SAT-Based Subsumption Resolution

CADE29

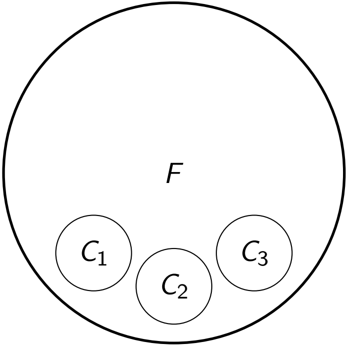
Robin Coutelier[1] Laura Kovács[2] Michael Rawson[2] Jakob Rath[2]

U. Liège, Liège, Belgium
robin.coutelier@student.uliege.be

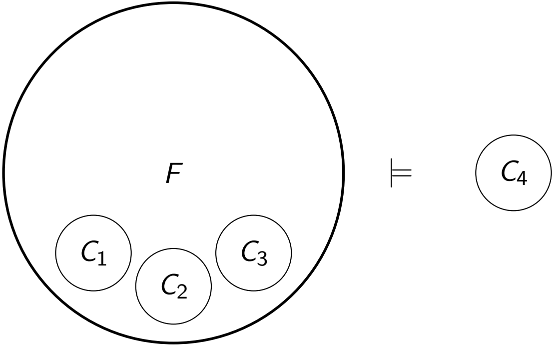
TU Wien, Vienna, Austria

2 July 2023

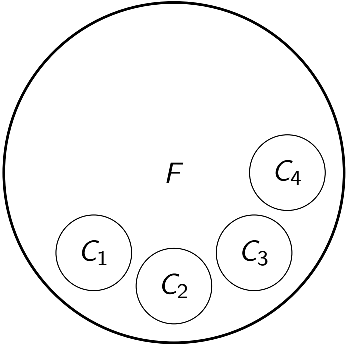
Saturation in FOL Theorem Proving



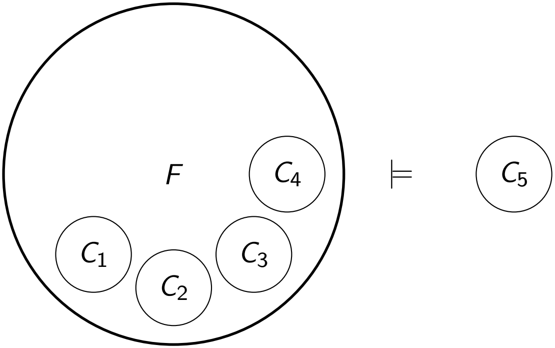
Saturation in FOL Theorem Proving



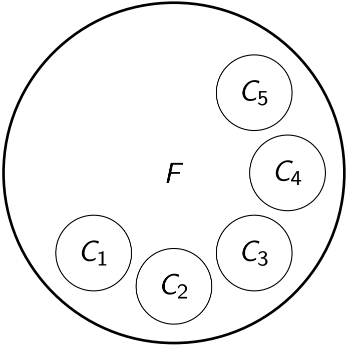
Saturation in FOL Theorem Proving



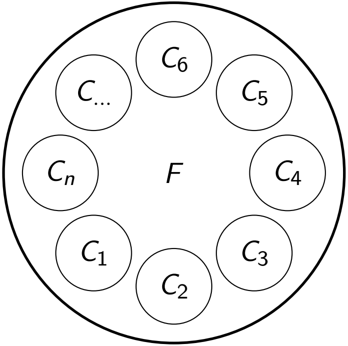
Saturation in FOL Theorem Proving



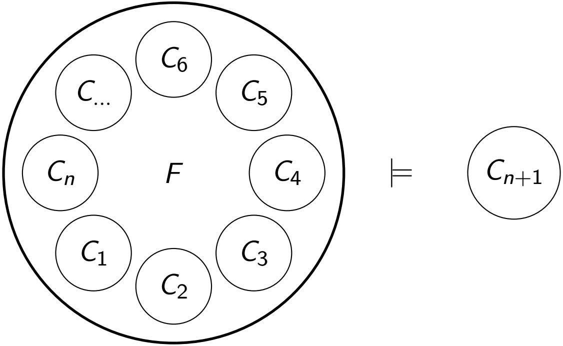
Saturation in FOL Theorem Proving



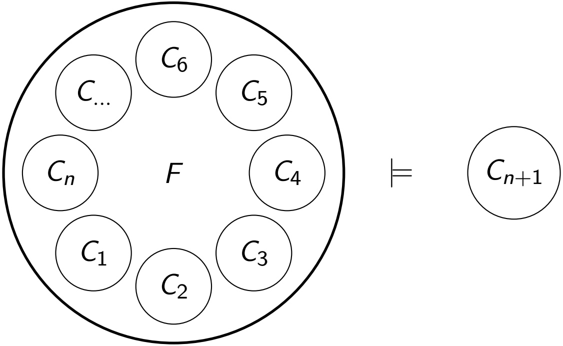
Saturation in FOL Theorem Proving



Saturation in FOL Theorem Proving



Saturation in FOL Theorem Proving



Out of memory!

Subsumption

Definition

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

$$\sigma(L) \subseteq^* M$$

where $\subseteq^*$ is the sub-multiset inclusion relation.

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

Subsumption - Examples

Example (propositional logic)

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

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

Subsumption - Examples

Example (propositional logic)

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

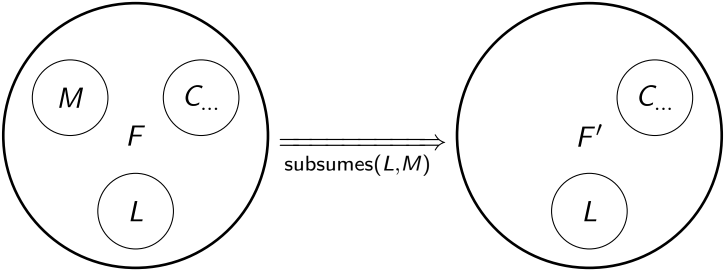
L subsumes M. It is "stronger" than M.

Example (FOL)

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

