Heuristic for Choosing SAT Encoding for Subsumption Resolution

The Workshop on Alignment of Proof Systems and Machine Learning

Robin Coutelier

robin.coutelier@tuwien.ac.at
TU Wien, Vienna, Austria

26 March 2024

Acknowledgements

We thank Pascal Fontaine (University of Liège, Belgium) for fruitful discussions. We acknowledge partial support from the ERC Consolidator Grant ARTIST 101002685, the FWF SFB project SpyCoDe F8504, the Austrian FWF project W1255-N23, the WWTF ICT22-007 Grant ForSmart, and the TU Wien Trustworthy Autonomous Cyber-Physical Systems Doctoral College. This research was funded in whole or in part by the Austrian Science Fund (FWF) [10.55776/F85, 10.55776/W1255]. For open access purposes, the author has applied a CC BY public copyright license to any author accepted manuscript version arising from this submission. Initial results on this work have been established during a research internship of Robin Coutelier at TU Wien, while he was still a master student at the University of Liège, Belgium.

erc European Research Council Established by the European Commission

FWF Der Wissenschaftsfonds.

W|W|T|F Vienna Science and Technology Fund

amazon web services

Who am I?

Curiculum Vitae

- 2017-2018: Study of Chinese language at Yunnan Normal University, Kunming, China,
- 2018-2021: Bachelor in Engineering, Computer Science and Mechanics at the University of Liège, Belgium,
- 2021-2023: Master in Computer Science and Engineering, focus on Machine Learning at the University of Liège, Belgium,
- 2023-now: PhD student at TU Wien, Vienna, Austria.

Who am I?

Curiculum Vitae

- 2017-2018: Study of Chinese language at Yunnan Normal University, Kunming, China,
- 2018-2021: Bachelor in Engineering, Computer Science and Mechanics at the University of Liège, Belgium,
- 2021-2023: Master in Computer Science and Engineering, focus on Machine Learning at the University of Liège, Belgium,
- 2023-now: PhD student at TU Wien, Vienna, Austria.

Research Interests

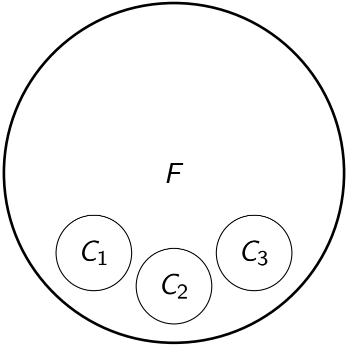
- Automated Theorem Proving (Vampire),
- SAT solving (Chronological Backtracking, NapSAT),
- Efficient data structures and algorithms in automated reasoning.

Introduction

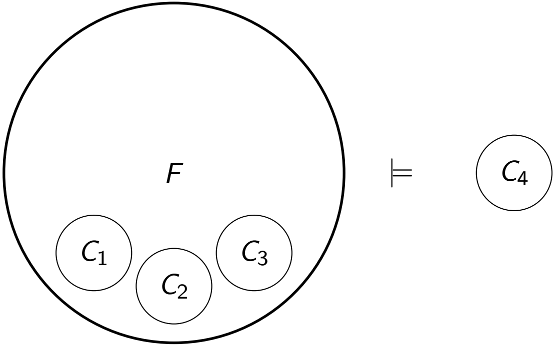
Related work

- First-Order Subsumption via SAT solving [Rath et al., 2022],
- SAT-based Subsumption Resolution [Coutelier et al., 2023],
- SAT Solving for Variants of First-Order Subsumption [Coutelier et al., 2024].

