs *can* contain tautologies and resolvents

Ex) Let a CNF $\varphi = (x_1 \vee x_2) \wedge (\neg x_2 \vee x_3)$.

By treating negated variables as regular variables, we can apply MD to φ, which yields the following CNF:

$\psi = (x_1 \vee \neg x_2) \wedge (\mathbf{x_1 \vee x_3}) \wedge (\mathbf{x_2 \vee \neg x_2}) \wedge (x_2 \vee x_3)$

⇒ It is *not* straightforward to use MD for NMD computation

MD vs. NMD (2/2)

- *Monotone* dualization (**MD**):
  - The output of MD is **unique**
- *Non-monotone* dualization (**NMD**):
  - The output of NMD is *not necessarily* **unique**

Ex) Let a CNF $\varphi = (x_1 \vee \neg x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_2 \vee x_3)$.

By treating negated variables as regular variables, we can apply MD to φ, which yields the following CNF:

$$\underbrace{(x_1 \vee x_2) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_3)}_{\psi_1} \wedge \underbrace{(x_1 \vee x_3) \wedge (\neg x_1 \vee \neg x_2) \wedge (\neg x_3 \vee x_2)}_{\psi_2}.$$

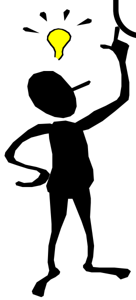
⇒ Both ψ_1 and ψ_2 are equivalent and prime irredundant CNF

NMD via MD [Y. Yamamoto, 2012]

Investigating NMD in the point of MD computation

- **Main result**:

NMD problem can be reduced into **two** MD problems



Key idea: adding *tautologies* to the input in advance

Key notions

- $MD(\varphi)$: the CNF obtained by applying MD to φ
- $\tau(\varphi)$: the CNF obtained by removing the tautologies in φ
- $Taut(\varphi)$: the CNF $(\boldsymbol{l_1} \vee \neg \boldsymbol{l_1}) \wedge \cdots \wedge (\boldsymbol{l_n} \vee \neg \boldsymbol{l_n})$ where l_i and $\neg l_i$ $(1 \leqq i \leqq n)$ are complementary literals in φ.

Ex) Let a CNF $\varphi = \{ \{a, b\}, \{\neg a, c\} \}$. Then,

$$MD(\varphi) = \{ \{a, \neg a\}, \{a, c\}, \{b, \neg a\}, \{b, c\} \},$$

$$\tau(MD(\varphi)) = \{ \{a, c\}, \{b, \neg a\}, \{b, c\} \},$$

$$Taut(\varphi) = \{ \{a, \neg a\} \}.$$

$$MD(\varphi \cup Taut(\varphi)) = \{ \{a, \neg a\}, \{a, c\}, \{b, \neg a\} \},$$

$$\tau(MD(\varphi \cup Taut(\varphi))) = \{ \{a, c\}, \{b, \neg a\} \}.$$

Redundant clauses are removed

Main theorem

ψ is an irredundant prime CNF of φ^d $\Leftrightarrow$ ψ satisfies

1. $\psi \subseteq \tau(\mathrm{MD}(\varphi))$

2-1. $\psi \succeq \tau(\mathrm{MD}(\varphi \cup \mathrm{Taut}(\varphi)))$ ← We call it ``Bottom theory''

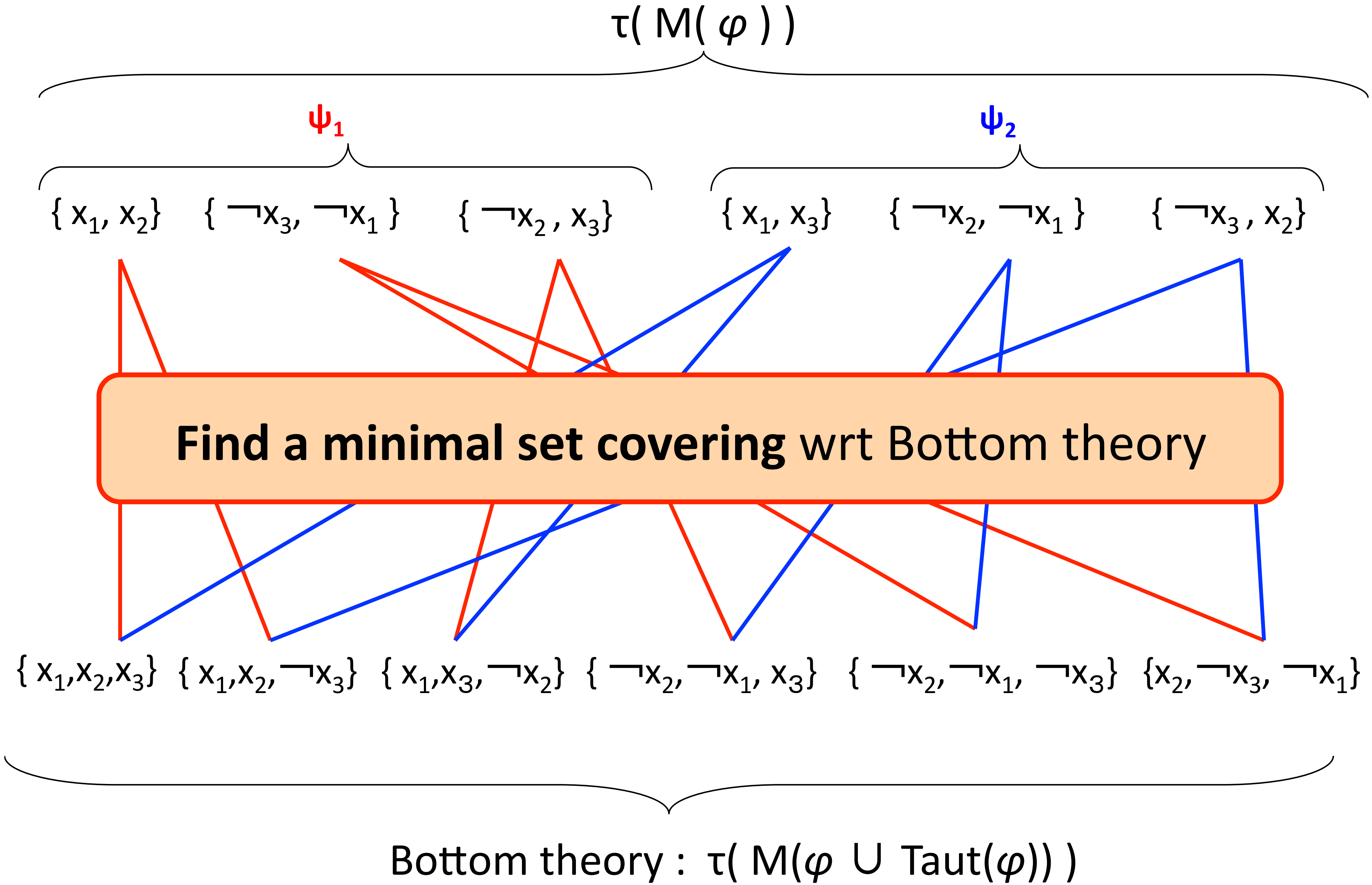
2-2. $\forall C \in \psi, \psi - \{C\} \not\succeq \tau(\mathrm{MD}(\varphi \cup \mathrm{Taut}(\varphi)))$