L 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)

$$\frac{L^* \vee l' \qquad \neg\sigma(l') \vee M^*}{\sigma(L^*) \vee M^*}$$

Subsumption Resolution

Resolution (Simplified)

$$\frac{L^* \vee l' \qquad \neg\sigma(l') \vee M^*}{\sigma(L^*) \vee M^*}$$

Definition

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

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

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

Subsumption Resolution - Example 1

Example (propositional logic)

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

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

Subsumption Resolution - Example 1

Example (propositional logic)

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

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

Subsumption Resolution - Example 2

Example (FOL)

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

Subsumption Resolution - Example 2

Example (FOL)

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

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

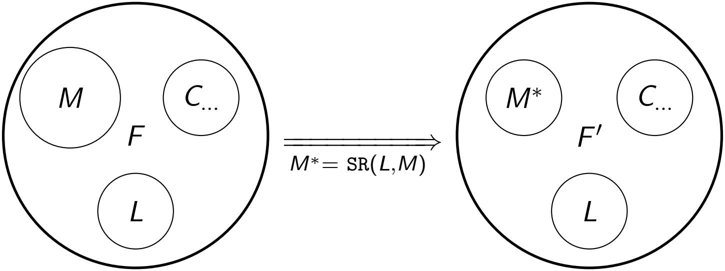
Subsumption Resolution - Example 2

Example (FOL)

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

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

Subsumption Resolution - 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

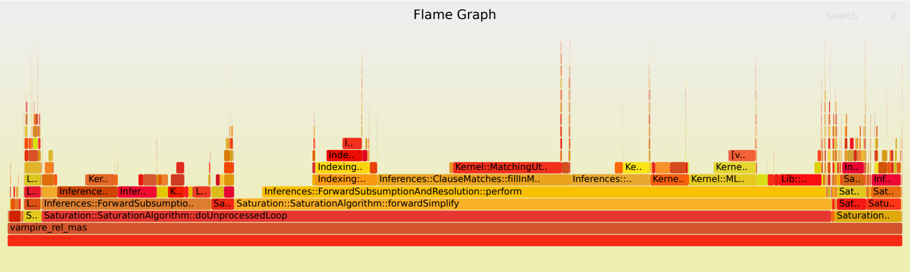


Figure: Typical profiling results for a TPTP problem (GRP001+6).