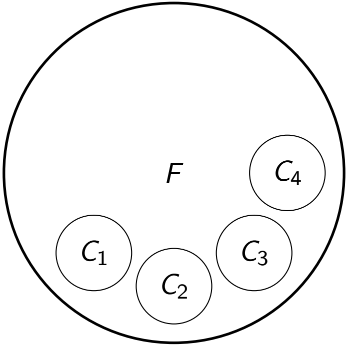
Saturation in FOL Theorem Proving



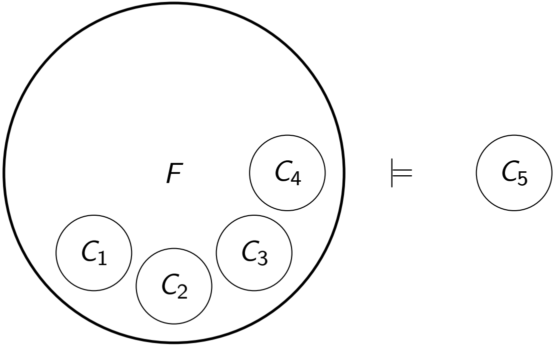
Saturation in FOL Theorem Proving



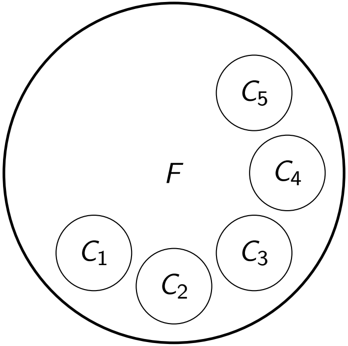
Saturation in FOL Theorem Proving



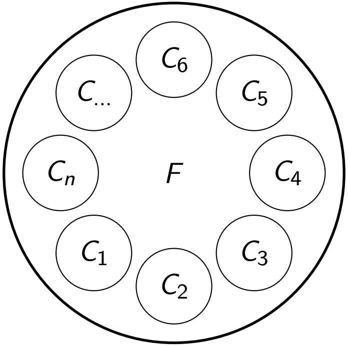
Saturation in FOL Theorem Proving



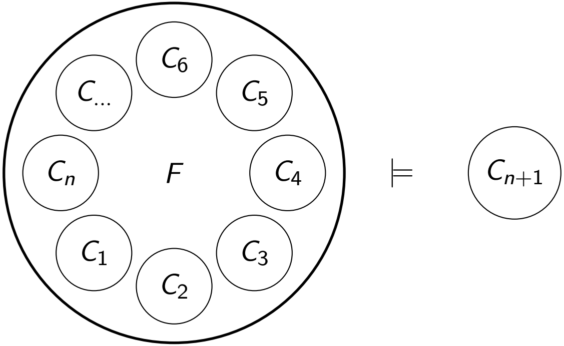
Saturation in FOL Theorem Proving



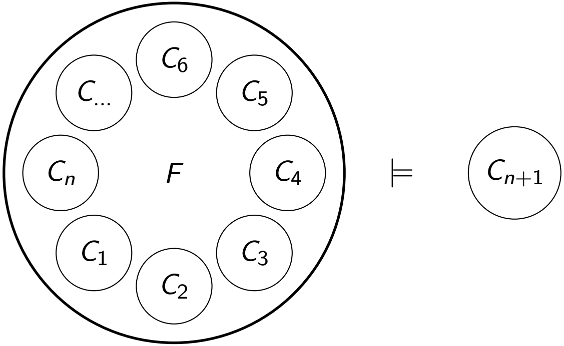
Saturation in FOL Theorem Proving



Saturation in FOL Theorem Proving



Saturation in FOL Theorem Proving



Out of memory!

Subsumption

Definition

A clause S *subsumes* a distinct clause M iff there is a substitution σ such that

$$\sigma(S) \sqsubseteq M$$

where $\sqsubseteq$ is the sub-multiset inclusion relation.

If S subsumes M, then M is redundant and can be removed from the formula.

Subsumption - Examples

Example (propositional logic)

$$S = a \vee b$$
$$M = a \vee b \vee c$$

S subsumes M. It is "stronger" than M.