1: Every clause in ψ is prime implicate of φ^d

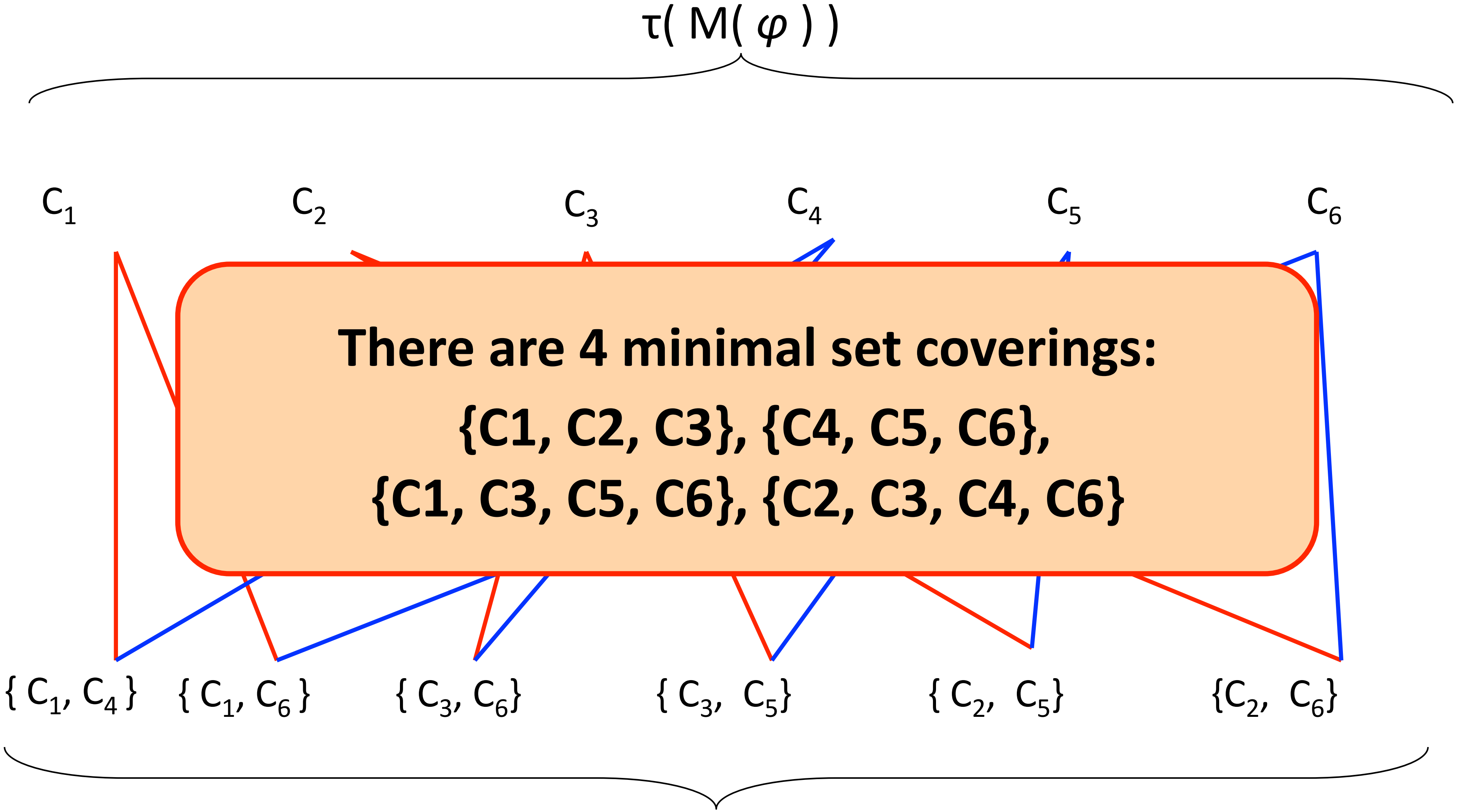
2-1: Every clause in the Bottom theory is subsumed by some clause in ψ

2-2: Every clause in ψ *uniquely* subsumes some clause in the Bottom theory

Ex) Recall the CNF $\varphi = \{ \{x1, \neg x2, \neg x3\}, \{\neg x1, x2, x3\} \}$.



Ex) Recall the CNF $\varphi = \{ \{x1, \neg x2, \neg x3\}, \{\neg x1, x2, x3\} \}$.



There are 4 minimal set coverings:

{C1, C2, C3}, {C4, C5, C6},

{C1, C3, C5, C6}, {C2, C3, C4, C6}

Bottom theory : $\tau(M(\varphi \cup Taut(\varphi)))$

Our approach

Stage 1. Computing the Bottom theory by **MD**

- τ(M(φ)) and τ(MD(φ ∪ Taut(φ))) are obtained

Our approach

Stage 1. Computing the bottom theory by **MD**

- τ(M(φ)) and τ(MD(φ ∪ Taut(φ))) are obtained

Stage 2. Creating the minimal set covering problem (MSC) using the bottom theory

Our approach

Stage 1. Computing the bottom theory by **MD**

- τ(M(φ)) and τ(MD(φ ∪ Taut(φ))) are obtained

Stage 2. Creating the minimal set covering problem (MSC) using the bottom theory

Stage 3. Computing a MHS of MSC by **MD**

Our approach

Stage 1. Computing the bottom theory by **MD**

– τ(M(φ)) and τ(MD($\varphi \cup$ Taut(φ))) are obtained

Stage 2. Creating the minimal minimal set covering problem (MSC) using the bottom theory

Stage 3. Computing a MHS of MSC by **MD**

We can use *state-of-the-art* MD methods

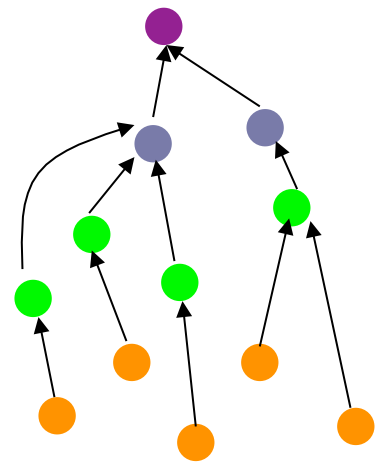
– Enumeration-tree based method [Uno 02']

– BDD/ZDD based method [Toda 13']

Enumeration tree based method [Uno, 02']

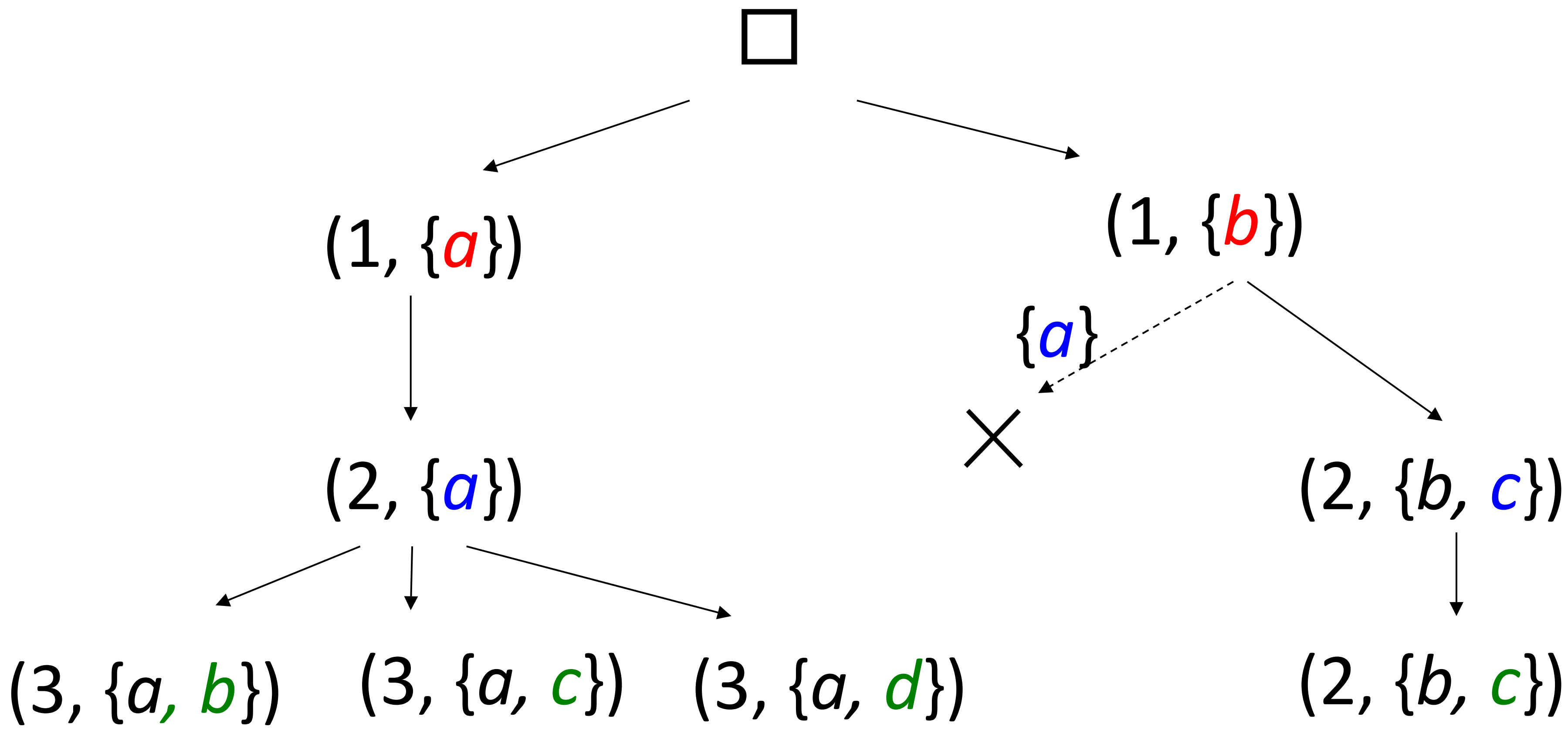
- Based on the principle of ***reverse search***
  - Introducing a relevant relation, called a **parent-child relationship** in the solution space
    - No ancestor solutions on the relationship that is equal to the original solution
    - This relationship is used to induce a search tree, called **enumeration tree**
  - Using an enumeration tree, we search

all the solutions with the depth-first strategy



Example

Let F_3 the family { {*a, b*}, {*a, c*}, {*b, c, d*} }.
Then, the enumeration tree is as follows:



BDD/ZDD based method [Toda & Minato 13']

Consisting 4 procedures each of which is done in BDD/ZDD

1. Compute a ZDD *p* that corresponds to a set family F ;
   (using Toda's fast construction algorithm)
2. Compute a BDD *q* that corresponds to the HSs of *S*(*p*), where *S*(*p*) is the set family corresponding to *p;*
3. Compute a ZDD *r* that corresponds to the MHSs of *S*(*q*);
4. Output the S(*r*) by exporting the ZDD *r* .