Building upon Previous Work

Previous Work

[Rath et al., 2022] introduced a SAT-based subsumption procedure.

- Encode subsumption as SAT problem.
- Tailor SAT solver to reason over substitutions.
- Use SAT solver to find a suitable substitution for subsumption.

Building upon Previous Work

Previous Work

[Rath et al., 2022] introduced a SAT-based subsumption procedure.

- Encode subsumption as SAT problem.
- Tailor SAT solver to reason over substitutions.
- Use SAT solver to find a suitable substitution for subsumption.

Our Contribution

We build upon the work of [Rath et al., 2022].

- Introduce constraints for subsumption resolution.
- Convert subsumption resolution to SAT problem.
- Integrate subsumption and subsumption resolution in Vampire.
- Optimize the simplifying loop of Vampire.

From Definition to Constraints

Theorem (Subsumption Resolution Constraints)

Clauses M and L are the main and side premise, respectively, of an instance of the subsumption resolution rule $\mathcal{SR}$ iff there exists a substitution σ that satisfies the following four properties:

existence $\exists i\, j.\, \sigma(l_i) = \neg m_j$

From Definition to Constraints

Theorem (Subsumption Resolution Constraints)

Clauses M and L are the main and side premise, respectively, of an instance of the subsumption resolution rule $\mathcal{SR}$ iff there exists a substitution σ that satisfies the following four properties:

existence	$\exists i\, j.\, \sigma(l_i) = \neg m_j$
uniqueness	$\exists j'.\, \forall i\, j.\, \big(\sigma(l_i) = \neg m_j \Rightarrow j = j'\big)$

From Definition to Constraints

Theorem (Subsumption Resolution Constraints)

Clauses M and L are the main and side premise, respectively, of an instance of the subsumption resolution rule $\mathcal{SR}$ iff there exists a substitution σ that satisfies the following four properties:

existence	$\exists i\, j.\, \sigma(l_i) = \neg m_j$
uniqueness	$\exists j'.\, \forall i\, j.\, \big(\sigma(l_i) = \neg m_j \Rightarrow j = j'\big)$
completeness	$\forall i.\, \exists j.\, \big(\sigma(l_i) = \neg m_j \vee \sigma(l_i) = m_j\big)$

From Definition to Constraints

Theorem (Subsumption Resolution Constraints)

Clauses M and L are the main and side premise, respectively, of an instance of the subsumption resolution rule $\mathcal{SR}$ iff there exists a substitution σ that satisfies the following four properties:

existence	$\exists i\, j.\, \sigma(l_i) = \neg m_j$
uniqueness	$\exists j'.\, \forall i\, j.\, \big(\sigma(l_i) = \neg m_j \Rightarrow j = j'\big)$
completeness	$\forall i.\, \exists j.\, \big(\sigma(l_i) = \neg m_j \vee \sigma(l_i) = m_j\big)$
coherence	$\forall j.\, \big(\exists i.\, \sigma(l_i) = m_j \Rightarrow \forall i.\, \sigma(l_i) \neq \neg m_j\big)$

SAT variables

Let

$$L = \{l_1, \dots, l_{|L|}\} \quad M = \{m_1, \dots, m_{|M|}\}$$

We define the following SAT variables:

- $b^+_{i,j} \Leftrightarrow \sigma(l_i) = m_j$
- $b^-_{i,j} \Leftrightarrow \sigma(l_i) = \neg m_j$

This encoding is an extension of the one proposed by [Rath et al., 2022].

$\sigma(l_i) = m_j$ means that the substitution $\sigma_{i,j}$ used to bind l_i to m_j is compatible with the other substitutions.