Subsumption - Examples

Example (propositional logic)

$$S = a \vee b$$
$$M = a \vee b \vee c$$

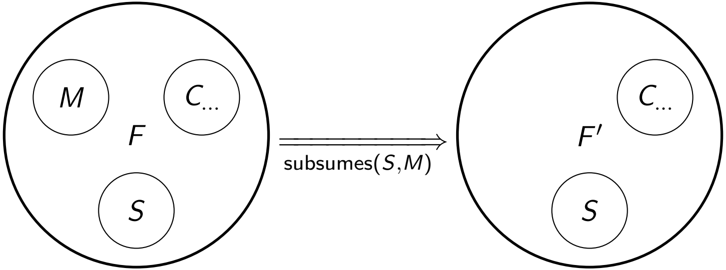
S subsumes M. It is "stronger" than M.

Example (FOL)

$$S = p(x_1, x_2) \vee p(f(x_2), x_3)$$
$$M = \neg p(f(c), d) \vee p(f(y), c) \vee p(f(c), g(d))$$

S subsumes M with the substitution $\sigma = \{x_1 \mapsto f(y), x_2 \mapsto c, x_3 \mapsto g(d)\}$.

Subsumption - Intuition



Subsumption Resolution

Resolution (Simplified)

$$\frac{S^* \vee s' \qquad \neg\sigma(s') \vee M^*}{\sigma(S^*) \vee M^*}$$

Subsumption Resolution

Resolution (Simplified)

$$\frac{S^* \vee s' \qquad \neg\sigma(s') \vee M^*}{\sigma(S^*) \vee M^*}$$

Definition

Clauses M and S are said to be the main and side premise of subsumption resolution, respectively, iff there is a substitution σ, a set of literals $S' \subseteq S$ and a literal $m' \in M$ such that

$$\sigma(S') = \{\neg m'\} \quad \text{and} \quad \sigma(S \setminus S') \subseteq M \setminus \{m'\}.$$

Subsumption Resolution aims to remove a literal from the main premise.

Subsumption Resolution - Example 1

Example (propositional logic)

$$\frac{S := \boxed{a} \lor b \qquad M := \boxed{\neg a} \lor b \lor c}{M^* := b \lor c}$$

$\neg a$ is the resolution literal. M^* subsumes M and can replace M in the clause set.

Subsumption Resolution - Example 1

Example (propositional logic)

$$\frac{S := \boxed{a} \vee b \qquad M := \cancel{\boxed{\neg a} \vee b \vee c}}{M^* := b \vee c}$$

$\neg a$ is the resolution literal. M^* subsumes M and can replace M in the clause set.

Subsumption Resolution - Example 2

Example (FOL)

$$S = p(x_1, x_2) \lor p(f(x_2), x_3)$$
$$M = \neg p(f(y), d) \lor p(g(y), c) \lor \neg p(f(c), e)$$
$$\sigma = \{x_1 \mapsto g(y), x_2 \mapsto c, x_3 \mapsto e\}$$

Subsumption Resolution - Example 2

Example (FOL)

$$
\begin{aligned}
S &= p(x_1, x_2) \lor p(f(x_2), x_3) \\
M &= \neg p(f(y), d) \lor p(g(y), c) \lor \neg p(f(c), e) \\
\sigma &= \{x_1 \mapsto g(y), x_2 \mapsto c, x_3 \mapsto e\}
\end{aligned}
$$

$$
\dfrac{\dfrac{p(x_1, x_2) \lor p(f(x_2), x_3)}{p(g(y), c) \lor \boxed{p(f(c), e)}} \qquad \neg p(f(y), d) \lor p(g(y), c) \lor \boxed{\neg p(f(c), e)}}{M^* := \neg p(f(y), d) \lor p(g(y), c)}
$$

