Tailoring Hypergraph Partitioning for Efficient d-DNNF Compilation

Jan Baudisch | 25.03.2025

SoftVarE

universität ulm

Technische Universität Braunschweig

Knowledge Compilation

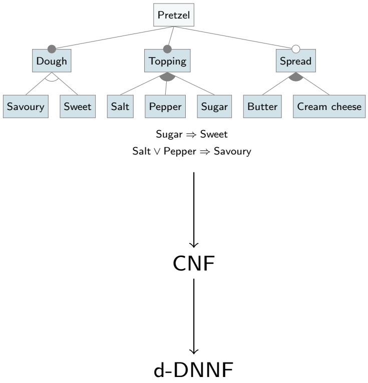
Feature Model Analysis

Expensive queries:

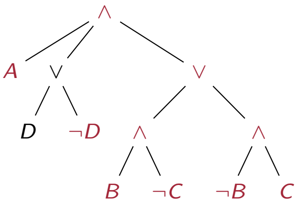
- Counting
- SAT
- Sampling
- Queries with assumptions
- ...

Knowledge Compilation

Transforming feature models into a format suitable for multiple (faster) analyses.



d-DNNF



d-DNNF

- Negation normal form: Negation only in literals
- Decomposable $\wedge$: Subtrees do not share variables
- Deterministic $\vee$: Subtrees are logically contradictory

Benefit

Analyses like SAT and counting in linear time.

d-DNNF Compilation

d4 d-DNNF Compiler

CNF $\rightarrow$ d-DNNF by using exhaustive DPLL

Variable Decision

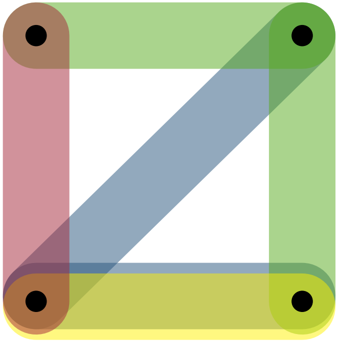
The branching variable has an impact on the overall runtime.

input : CNF formula F
output: d-DNNF of F

$S \leftarrow \mathit{solve}(F)$;
if $S = \emptyset$ **then** return $\bot$;
if $F = \emptyset$ **then** return $\mathit{andNode}(S, \top)$;
$\mathit{components} \leftarrow \mathit{split}(F)$;
$\mathit{sub} \leftarrow \{\}$;
for $C \in \mathit{components}$ **do**
  $V \leftarrow \mathit{choose}(C)$;
  $\mathit{node} \leftarrow \mathit{orNode}(V, d4(C|V), d4(C|\neg V))$;
  $\mathit{add}(\mathit{sub}, \mathit{node})$;
end
return $\mathit{andNode}(S, \mathit{sub})$;

Hypergraphs

$(B \lor C \lor D) \land (A \lor B \lor D) \land (A \lor C) \land (C \lor D)$



Hypergraph

Generalization of graphs: nets (edges) contain many vertices.

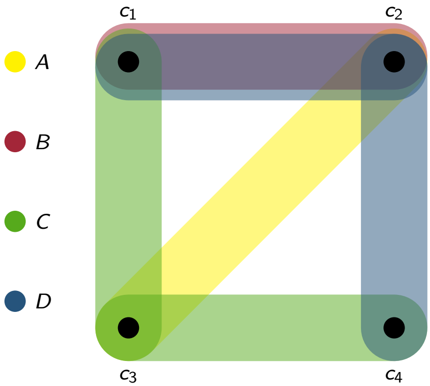
$H = (V, N)$

CNF Representation

Variables $\rightarrow$ Vertices
Clauses $\rightarrow$ Nets

Dual Hypergraphs

$(B \lor C \lor D) \land (A \lor B \lor D) \land (A \lor C) \land (C \lor D)$



Dual Hypergraph

Vertices and nets switch roles.