BDD/ZDD based method [Toda & Minato 13']

Consisting 4 procedures each of which is done in BDD/ZDD

1. Compute a ZDD *p* that corresponds to a set family F ; (using Toda's fast construction algorithm)
2. Compute a BDD *q* that corresponds to the HSs of *S*(*p*), where *S*(*p*) is the set family corresponding to *p;*
3. Compute a ZDD *r* that corresponds to the MHSs of *S*(*q*);
4. Output the S(*r*) by exporting the ZDD *r* .

Example

Assume that every set is listed by the alphabetical order

- Let F_3 the family { {*a*, *b*}, {*a*, *c*}, {*b*, *c*, *d*} }.

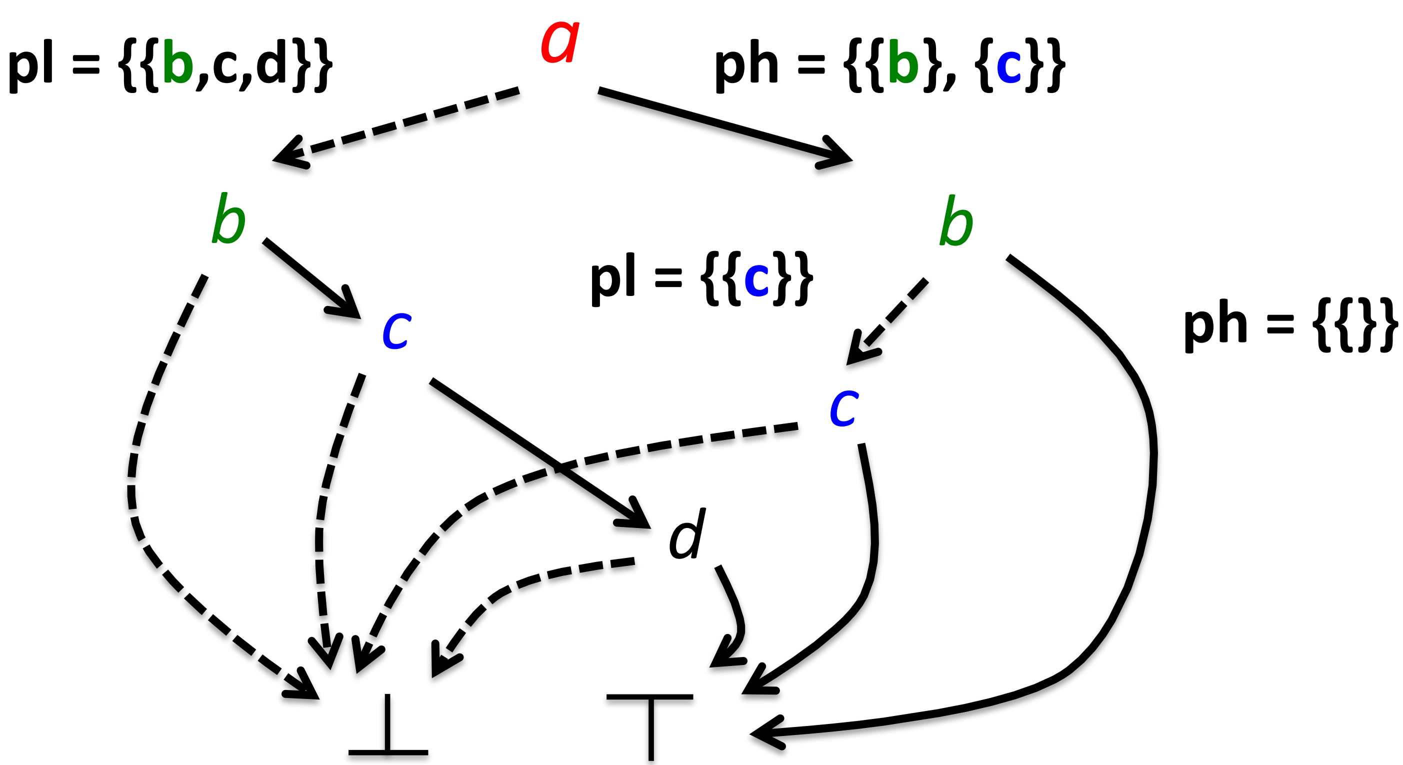
1. Compute a ZDD r from F_3 as follows:

Example

Assume that every set is listed by the alphabetical order

- Let F_3 the family { {*a*, *b*}, {*a*, *c*}, {*b*, *c*, *d*} }.

1. Compute a ZDD r from F_3 as follows:

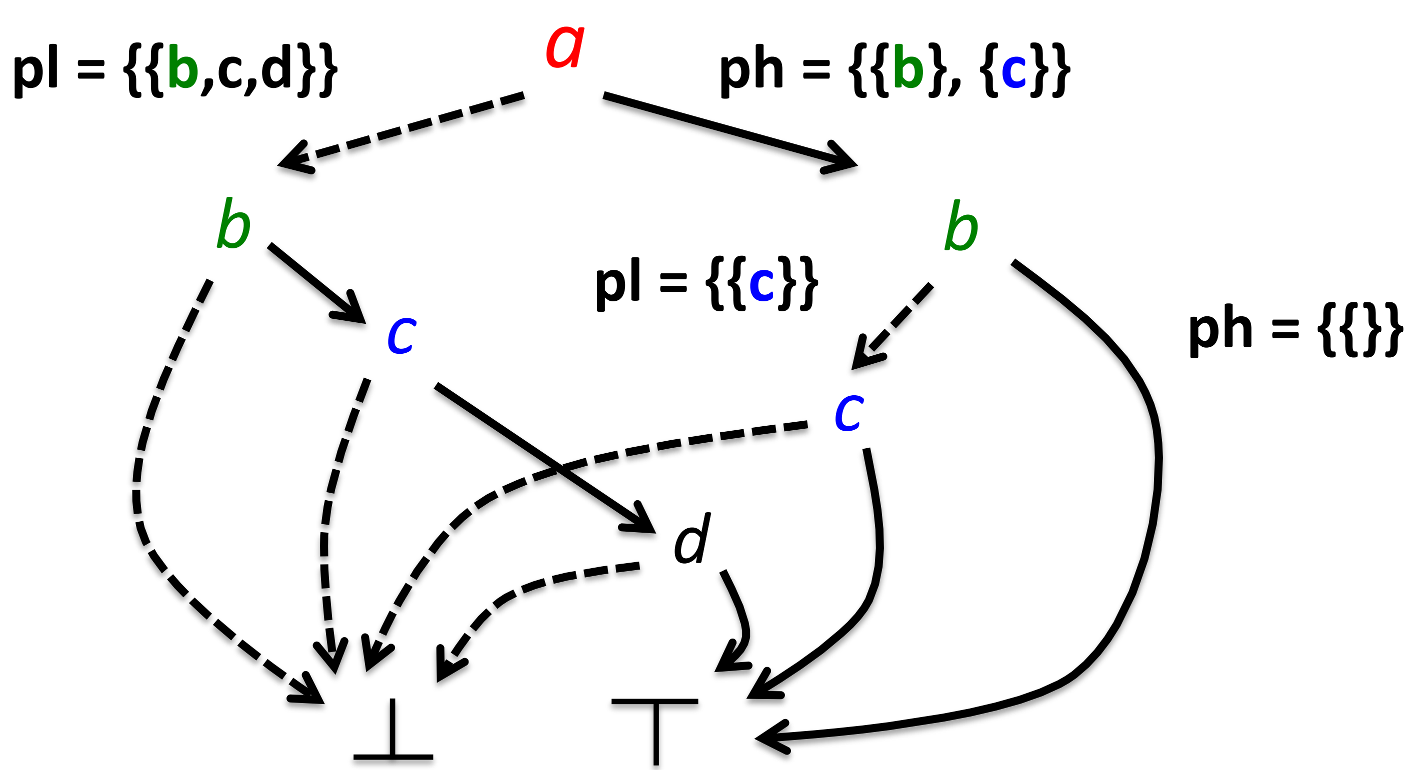


Example

Assume that every set is listed by the alphabetical order

- Let F_3 the family { {*a*, *b*}, {*a*, *c*}, {*b*, *c*, *d*} }.