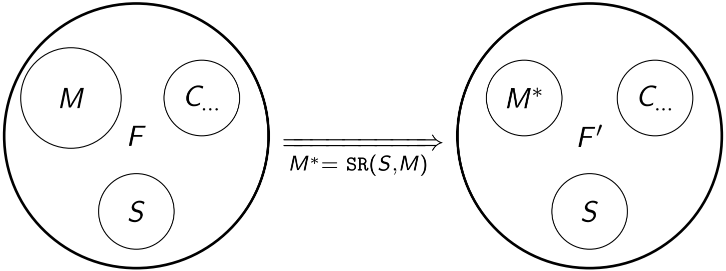
Subsumption Resolution - Example 2

Example (FOL)

$$S = p(x_1, x_2) \lor p(f(x_2), x_3)$$
$$M = \neg p(f(y), d) \lor p(g(y), c) \lor \neg p(f(c), e)$$
$$\sigma = \{x_1 \mapsto g(y), x_2 \mapsto c, x_3 \mapsto e\}$$

$$\frac{\dfrac{p(x_1, x_2) \lor p(f(x_2), x_3)}{p(g(y), c) \lor \boxed{p(f(c), e)}} \qquad \cancel{\neg p(f(y), d) \lor p(g(y), c) \lor \boxed{\neg p(f(c), e)}}}{M^* := \neg p(f(y), d) \lor p(g(y), c)}$$

Subsumption Resolution - Intuition



Importance of Redundancy Elimination

```
$ vampire Problems/GRP/GRP140-1.p -fsr off -t 30
...
132544.  $ false
% Termination reason:  Refutation
% Memory used [KB]: 308054
% Time elapsed:  6.654 s
--------------------------------------------------
```

Importance of Redundancy Elimination

```
$ vampire Problems/GRP/GRP140-1.p -fsr off -t 30
...
132544.  $ false
% Termination reason:  Refutation
% Memory used [KB]: 308054
% Time elapsed:  6.654 s
------------------------------------------------------------
$ vampire Problems/GRP/GRP140-1.p -fsr on -t 30
...
4918.  $ false
% Termination reason:  Refutation
% Memory used [KB]: 12025
% Time elapsed:  0.150 s
```

Relevance of Speed

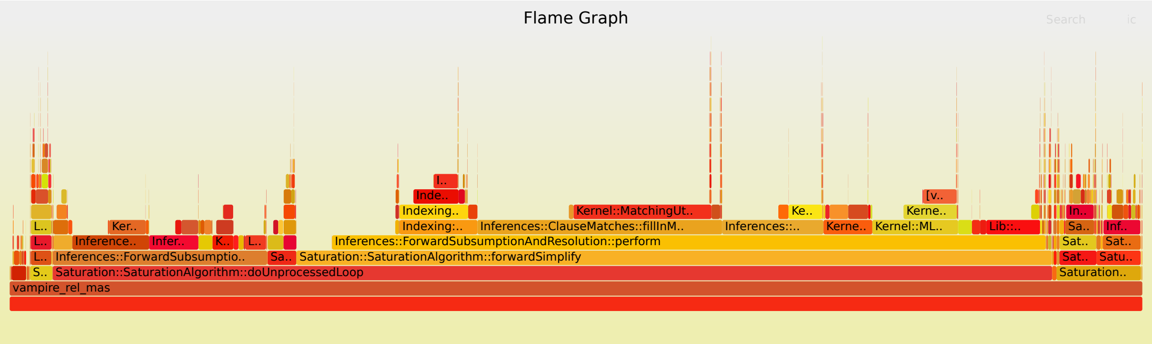
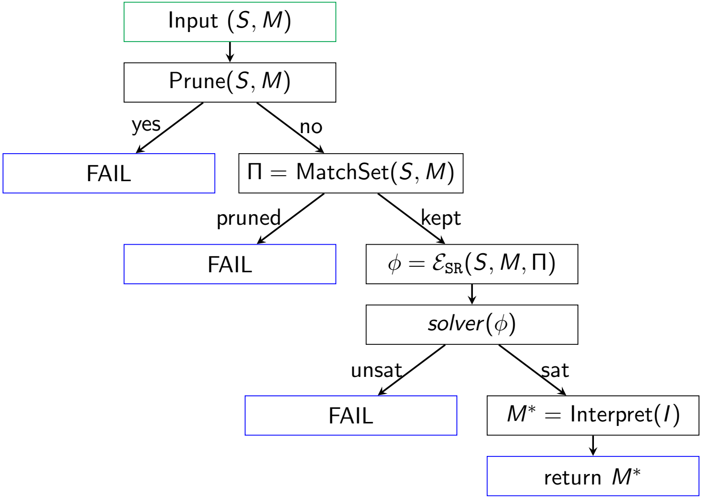