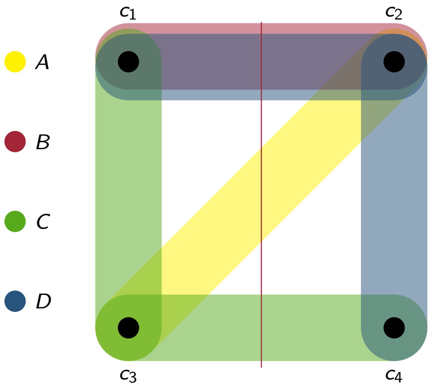
$H = (V, N)$
$H^* = (N, V)$

CNF Representation

Variables $\rightarrow$ Nets
Clauses $\rightarrow$ Vertices

Hypergraph Partitioning

$(B \lor C \lor D) \land (A \lor B \lor D) \land (A \lor C) \land (C \lor D)$



Partitioning

Division of the hypergraph into k blocks while optimizing a given metric.

Cutset

Nets containing vertices in multiple blocks.

Result

Cutset $\rightarrow$ Variables (to be decided)
Blocks $\rightarrow$ Clauses (components)

d-DNNF Compilation

Variable decision

Cutset assignment $\rightarrow$ disconnected components

input : CNF formula F

output: d-DNNF of F

$S \leftarrow solve(F)$;
if $S = \emptyset$ **then** return $\perp$;
if $F = \emptyset$ **then** return $andNode(S, \top)$;
$components \leftarrow split(F)$;
$sub \leftarrow \{\}$;
for $C \in components$ **do**
    $V \leftarrow choose(C)$;
    $node \leftarrow orNode(V, d4(C|V), d4(C|\neg V))$;
    $add(sub, node)$;
end
return $andNode(S, sub)$;

Research Questions

Existing partitioners

Performance for d-DNNF compilation?

CNF and hypergraph metrics

Correlation CNF/hypergraph metrics $\rightarrow$ difficulty of resulting formulas?

New partitioner

Specifically for DPLL style algorithms

Dataset

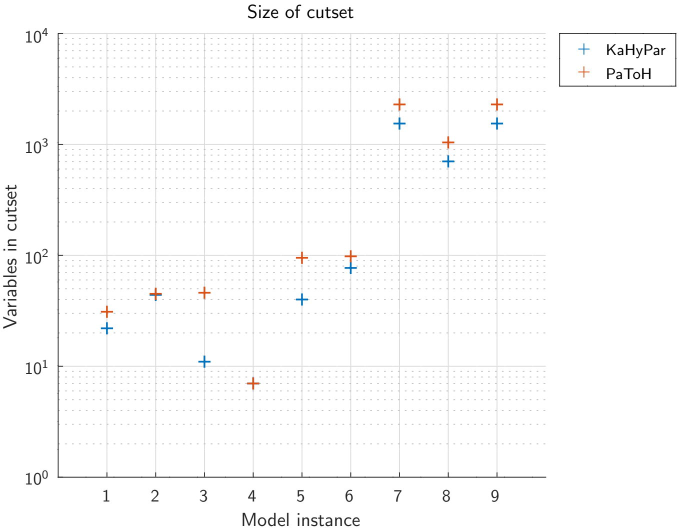
Source

- Feature Model Benchmark
- Industry models

Filter

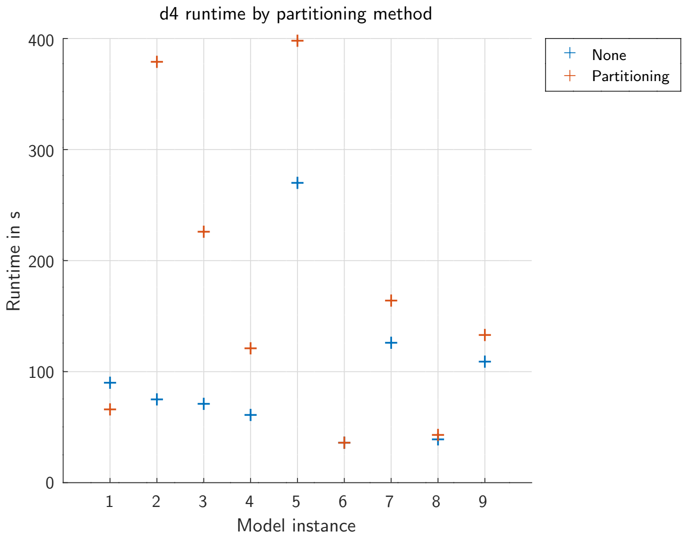
Solvable by d4 in $10s \dots 2h$

Cut Size



Size of cutset
Variables in cutset
Model instance
10^0
10^1
10^2
10^3
10^4
1
2
3
4
5
6
7
8
9
KaHyPar
PaToH

