A Unifying Splitting Framework

Gabriel Ebner
(joint work with Jasmin Blanchette and Sophie Tourret)

Vrije Universiteit Amsterdam

2021-06-09
Motivation

Saturation and redundancy

Splitting calculus

Local fairness and saturation

Local redundancy and locking

Given-clause procedure

Conclusion
Splitting

\[ C \lor D \]

\[
\frac{C \lor D}{C \quad D}
\]

(if \( C \) and \( D \) are clauses with disjoint variables)
Splitting approaches

- Splitting without backtracking (Riazanov and Voronkov 2001)
- Labelled splitting (Fietzke and Weidenbach 2009)
- Avatar (Voronkov 2014)
  - Very effective: solves 421 previously unsolved problems
Avatar example in our notation
Motivation

Saturation and redundancy

Splitting calculus

Local fairness and saturation

Local redundancy and locking

Given-clause procedure

Conclusion
Saturation framework

- General conditions for completeness of saturation provers
- Bachmair and Ganzinger 2001 (handbook article)
- Waldmann et al. 2020
Formulas and consequences

- Abstract formulas $\mathbf{F} = \{C, D, \ldots\}$
- $\bot \in \mathbf{F}$
- Consequence relation $M \models C \quad (M \subseteq \mathbf{F})$
Abstract formulas $F = \{C, D, \ldots\}$

$\bot \in F$

Consequence relation $M \models C \quad (M \subseteq F)$

- $\{\bot\} \models C$
- $\{C\} \models C$
- $M \models C$ implies $M \cup N \models C$
- $\forall C \in N (M \models C)$ and $N \models C$ implies $M \models C$
Formulas and consequences

- Abstract formulas $\mathbf{F} = \{C, D, \ldots\}$
- $\bot \in \mathbf{F}$
- Consequence relation $M \models C \ (M \subseteq \mathbf{F})$
  - $\{\bot\} \models C$
  - $\{C\} \models C$
  - $M \models C$ implies $M \cup N \models C$
  - $\forall C \in N (M \models C)$ and $N \models C$ implies $M \models C$

- For example: $\mathbf{F} =$ set of first-order clauses
- $\{p(x) \lor \neg q(x), q(c)\} \models \{p(c)\}$
Formula redundancy

- $\text{Red}_F : \mathcal{P}(F) \rightarrow \mathcal{P}(F)$
  - $\text{Red}_F(M) \subseteq \text{Red}_F(M \cup N)$
  - $N \setminus \text{Red}_F(N) \models \bot \iff N \models \bot$
  - $\text{Red}_F(M \cup \text{Red}_F(M)) = \text{Red}_F(M)$
Formula redundancy

\[ \text{Red}_F : \mathcal{P}(F) \to \mathcal{P}(F) \]

\[ \text{Red}_F(M) \subseteq \text{Red}_F(M \cup N) \]

\[ N \setminus \text{Red}_F(N) = \bot \iff N = \bot \]

\[ \text{Red}_F(M \cup \text{Red}_F(M)) = \text{Red}_F(M) \]

Standard redundancy criterion:
“redundant if entailed by smaller formulas”
\[ C \in \text{Red}_F(M) \iff \{ D \in M \mid D \prec C \} \vdash C \]

\[ p(c) \in \text{Red}_F(\{p(x)\}) \]

\[ p(t) \in \text{Red}_F(\{t = s, p(s)\}) \text{ (assuming } s \prec t) \]

\[ p(y) \notin \text{Red}_F(\{p(x)\}) \]

\[ p(x) \lor \neg p(x) \in \text{Red}_F(\emptyset) \]
Inference “redundancy”

- Inferences \((C_n, \ldots, C_1, D) \in Inf\)

- \(Red_I : \mathcal{P}(F) \rightarrow \mathcal{P}(Inf)\)
  - \(Red_I(M) \subseteq Red_I(M \cup N)\)
  - \(Red_I(\mathcal{M} \cup Red_F(\mathcal{M})) = Red_I(\mathcal{M})\)
  - \(D \in M\) implies \((C_n, \ldots, C_1, D) \in Red_I(M)\)

- \(Inf \setminus Red_I(\cdot) = \) inferences that must be performed
Dynamic completeness

**Theorem**

If \((\text{Inf}, (\text{Red}_F, \text{Red}_1))\) is statically complete, then it is also dynamically complete.
Extension by labels
Motivation