Figure: Typical profiling results for a TPTP problem (GRP001+6).

SAT-Based Subsumption Resolution



Two Encodings

Direct Encoding $\mathcal{E}^{d}_{\text{SR}}(S, M)$

positive compatibility	$\bigwedge_i \bigwedge_j \left(b^+_{i,j} \Rightarrow \Sigma^+_{i,j} \subseteq \sigma \right)$
negative compatibility	$\bigwedge_i \bigwedge_j \left(b^-_{i,j} \Rightarrow \Sigma^-_{i,j} \subseteq \sigma \right)$
existence	$\bigvee_i \bigvee_j b^-_{i,j}$
uniqueness	$\bigwedge_j \bigwedge_i \bigwedge_{i' \geq i} \bigwedge_{j' > j} \neg b^-_{i,j} \vee \neg b^-_{i',j'}$
completeness	$\bigwedge_i \bigvee_j b^+_{i,j} \vee b^-_{i,j}$
coherence	$\bigwedge_j \bigwedge_i \bigwedge_{i'} \neg b^+_{i,j} \vee \neg b^-_{i',j}$

Indirect Encoding $\mathcal{E}^{i}_{\text{SR}}(S, M)$

positive compatibility	$\bigwedge_i \bigwedge_j \left(b^+_{i,j} \Rightarrow \Sigma^+_{i,j} \subseteq \sigma \right)$
negative compatibility	$\bigwedge_i \bigwedge_j \left(b^-_{i,j} \Rightarrow \Sigma^-_{i,j} \subseteq \sigma \right)$
structurality	$\bigwedge_j \left[\neg c_j \vee \bigvee_i b^-_{i,j} \right] \wedge \bigwedge_j \bigwedge_i \left(c_j \vee \neg b^-_{i,j} \right)$
revised existence	$\bigvee_j c_j$
revised uniqueness	$AMO(\{c_j, j = 1, ..., \lvert M \rvert\})$
completeness	$\bigwedge_i \bigvee_j b^+_{i,j} \vee b^-_{i,j}$
revised coherence	$\bigwedge_j \bigwedge_i \left(\neg c_j \vee \neg b^+_{i,j} \right)$

Two Encodings

Direct Encoding $\mathcal{E}^{d}_{\mathrm{SR}}(S, M)$

positive compatibility	$\bigwedge_i \bigwedge_j \left(b^+_{i,j} \Rightarrow \Sigma^+_{i,j} \subseteq \sigma\right)$
negative compatibility	$\bigwedge_i \bigwedge_j \left(b^-_{i,j} \Rightarrow \Sigma^-_{i,j} \subseteq \sigma\right)$
existence	$\bigvee_i \bigvee_j b^-_{i,j}$
uniqueness	$\bigwedge_j \bigwedge_i \bigwedge_{i' \geq i} \bigwedge_{j' > j} \neg b^-_{i,j} \vee \neg b^-_{i',j'}$
completeness	$\bigwedge_i \bigvee_j b^+_{i,j} \vee b^-_{i,j}$
coherence	$\bigwedge_j \bigwedge_i \bigwedge_{i'} \neg b^+_{i,j} \vee \neg b^-_{i',j}$