SAT variables - Example

$$L = p(x_1, x_2) \lor p(f(x_2), x_3)$$
$$M = \neg p(f(y), d) \lor p(g(y), c) \lor \neg p(f(c), e)$$

- $b^{-}_{1,1} \Leftrightarrow \{x_1 \mapsto f(y), x_2 \mapsto d\} \subseteq \sigma$

SAT variables - Example

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

- $b^{-}_{1,1} \Leftrightarrow \{x_1 \mapsto f(y), x_2 \mapsto d\} \subseteq \sigma$
- $b^{+}_{1,2} \Leftrightarrow \{x_1 \mapsto g(y), x_2 \mapsto c\} \subseteq \sigma$

SAT variables - Example

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

- $b^{-}_{1,1} \Leftrightarrow \{x_1 \mapsto f(y), x_2 \mapsto d\} \subseteq \sigma$
- $b^{+}_{1,2} \Leftrightarrow \{x_1 \mapsto g(y), x_2 \mapsto c\} \subseteq \sigma$
- $b^{-}_{1,3} \Leftrightarrow \{x_1 \mapsto f(c), x_2 \mapsto e\} \subseteq \sigma$

SAT variables - Example

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

- $b^{-}_{1,1} \Leftrightarrow \{x_1 \mapsto f(y), x_2 \mapsto d\} \subseteq \sigma$
- $b^{+}_{1,2} \Leftrightarrow \{x_1 \mapsto g(y), x_2 \mapsto c\} \subseteq \sigma$
- $b^{-}_{1,3} \Leftrightarrow \{x_1 \mapsto f(c), x_2 \mapsto e\} \subseteq \sigma$
- $b^{-}_{2,1} \Leftrightarrow \{x_2 \mapsto y, x_3 \mapsto d\} \subseteq \sigma$

SAT variables - Example

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

- $b^-_{1,1} \Leftrightarrow \{x_1 \mapsto f(y), x_2 \mapsto d\} \subseteq \sigma$
- $b^+_{1,2} \Leftrightarrow \{x_1 \mapsto g(y), x_2 \mapsto c\} \subseteq \sigma$
- $b^-_{1,3} \Leftrightarrow \{x_1 \mapsto f(c), x_2 \mapsto e\} \subseteq \sigma$
- $b^-_{2,1} \Leftrightarrow \{x_2 \mapsto y, x_3 \mapsto d\} \subseteq \sigma$
- $b^+_{2,2} \Leftrightarrow \{\bot\} \subseteq \sigma$

SAT variables - Example

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

- $b^-_{1,1} \Leftrightarrow \{x_1 \mapsto f(y), x_2 \mapsto d\} \subseteq \sigma$
- $b^+_{1,2} \Leftrightarrow \{x_1 \mapsto g(y), x_2 \mapsto c\} \subseteq \sigma$
- $b^-_{1,3} \Leftrightarrow \{x_1 \mapsto f(c), x_2 \mapsto e\} \subseteq \sigma$
- $b^-_{2,1} \Leftrightarrow \{x_2 \mapsto y, x_3 \mapsto d\} \subseteq \sigma$
- $b^+_{2,2} \Leftrightarrow \{\bot\} \subseteq \sigma$
- $b^-_{2,3} \Leftrightarrow \{x_2 \mapsto c, x_3 \mapsto e\} \subseteq \sigma$

SAT variables - Example

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

- $b^-_{1,1} \Leftrightarrow \{x_1 \mapsto f(y), x_2 \mapsto d\} \subseteq \sigma$
- $b^+_{1,2} \Leftrightarrow \{x_1 \mapsto g(y), x_2 \mapsto c\} \subseteq \sigma$
- $b^-_{1,3} \Leftrightarrow \{x_1 \mapsto f(c), x_2 \mapsto e\} \subseteq \sigma$
- $b^-_{2,1} \Leftrightarrow \{x_2 \mapsto y, x_3 \mapsto d\} \subseteq \sigma$
- $b^+_{2,2} \Leftrightarrow \{\bot\} \subseteq \sigma$
- $b^-_{2,3} \Leftrightarrow \{x_2 \mapsto c, x_3 \mapsto e\} \subseteq \sigma$