2. Compute a BDD *q* corresponding to the HSs of *S*(*p*)

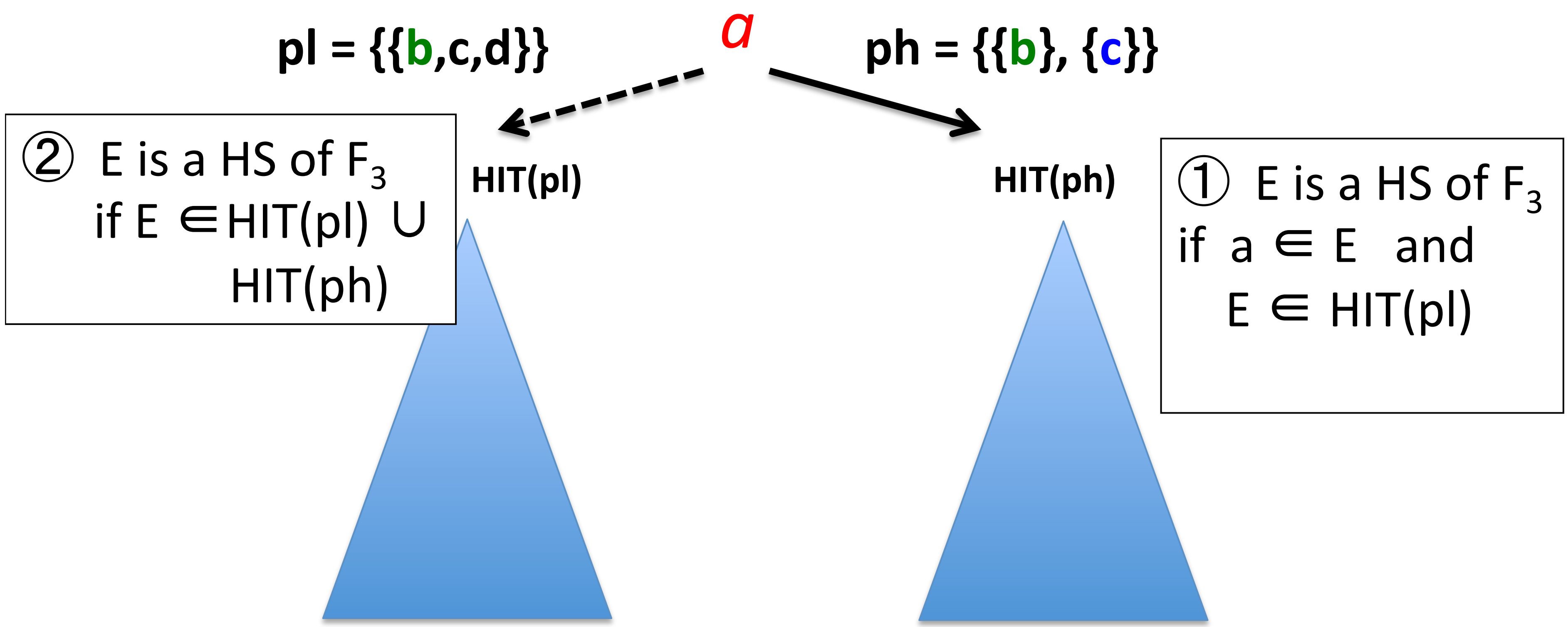


Example

Assume that every set is listed by the alphabetical order

- Let F_3 the family { {*a*, *b*}, {*a*, *c*}, {*b*, *c*, *d*} }.

2. Compute a BDD *q* corresponding to the HSs of *S*(*p*)



Available packages

- Uno's method
  - SHD3.1 (C code) by Murakami and Uno, 2013
  - A local one (Java code) by Nabeshima, 2009

  Note: the second one has been embedded into CF-induction (our Inductive learning system)

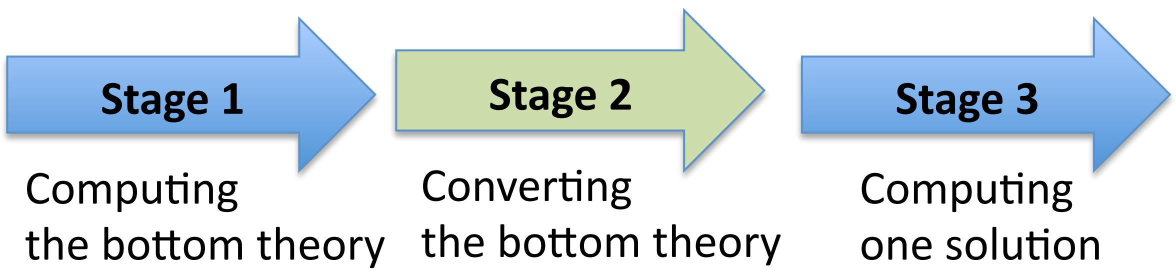
- Toda's method
  - HTC-BDD v1.0.0 (C code) by Toda, 2013

  Note: using BDD Package SAPPRO-Edition-1.0

Preliminary experiments for comparisons

- Empirical evaluation of the two methods wrt
  - the scalability and speed using…
- Randomly generated *one* CNF problem with 3 parameters:
  - Num. of variables: **v** ∈ { 5, 6, …, 30 }
  - Ratio of Num. of clauses to Num. of variables : **r** ∈ {1, 2, 3, 4, 5}
  - Average probability: **p** = 50 % (meaning that each variable or its negation appears in each clause)
- Environment: 8GB, Mac OS X, 2.66 GHz
- Time out: 60 sec