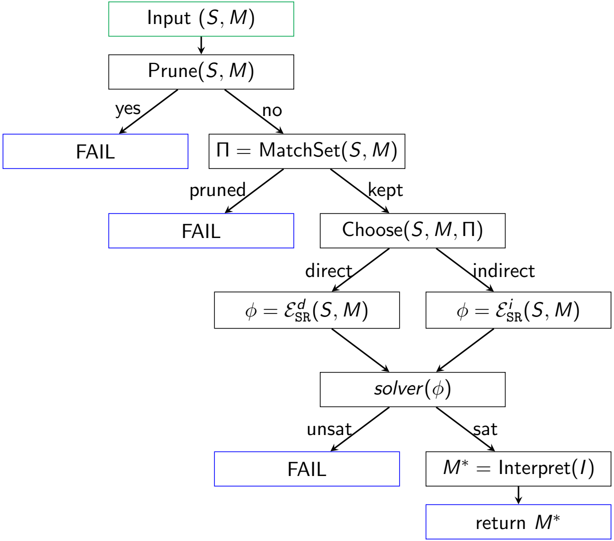
Indirect Encoding $\mathcal{E}^{i}_{\mathrm{SR}}(S, M)$

positive compatibility	$\bigwedge_i \bigwedge_j \left(b^+_{i,j} \Rightarrow \Sigma^+_{i,j} \subseteq \sigma\right)$
negative compatibility	$\bigwedge_i \bigwedge_j \left(b^-_{i,j} \Rightarrow \Sigma^-_{i,j} \subseteq \sigma\right)$
structurality	$\bigwedge_j \left[\neg c_j \vee \bigvee_i b^-_{i,j}\right] \wedge \bigwedge_j \bigwedge_i \left(c_j \vee \neg b^-_{i,j}\right)$
revised existence	$\bigvee_j c_j$
revised uniqueness	$AMO(\{c_j, j = 1, \ldots, \lvert M \rvert\})$
completeness	$\bigwedge_i \bigvee_j b^+_{i,j} \vee b^-_{i,j}$
revised coherence	$\bigwedge_j \bigwedge_i \left(\neg c_j \vee \neg b^+_{i,j}\right)$

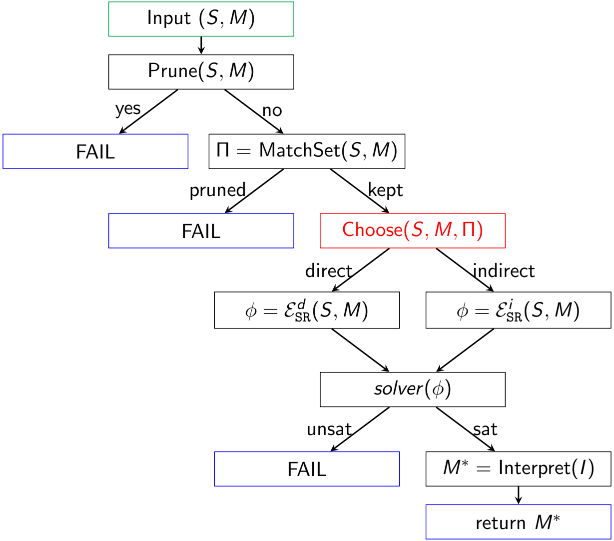
Complexities

- $\mathcal{E}^{d}_{\mathrm{SR}}(S, M)$ has $O(|\Pi|)$ variables and $O(|\Pi|^2)$ clauses.
- $\mathcal{E}^{i}_{\mathrm{SR}}(S, M)$ has $O(|\Pi| + |M|)$ variables and $O(|\Pi|)$ clauses.

Choosing the Encoding



Choosing the Encoding



Choosing Features

The features should be

- fast to compute;
- informative;
- independent.

Choosing Features

The features should be

- fast to compute;
- informative;
- independent.

1. Number of literals of S;
2. Number of literals of M;
3. Sparsity of the match set $\frac{|\Pi|}{|S| \cdot |M|}$.

Choosing the Architecture

What do we want?