Constraint to SAT Encoding - Completeness

The constraints can be encoded using a simple procedure. For example the completeness constraint:

$$\forall i.\, \exists j.\, \big(\sigma(l_i) = \neg m_j \vee \sigma(l_i) = m_j\big)$$

Constraint to SAT Encoding - Completeness

The constraints can be encoded using a simple procedure. For example the completeness constraint:

$$\forall i.\, \exists j.\, \big(\sigma(l_i) = \neg m_j \vee \sigma(l_i) = m_j\big)$$
$$\forall i.\, \exists j.\, \big(b^{-}_{i,j} \vee b^{+}_{i,j}\big)$$

Constraint to SAT Encoding - Completeness

The constraints can be encoded using a simple procedure. For example the completeness constraint:

$$\forall i.\, \exists j.\, \big(\sigma(l_i) = \neg m_j \vee \sigma(l_i) = m_j\big)$$

$$\forall i.\, \exists j.\, \big(b^-_{i,j} \vee b^+_{i,j}\big)$$

$$\bigwedge_i \bigvee_j b^-_{i,j} \vee b^+_{i,j}$$

Constraint to SAT Encoding - Completeness

The constraints can be encoded using a simple procedure. For example the completeness constraint:

$$\forall i.\, \exists j.\, \big(\sigma(l_i) = \neg m_j \lor \sigma(l_i) = m_j\big)$$

$$\forall i.\, \exists j.\, \big(b^-_{i,j} \lor b^+_{i,j}\big)$$

$$\bigwedge_i \bigvee_j b^-_{i,j} \lor b^+_{i,j}$$

$$\bigwedge_i \bigvee_j b_{i,j}$$

SR Direct Encoding

SAT-based compatibility $$\bigwedge_i \bigwedge_j [b_{i,j} \Rightarrow \sigma_{i,j} \subseteq \sigma]$$

SAT-based existence $$\bigvee_i \bigvee_j b^-_{i,j}$$

SAT-based uniqueness $$\bigwedge_j \bigwedge_i \bigwedge_{i' \geq i} \bigwedge_{j' > j} \neg b^-_{i,j} \vee \neg b^-_{i',j'}$$

SAT-based completeness $$\bigwedge_i \bigvee_j b_{i,j}$$

SAT-based coherence $$\bigwedge_j \bigwedge_i \bigwedge_{i'} \neg b^+_{i,j} \vee \neg b^-_{i',j}$$

Structuring Variables

We define the following SAT variables:

- c_j is true iff m_j is the resolution literal.

$$c_j \Leftrightarrow \exists i.\, \sigma(l_i) = \neg m_j$$

.

Illustration

$$\begin{aligned}
c_1 &\Leftrightarrow b^-_{1,1} \vee \ldots \vee b^-_{n,1} \\
&\Leftrightarrow \sigma(l_1) = \neg m_1 \vee \ldots \vee \sigma(l_n) = \neg m_1 \\
c_2 &\Leftrightarrow b^-_{1,2} \vee \ldots \vee b^-_{n,2} \\
&\Leftrightarrow \sigma(l_1) = \neg m_2 \vee \ldots \vee \sigma(l_n) = \neg m_2 \\
&\vdots
\end{aligned}$$

SR Indirect Encoding

SAT-based compatibility
$$\bigwedge_i \bigwedge_j [b_{i,j} \Rightarrow \sigma_{i,j} \subseteq \sigma]$$

Structurality
$$\bigwedge_j \left[\neg c_j \vee \bigvee_i b^-_{i,j}\right] \wedge \bigwedge_j \bigwedge_i \left(c_j \vee \neg b^-_{i,j}\right)$$

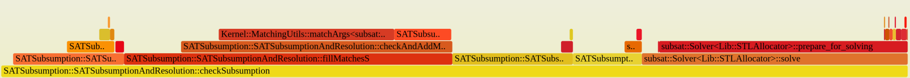
Revised existence
$$\bigvee_j c_j$$

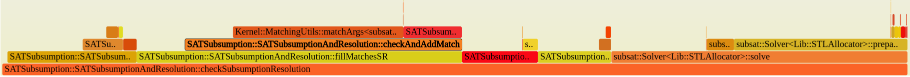
Revised uniqueness
$$AMO(\{c_j, j = 1, ..., |M|\})$$

Revised completeness
$$\bigwedge_i \bigvee_j b_{i,j}$$

Revised coherence
$$\bigwedge_j \bigwedge_i \left(\neg c_j \vee \neg b^+_{i,j}\right)$$

Setting up is Expensive





The setup time takes a significant portion of the total runtime. We can reduce the setup time by setting up both subsumption and SR at the same time.