Current system with Uno's method



Comparison of two methods on the scalability

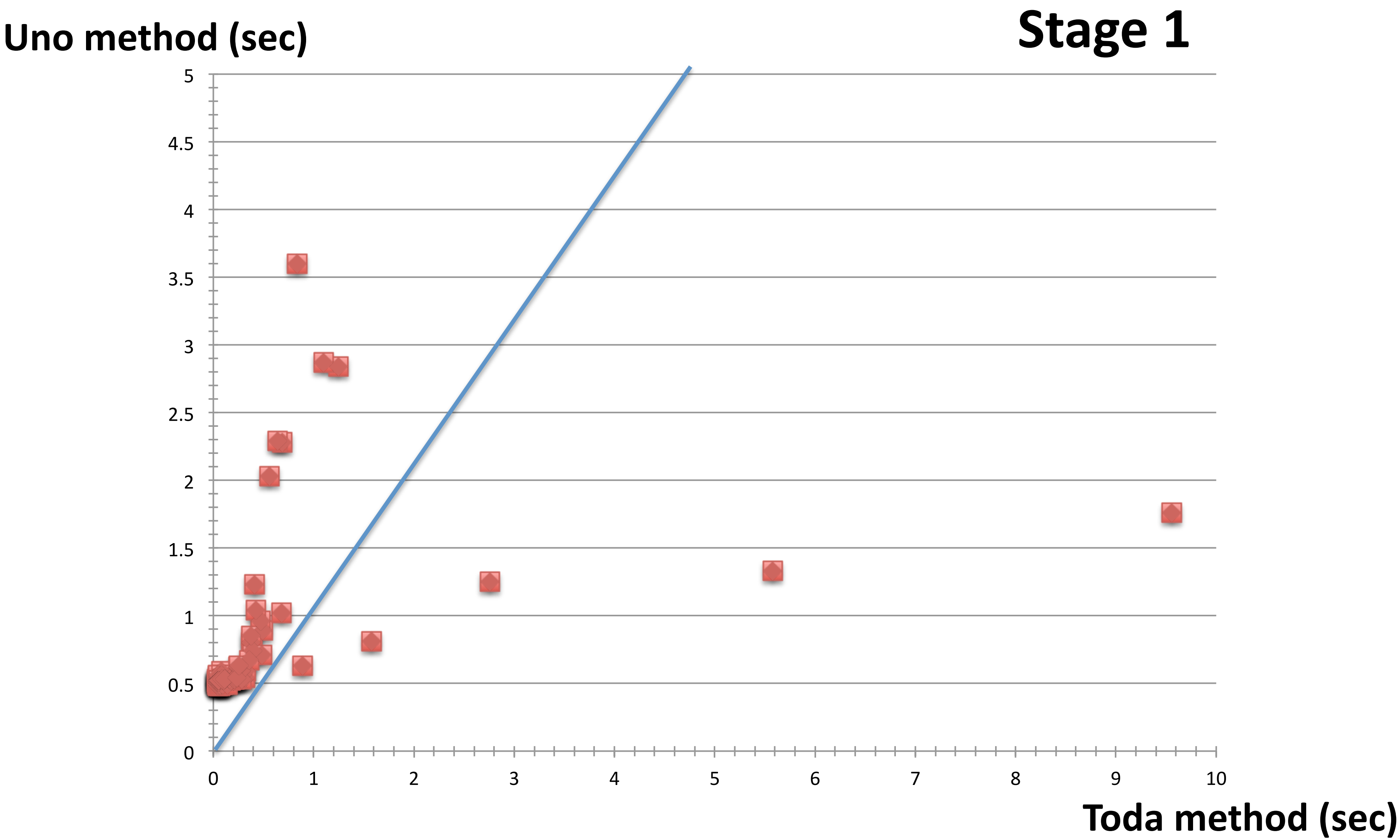
Uno's method with enum. tree

	Computation in Stage 1 (sec)				
V	R=1	R=2	R=3	R=4	R=5
5	0.042	0.043	0.044	0.046	0.052
6	0.04	0.041	0.045	0.043	0.047
7	0.044	0.051	0.048	0.048	0.051
8	0.051	0.059	0.057	0.055	0.054
9	0.065	0.082	0.082	0.081	0.088
10	0.07	0.098	0.115	0.104	0.116
11	0.094	0.16	0.211	0.213	0.233
12	0.138	0.239	0.282	0.249	0.253
13	0.188	0.217	0.276	0.359	0.376
14	0.177	0.286	0.392	0.425	0.409
15	0.316	0.38	0.473	0.56	0.64
16	0.322	0.491	0.687	0.838	1.125
17	0.485	0.684	1.1	1.564	T.O.
18	0.89	1.25	2.24	T.O.	T.O.
19	1.575	T.O.	T.O.	T.O.	T.O.
20	2.759	T.O.	T.O.	T.O.	T.O.
21	5.575	T.O.	T.O.	T.O.	T.O.
22	9.558	T.O.	T.O.	T.O.	T.O.
23	22.036	T.O.	T.O.	T.O.	T.O.
24	T.O.	T.O.	T.O.	T.O.	T.O.
25	T.O.	T.O.	T.O.	T.O.	T.O.
26	T.O.	T.O.	T.O.	T.O.	T.O.
27	T.O.	T.O.	T.O.	T.O.	T.O.
28	T.O.	T.O.	T.O.	T.O.	T.O.
29	T.O.	T.O.	T.O.	T.O.	T.O.
30	T.O.	T.O.	T.O.	T.O.	T.O.

Toda's method with BDD/ZDD

	Computation in Stage 1 (sec)				
V	R=1	R=2	R=3	R=4	R=5
5	0.51	0.53	0.55	0.52	0.52
6	0.49	0.52	0.56	0.51	0.52
7	0.48	0.53	0.52	0.51	0.52
8	0.51	0.52	0.52	0.51	0.52
9	0.48	0.59	0.56	0.52	0.53
10	0.49	0.52	0.53	0.52	0.53
11	0.49	0.53	0.53	0.54	0.54
12	0.49	0.54	0.53	0.62	0.63
13	0.55	0.53	0.59	0.67	0.85
14	0.52	0.60	0.74	1.04	1.23
15	0.54	0.80	0.96	2.03	2.29
16	0.59	0.89	2.28	3.60	7.39
17	0.71	1.02	2.87	6.79	3.44
18	0.63	2.84	8.82	5.04	8.17
19	0.81	3.78	7.96	10.56	T.O
20	1.25	10.68	8.63	T.O	T.O
21	1.33	5.87	15.25	T.O	T.O
22	1.76	9.05	T.O	T.O	T.O
23	3.34	12.64	T.O	T.O	T.O
24	4.99	T.O	T.O	T.O	T.O
25	7.01	T.O	T.O	T.O	T.O
26	16.07	T.O	T.O	T.O	T.O
27	7.80	T.O	T.O	T.O	T.O
28	19.45	T.O	T.O	T.O	T.O
29	17.30	T.O	T.O	T.O	T.O
30	7.37	T.O	T.O	T.O	T.O

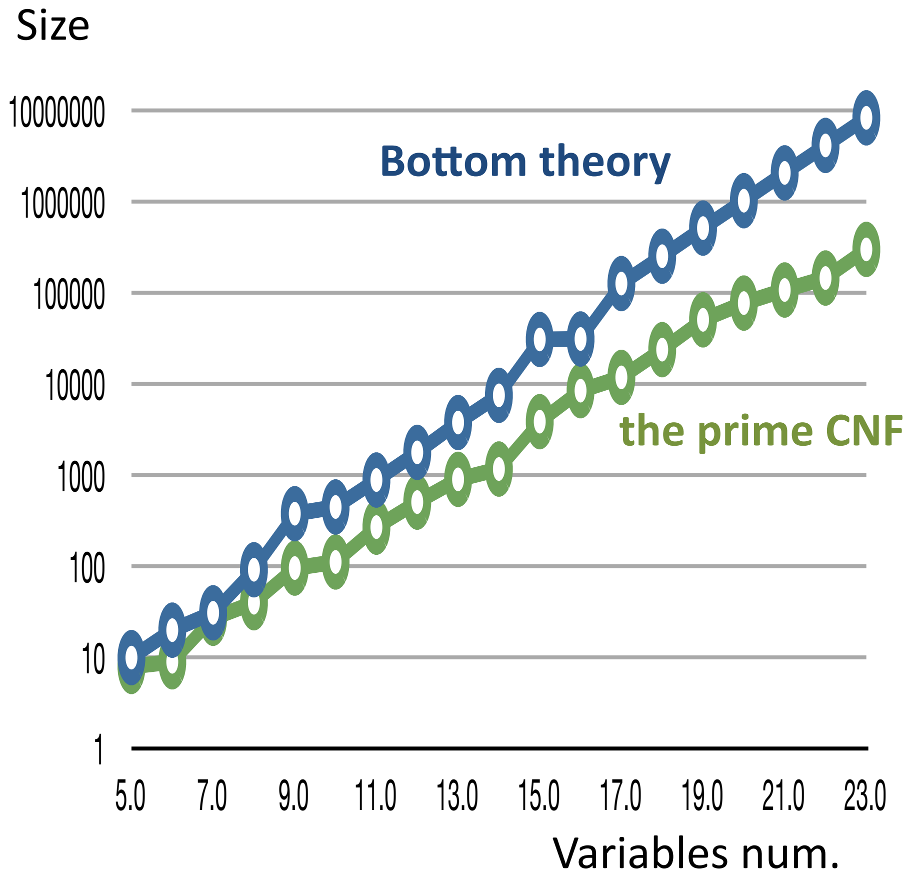
Comparison of two methods on the speed



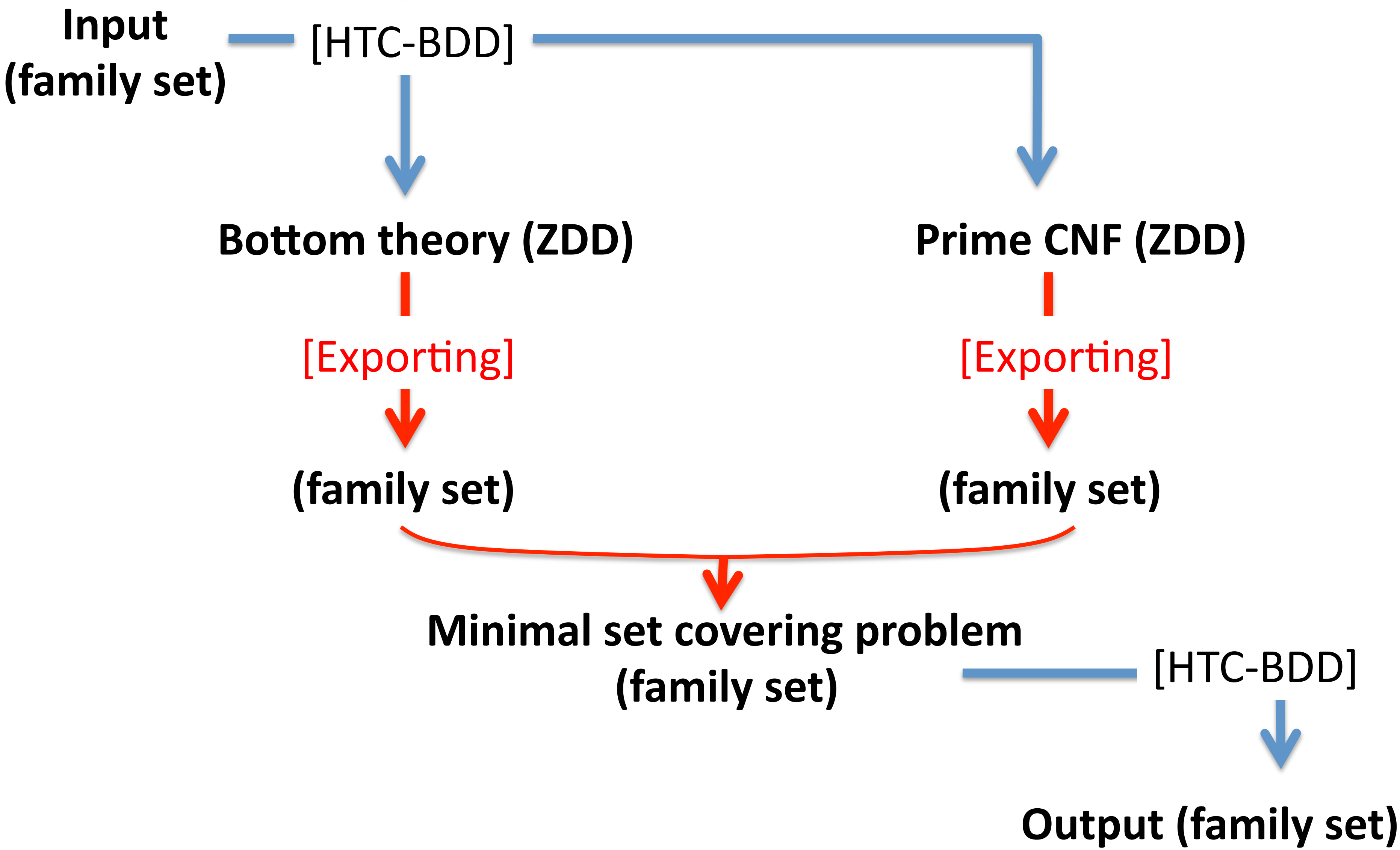
Bottleneck in our approach

- Converting the prime CNF and bottom theory into a new minimal set covering problem [**Stage 2**]