We want a model that is

- fast to compute;
- generalisable;
- interpretable;
- easy to train.

Choosing the Architecture

What do we want?

We want a model that is

- fast to compute;
- generalisable;
- interpretable;
- easy to train.

Decision Trees are (almost) perfect

- can be hard coded in a few lines;
- not prone to overfitting;
- can be visualised;
- ... but cannot easily be trained online ...

Big Dataset without Online Learning

Objective function

$$\arg\min_{f \in \mathcal{F}} \mathbb{E}_{\substack{x \sim \mathcal{X} \\ (y_0, y_1) \sim \mathcal{D}(\cdot|x)}} \left[y_{f(x)} \right]$$

Big Dataset without Online Learning

Objective function

$$\arg\min_{f \in \mathcal{F}} \mathbb{E}_{\substack{x \sim \mathcal{X} \\ (y_0, y_1) \sim \mathcal{D}(\cdot | x)}} \left[y_{f(x)} \right]$$

What if the dataset cannot be loaded in memory?

Big Dataset without Online Learning

Objective function

$$\arg\min_{f \in \mathcal{F}} \mathbb{E}_{\substack{x \sim \mathcal{X} \\ (y_0, y_1) \sim \mathcal{D}(\cdot|x)}} \left[y_{f(x)} \right]$$

What if the dataset cannot be loaded in memory?

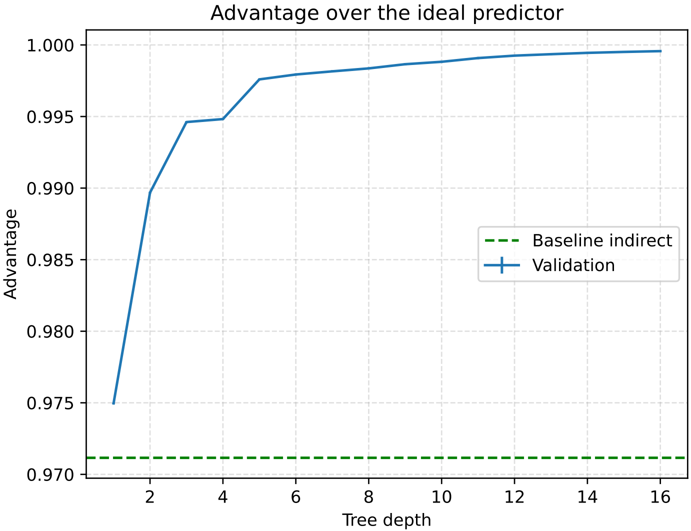
Revised objective function

We condense de dataset $\{(x, y_0, y_1)\}$ into $\mathcal{S} = \{(x, \hat{y}_0, \hat{y}_1)\}$ where $\hat{y}_0$ is the sum of the y_0 with the same x and $\hat{y}_1$ is the sum of the y_1 with the same x. Then we have

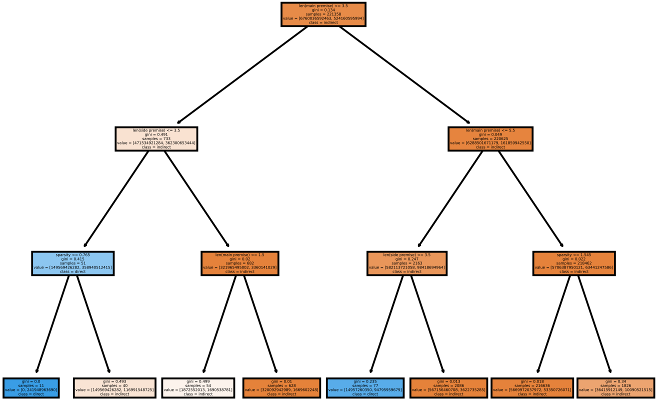
$$\arg\min_{f \in \mathcal{F}} \sum_{(x, \hat{y}_0, \hat{y}_1) \in \mathcal{S}} \left[|\hat{y}_0 - \hat{y}_1| * (f(x) - H(\hat{y}_0 - \hat{y}_1))^2 \right]$$