Optimized Forward Loop

```
procedure Simplify(F, M)
    for L ∈ F \ {M} do
        if checkS(L, M) then
            F ← F \ {L}
            return ⊤
    for L ∈ F \ {M} do
        M* ← checkSR(L, M)
        if M* ≠ ⊥ then
            F ← F \ {L} ∪ {M*}
            return ⊤
    return ⊥
```

Optimized Forward Loop

```
procedure Simplify(F, M)
    for L ∈ F \ {M} do
        if checkS(L, M) then
            F ← F \ {L}
            return ⊤
    for L ∈ F \ {M} do
        M* ← checkSR(L, M)
        if M* ≠ ⊥ then
            F ← F \ {L} ∪ {M*}
            return ⊤
    return ⊥
```

```
procedure Simplify*(F, M)
    M* ← ⊥
    for L ∈ F \ {M} do
        if checkS(L, M) then
            F ← F \ {L}
            return ⊤
        if M* = ⊥ then
            M* ← checkSR(L, M)
    if M* ≠ ⊥ then
        F ← F \ {L} ∪ {M*}
        return ⊤
    return ⊥
```

Results - Graph

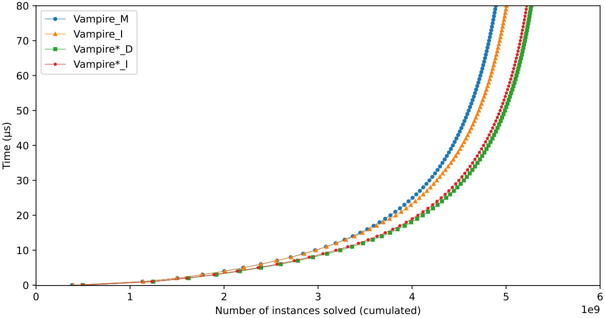


Figure: Comparison of the cumulative number of forward simplification loops solved by the different configurations of Vampire. The graph shows all the loops performed on all the TPTP problems.

Results - Tables

Prover	**Average**	**Std. Dev.**	**Speedup**
Vampire_M	$42.63\,\mu s$	$1609.06\,\mu s$	$0\,\%$
Vampire_I	$40.13\,\mu s$	$1554.52\,\mu s$	$6.2\,\%$
Vampire_D^*	$34.39\,\mu s$	$1047.85\,\mu s$	$23.9\,\%$
Vampire_I^*	$34.55\,\mu s$	$250.25\,\mu s$	$23.4\,\%$

Table: Average and standard deviation of the runtime of forward simplification loop on the TPTP problems.

Prover	**Total Solved**	**Gain/Loss**
Vampire_M	10 555	baseline
Vampire_D^*	10 667	$(+141, -29)$
Vampire_I^*	10 658	$(+133, -30)$

Table: Number of TPTP problems solved by the different configurations of Vampire. The options `-sa otter -av off -t 60` were used for all runs.

Future Work

- Heuristically choose between direct and indirect encoding
- Extend technique to subsumption demodulation
- Investigate the drop in variance.
- Extend subsumption resolution to use an m.g.u. for the resolution literal.

Conclusion

- We have introduced a new method for subsumption resolution.
- SAT-based methods harness the power of modern SAT solvers.
- The setup time of the SAT-based methods is significant. However, we can reduce it by combining the setup of subsumption and SR.
- SAT-based methods are competitive with the state of the art.
- SAT-based methods are also very flexible and can be fine-tuned easily.

References

Rath, J., Biere, A., and Kovács, L. (2022).
First-Order Subsumption via SAT Solving.
In *FMCAD*, page 160.