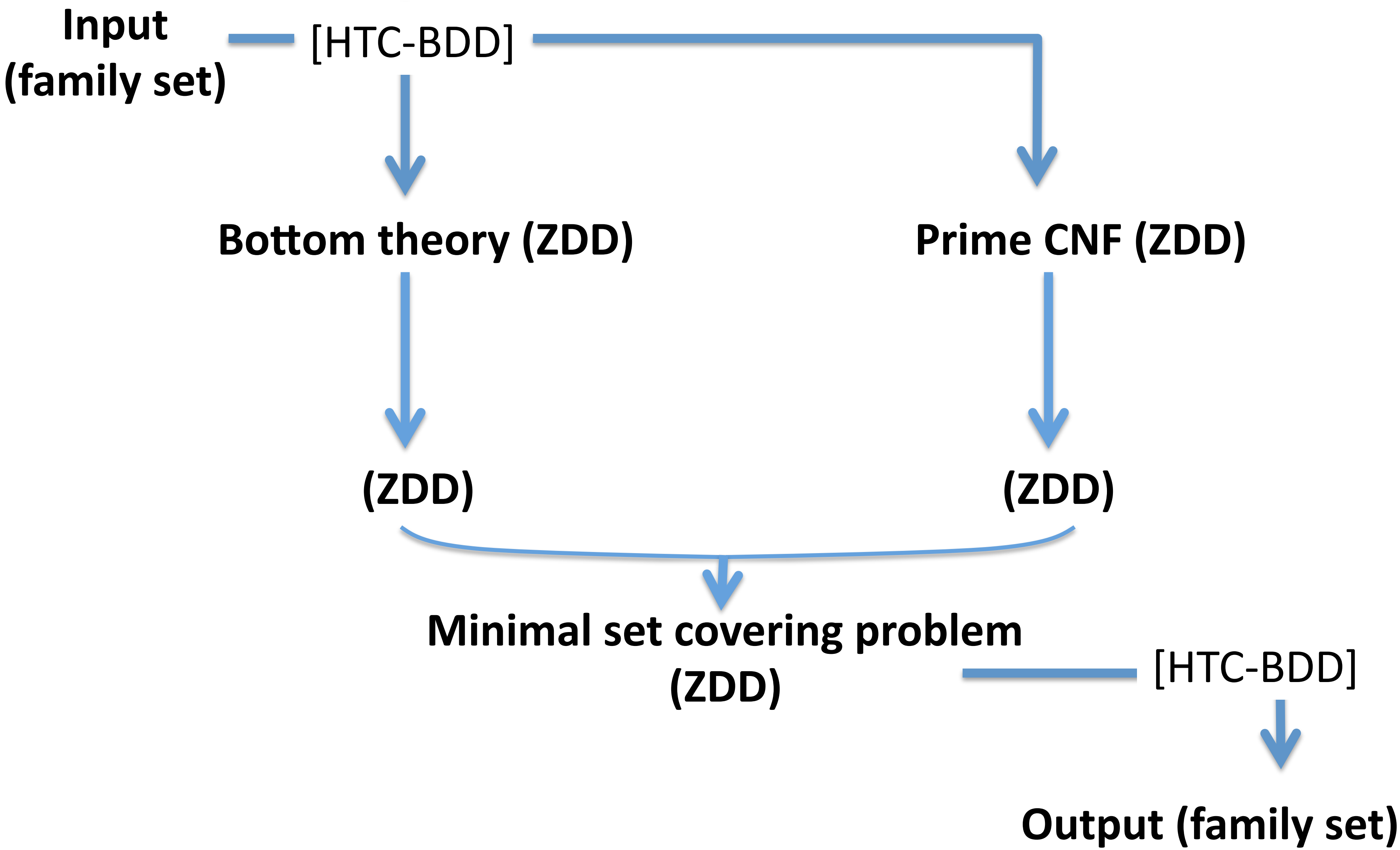
	Computation in Stage 2 (sec)				
V	R=1	R=2	R=3	R=4	R=5
5	0.00	0.00	0.00	0.00	0.00
6	0.00	0.00	0.00	0.00	0.00
7	0.00	0.00	0.00	0.00	0.00
8	0.01	0.01	0.00	0.00	0.00
9	0.02	0.02	0.02	0.01	0.01
10	0.03	0.04	0.04	0.02	0.02
11	0.08	0.11	0.10	0.09	0.08
12	0.10	0.15	0.16	0.13	0.11
13	0.17	0.45	0.50	0.67	T.O.
14	0.34	T.O.	T.O.	T.O.	T.O.
15	3.02	T.O.	T.O.	T.O.	T.O.
16	5.95	T.O.	T.O.	T.O.	T.O.
17	34.56	T.O.	T.O.	T.O.	T.O.
18	T.O.	T.O.	T.O.	T.O.	T.O.
19	T.O.	T.O.	T.O.	T.O.	T.O.
20	T.O.	T.O.	T.O.	T.O.	T.O.
21	T.O.	T.O.	T.O.	T.O.	T.O.
22	T.O.	T.O.	T.O.	T.O.	T.O.
23	T.O.	T.O.	T.O.	T.O.	T.O.
24	T.O.	T.O.	T.O.	T.O.	T.O.
25	T.O.	T.O.	T.O.	T.O.	T.O.
26	T.O.	T.O.	T.O.	T.O.	T.O.
27	T.O.	T.O.	T.O.	T.O.	T.O.
28	T.O.	T.O.	T.O.	T.O.	T.O.
29	T.O.	T.O.	T.O.	T.O.	T.O.
30	T.O.	T.O.	T.O.	T.O.	T.O.



Current system design with BDD/ZDD



Elaborative system design in BDD/ZDD



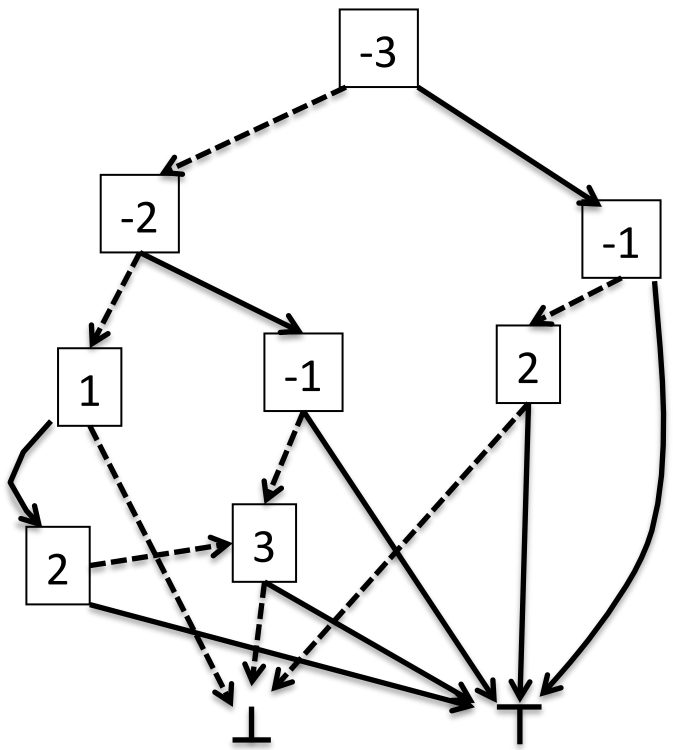
Ex) Recall the previous example (we delete `` x '' for simplicity)