with H the step function

Setting Model Complexity



Final Tree



len(main premise) <= 3.5
gini = 0.134
samples = 221358
value = [6760036592463, 524160595994]
class = indirect
len(side premise) <= 3.5
gini = 0.491
samples = 733
value = [471534921284, 362300653444]
class = indirect
len(main premise) <= 5.5
gini = 0.049
samples = 220625
value = [6288501671179, 161859942550]
class = indirect
sparsity <= 0.765
gini = 0.415
samples = 51
value = [149569426282, 358940512415]
class = direct
len(main premise) <= 1.5
gini = 0.02
samples = 682
value = [321965495002, 3360141029]
class = indirect
len(side premise) <= 3.5
gini = 0.247
samples = 2163
value = [582113721058, 98418694964]
class = indirect
sparsity <= 1.545
gini = 0.022
samples = 218462
value = [5706387950121, 63441247586]
class = indirect
gini = 0.0
samples = 11
value = [0, 241948963690]
class = direct
gini = 0.493
samples = 40
value = [149569426282, 116991548725]
class = indirect
gini = 0.499
samples = 54
value = [1872552013, 1690538781]
class = indirect
gini = 0.01
samples = 628
value = [320092942989, 1669602248]
class = indirect
gini = 0.235
samples = 77
value = [14957260350, 94795959679]
class = direct
gini = 0.013
samples = 2086
value = [567156460708, 3622735285]
class = indirect
gini = 0.018
samples = 216636
value = [5669972037972, 53350726071]
class = indirect
gini = 0.34
samples = 1826
value = [36415912149, 10090521515]
class = indirect

Results (1/2)

Prover	Average	Std. Dev.	Boost
VAMPIRE_M	33.63 μs	1839.25 μs	1.00
VAMPIRE^*_D	25.38 μs	1241.86 μs	1.32
VAMPIRE^*_I	24.93 μs	196.38 μs	1.35
VAMPIRE^*_H	24.73 μs	190.69 μs	1.36

Table: Average time spent in the forward simplify loop.

Results (2/2)

Prover	Total Solved	Gain/Loss
Vampire_M	10 728	baseline
Vampire^*_D	10 791	(+94, −31)
Vampire^*_I	10 785	(+92, −35)
Vampire^*_H	10 794	(+97, −31)

Table: Number of TPTP problems solved by the considered versions of Vampire. The run was made using the options `-sa otter -av off` with a timeout of 60 s. The **Gain/Loss** column reports the difference of solved instances compared to Vampire_M.

Overfitting?

Not likely!

- The model is simple;
- The dataset is large;
- The feature space is small.

Overfitting?

Not likely!

- The model is simple;
- The dataset is large;
- The feature space is small.

In any case...

We want to solve problems from TPTP, generalisation is not our main goal.

References

Coutelier, R., Kovács, L., Rawson, M., and Rath, J. (2023).
Sat-based subsumption resolution.
In Pientka, B. and Tinelli, C., editors, *Automated Deduction - CADE 29 - 29th International Conference on Automated Deduction, Rome, Italy, July 1-4, 2023, Proceedings*, volume 14132 of *Lecture Notes in Computer Science*, pages 190–206. Springer.

Coutelier, R., Rath, J., Rawson, M., and Kovács, A. B. L. (2024).
Sat solving for variants of first-order subsumption.

Rath, J., Biere, A., and Kovács, L. (2022).
First-order subsumption via SAT solving.
In Griggio, A. and Rungta, N., editors, *22nd Formal Methods in Computer-Aided Design, FMCAD 2022, Trento, Italy, October 17-21, 2022*, pages 160–169. IEEE.