Performance



Improvements

Current state

Unweighted partitioning

CNF $\rightarrow$ unweighted hypergraph $\rightarrow$ partition

Fixed block size

Input CNF $\rightarrow$ 2 blocks

Possible improvements

Weighted partitioning

Net weights based on heuristics, e.g. $DLCS(v) =$ occurences of variable v in CNF

Community detection

Community detection algorithm $\rightarrow$ dynamic block size

Improvements

Current state

Static partitioning

Decomposition tree precalculated before DPLL

Cut based partitioning

Partitioners optimize cut size/weight

Possible improvements

Dynamic partitioning

New hypergraph for each sub-problem

Feature model based partitioning

Feature model subtrees $\rightarrow$ blocks based on cross-tree constraints

Weighted Partitioning

MAXO (DLCS)

$MAXO(V) =$ occurrences of V

MOMS

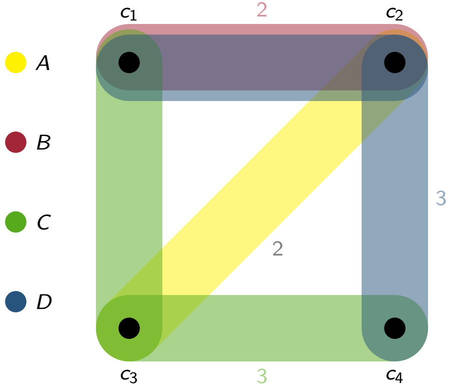
$MOMS(V) =$ occurrences of V in minimal clauses

MAMS

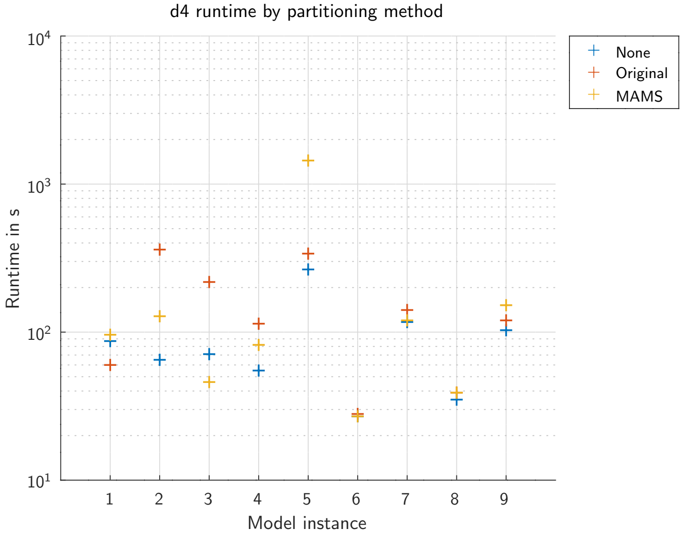
$MAMS(V) = MAXO(V) + MOMS(V)$

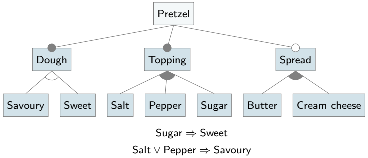
MAXO

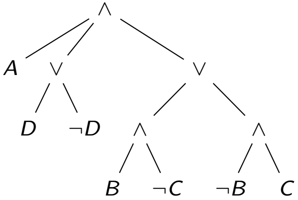
$$(B \vee C \vee D) \wedge (A \vee B \vee D) \wedge (A \vee C) \wedge (C \vee D)$$