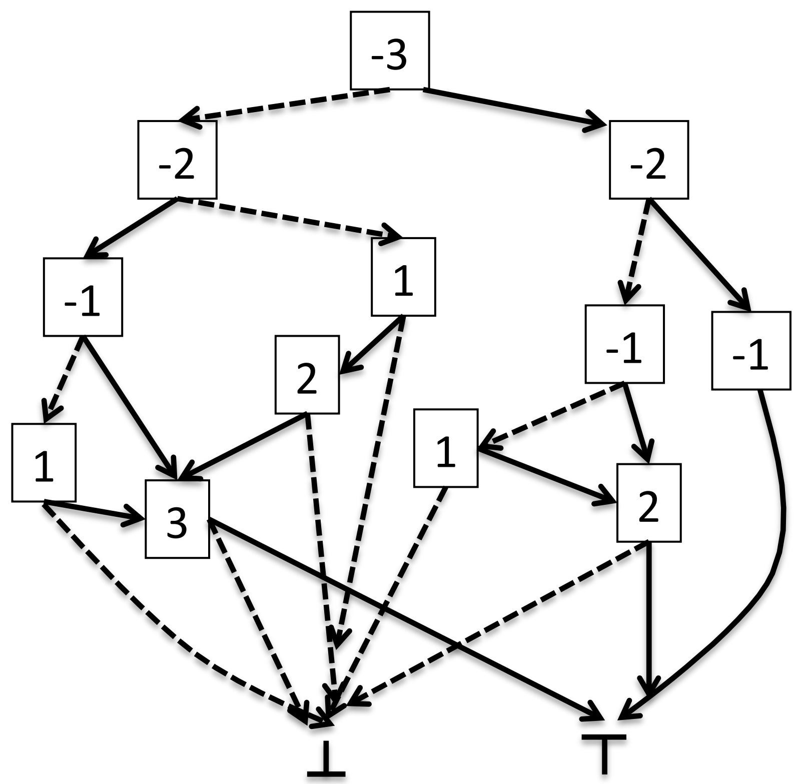
Prime CNF:

{ {-3, -1}, {-3, 2}, {-2, -1}, {-2, 3}, {1, 2}, {1, 3} }

Bottom theory:

{ {-3, -2, -1}, {-3, -1, 2}, {-3, 1, 2 }, {-2, -1, 3}, {-2, 1, 3}, {1, 2, 3} }



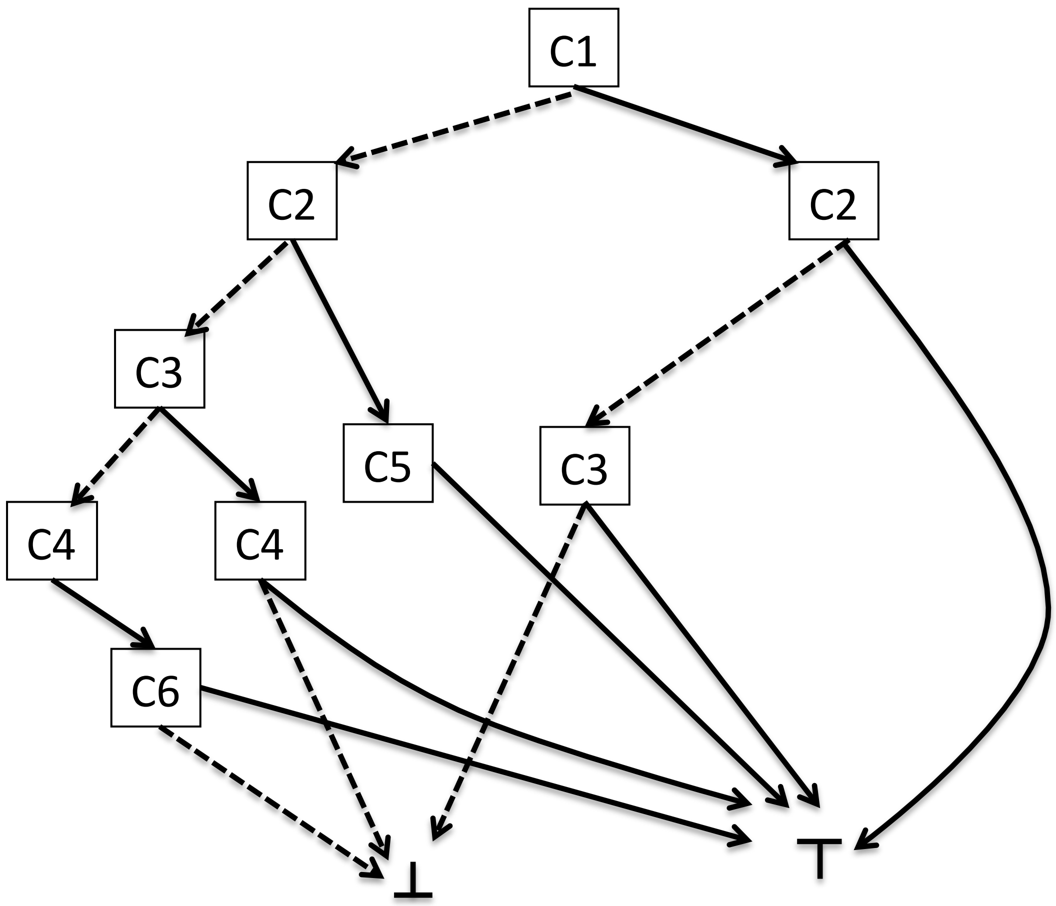


C1 C2 C3 C4 C5 C6

Prime: { {-3, -1}, {-3, 2}, {-2, -1}, {-2, 3}, {1, 2}, {1, 3} }

Bottom: { {-3, -2, -1}, {-3, -1, 2}, {-3, 1, 2 }, {-2, -1, 3}, {-2, 1, 3}, {1, 2, 3} }