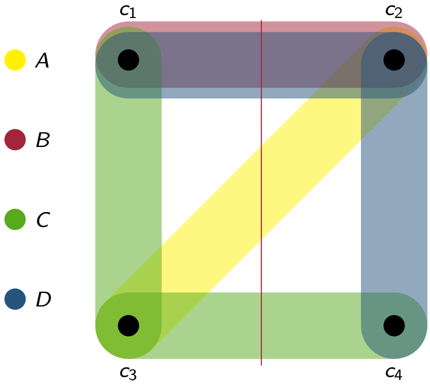


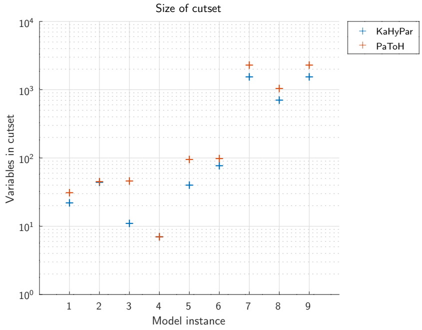
Weighted Partitioning




Pretzel
Dough
Topping
Spread
Savoury
Sweet
Salt
Pepper
Sugar
Butter
Cream cheese
Sugar ⇒ Sweet
Salt ∨ Pepper ⇒ Savoury


∧
A
∨
∨
D
¬D
∧
∧
B
¬C
¬B
C


A
B
C
D
c1
c2
c3
c4


Size of cutset
KaHyPar
PaToH
Variables in cutset
Model instance

Tools

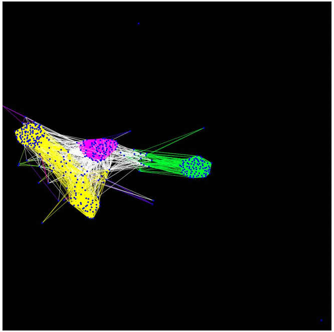
ddnnife d-DNNF reasoner

`github.com/SoftVarE-Group/d-dnnf-reasoner`

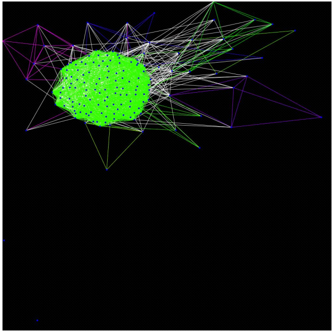
d4 Compiler

`github.com/SoftVarE-Group/d4v2`

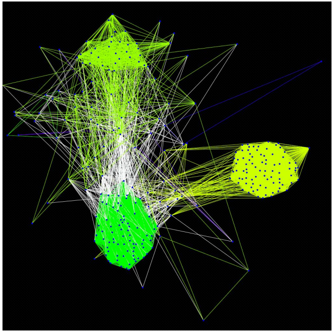
Community detection



Original CNF



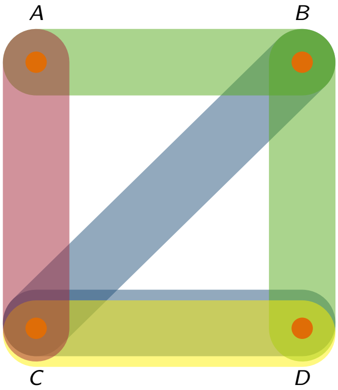
Block 1



Block 2

Hypergraphs

$(B \lor C \lor D) \land (A \lor B \lor D) \land (A \lor C) \land (C \lor D)$

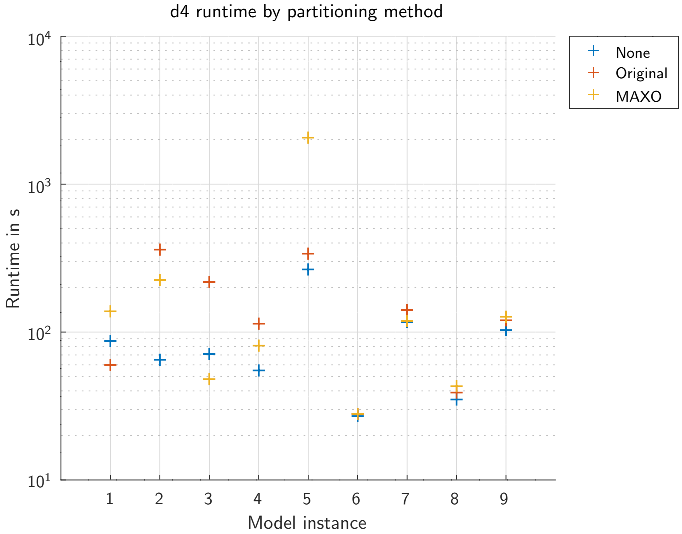


Pin

Connection between net and vertex.

$P = (n, v)$

Community detection



Community detection