Problem: { { C1, C3}, {C1, C2}, {C2, C5}, {C3, C4}, {C4, C6}, {C5, C6} }

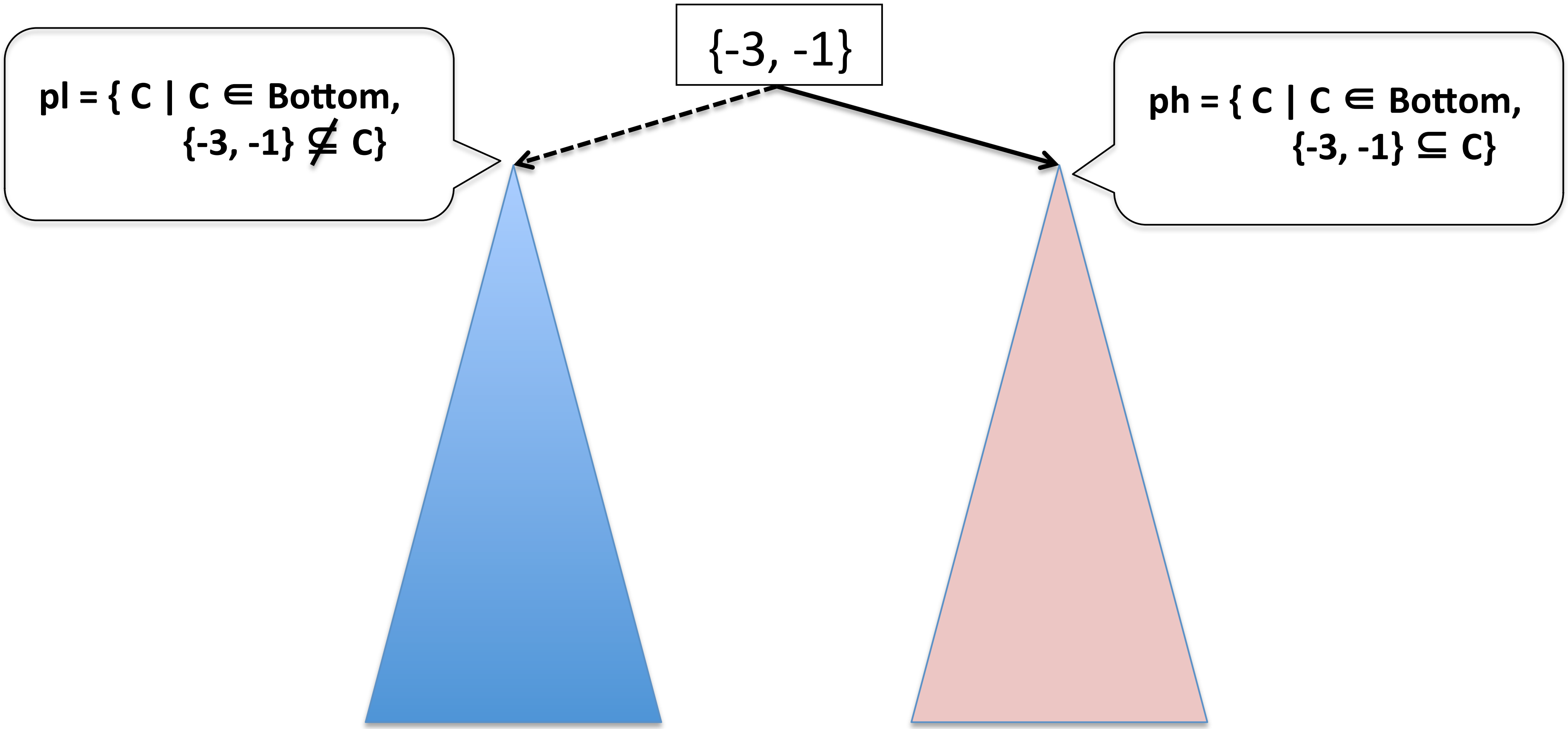


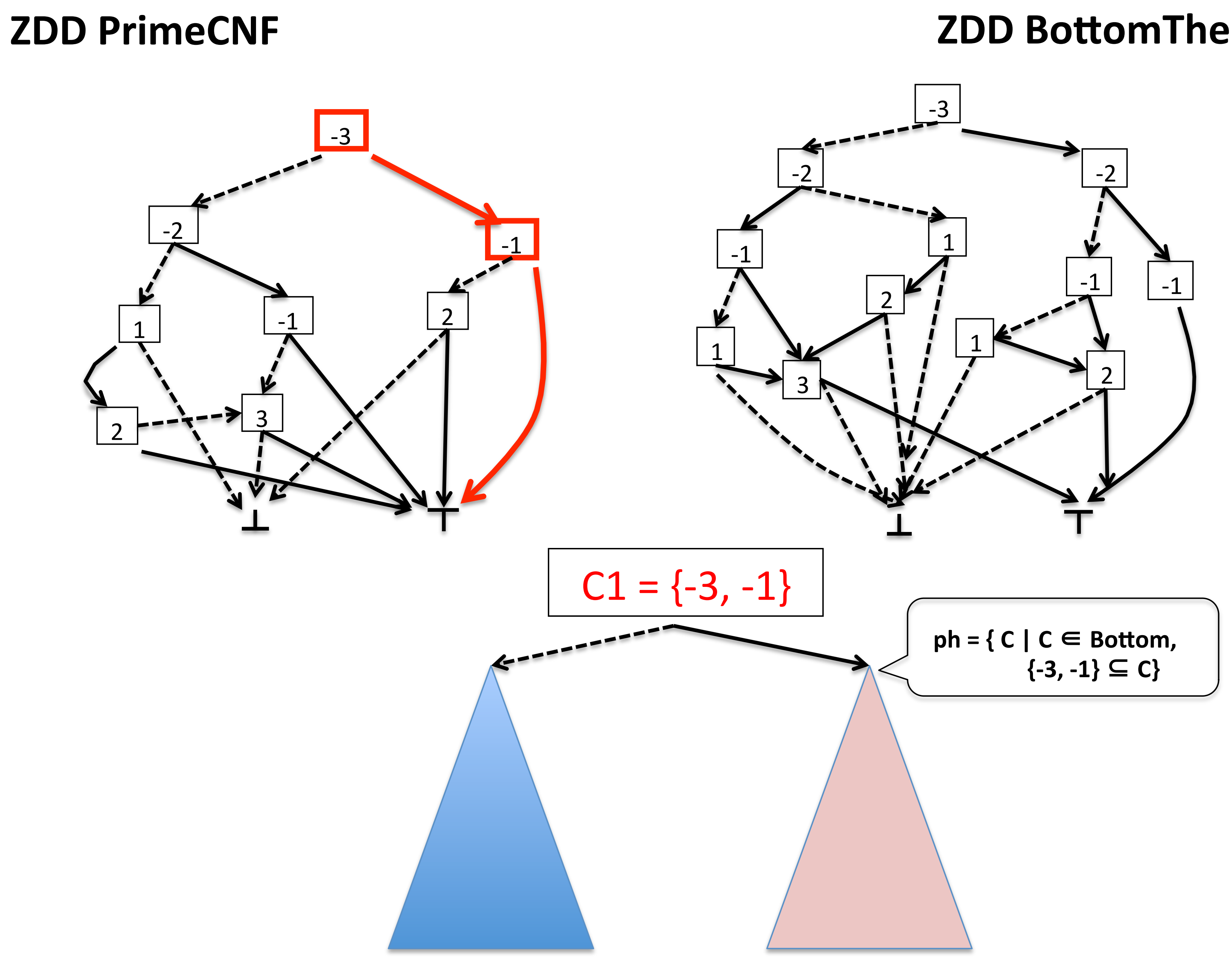
C1 C2 C3 C4 C5 C6

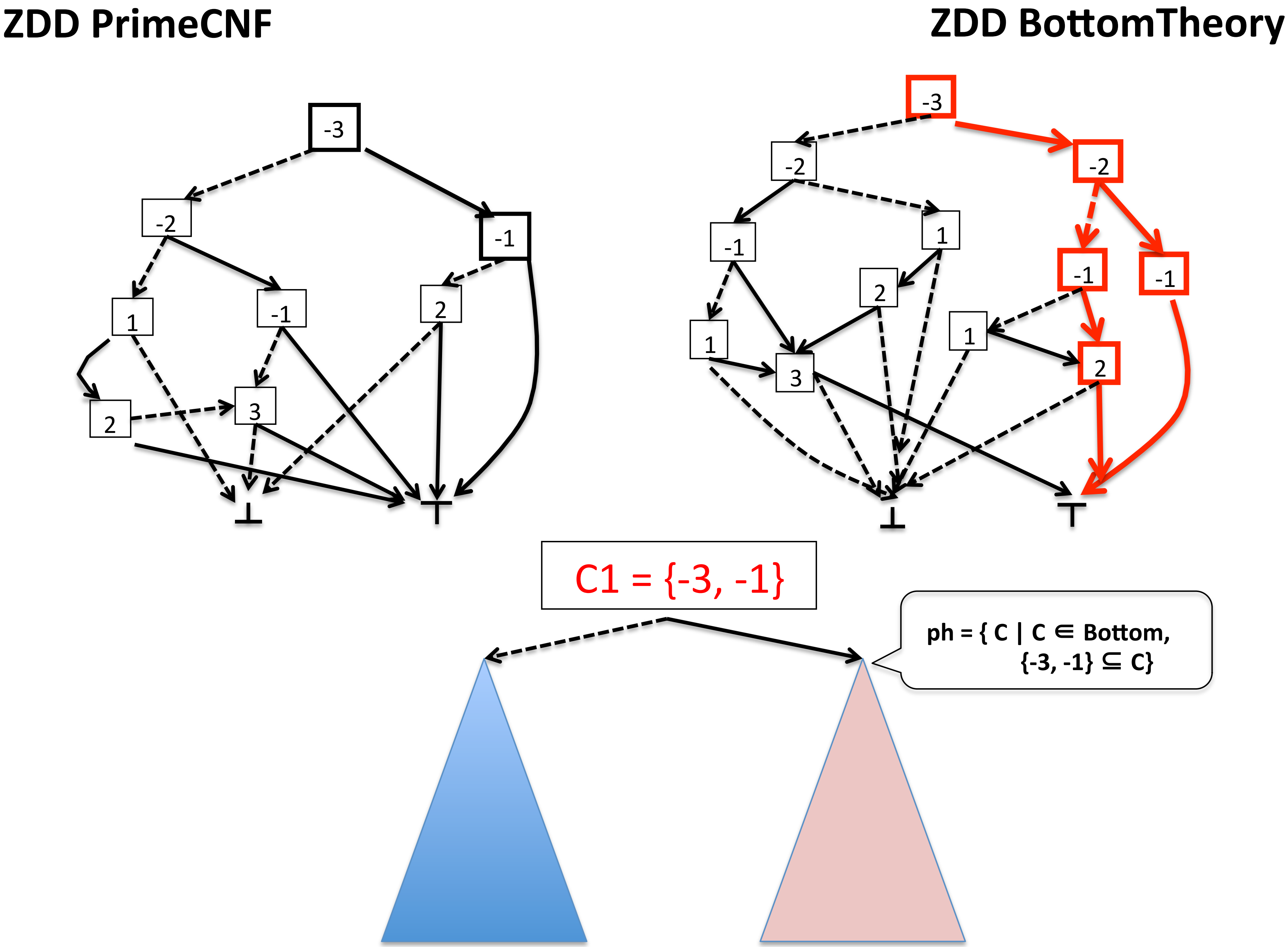
Prime: { {-3, -1}, {-3, 2}, {-2, -1}, {-2, 3}, {1, 2}, {1, 3} }

Bottom: { {-3, -2, -1}, {-3, -1, 2}, {-3, 1, 2 }, {-2, -1, 3}, {-2, 1, 3}, {1, 2, 3} }

Problem: { { C1, C3}, {C1, C2}, {C2, C5}, {C3, C4}, {C4, C6}, {C5, C6} }




ZDD PrimeCNF
ZDD BottomTheory
C1 = {-3, -1}
ph = { C | C ∈ Bottom,
{-3, -1} ⊆ C}



ZDD PrimeCNF
ZDD BottomTheory
C1 = {-3, -1}
ph = { C | C ∈ Bottom, {-3, -1} ⊆ C}

Conclusion and future work

- Non-monotone dualization (NMD)
  - Reducing NMD into two MD problems
- Monotone dualization (MD)
  - Uno's Enumeration tree based method
  - Toda's BDD/ZDD based method
- Preliminary comparisons of them in NMD
  - Scalability: Toda's method >> Uno's method
  - Speed: Toda's method <= Uno's method
- Considering a concrete algorithm for converting the bottom theory to a MD problem in ZDD
-