Saturation and redundancy

**Splitting calculus**

Local fairness and saturation

Local redundancy and locking

Given-clause procedure

Conclusion
A-formulas

- Propositional variables $V$
- $\text{fml}(\cdot) : V \rightarrow F \cup F_\sim$

- A-formula: $C \leftarrow \{a_1, \ldots, a_n\}$
  - $C \in F$
  - $a_i \in A = V \cup \neg V$

- Intended interpretation: $\text{fml}(a_1) \land \cdots \land \text{fml}(a_n) \rightarrow C$
Interpretations

▶ \( J \subseteq A \) is a propositional interpretation if it contains exactly one of \( \nu \) or \( \neg \nu \) for all \( \nu \in V \)

▶ \( C \leftarrow A \) is enabled in \( J \) if \( A \subseteq J \)

▶ \( \lfloor C \leftarrow A \rfloor = C \)

▶ \( \lfloor N \rfloor \supseteq N_J \) = all enabled formulas in \( N \)

▶ \( \bot \leftarrow A \) is called a propositional clause
  ▶ \( \bot \leftarrow \{a, \neg b\} \) means \( \neg \text{fml}(a) \lor \text{fml}(b) \)

▶ \( N_\bot \) = all propositional clauses in \( N \)
Consequence relation

- Assume consequence relations $\models \subseteq \models$ on $\mathbf{F}$
- $\mathcal{M} \models \mathcal{N}$ if and only if $\mathcal{M}_j \models [\mathcal{N}]$ for every $j$ in which $\mathcal{N}$ is enabled
- $\mathcal{M} \models \mathcal{N}$ if and only if $\text{fml}(J) \cup \mathcal{M}_j \models [\mathcal{N}]$ for every $j$ in which $\mathcal{N}$ is enabled.
Redundancy

- Assume redundancy criterion \((F\text{Red}_F, F\text{Red}_I)\) on \(F\)
- \(C \in F\text{Red}_F(\mathcal{N}_\mathcal{J})\) for all \(\mathcal{J} \supseteq A\); or
- exists \(C \leftarrow B \in \mathcal{N}\) such that \(B \subset A\).
Inference rules

\[
\begin{align*}
(C_i \leftarrow A_i)_{i=1}^n & \quad \text{BASE} \\
D \leftarrow A_1 \cup \cdots \cup A_n & \\
\end{align*}
\]

\[
\begin{align*}
(\bot \leftarrow A_i)_{i=1}^n & \quad \text{UNSAT} \\
\bot & \\
\end{align*}
\]

where \( C_n \cdots C_1 \)

\[
\begin{align*}
D & \quad \text{FlInf} \\
\end{align*}
\]

where \( (\bot \leftarrow A_i)_{i=1}^n \) is unsat

\[
\begin{align*}
\text{where} & \\
\text{where} & \\
\end{align*}
\]
Admissible inference rules

Theorem
The following are sound inference rules w.r.t. $\equiv$, and inferences with = are also simplification rules w.r.t. SRed:

\[ \begin{align*}
  C & \leftarrow A \\
  \bot & \leftarrow \{\neg a_1, \ldots, \neg a_n\} \cup A \quad (C_i \leftarrow \{a_i\})_{i=1}^n
\end{align*} \]

\[ \begin{align*}
  (\bot \leftarrow A_i)_{i=1}^n & \quad C \leftarrow A \cup B \\
  (\bot \leftarrow A_i)_{i=1}^n & \quad C \leftarrow B
\end{align*} \]

\[ \begin{align*}
  \text{TRIM} & \quad (\bot \leftarrow A_i)_{i=1}^n \\
  \text{APPROX} & \quad \bot \leftarrow \{\neg a\} \cup A
\end{align*} \]

(where $\{\bot \leftarrow A_i\}_{i=1}^n \cup \{\bot \leftarrow A\} \equiv \{\bot \leftarrow B\}$)

\[ \ldots \]
Motivation

Saturation and redundancy

Splitting calculus

**Local fairness and saturation**

Local redundancy and locking

Given-clause procedure

Conclusion
Levels of refinement

1. Base calculus ($F_{Inf}, F_{Red}$)
2. Splitting calculus ($S_{Inf}, S_{Red}$)
3. Model-guided prover MG
4. Locking prover L
5. Given-clause procedure AV
Standard completeness

Assume that \((F_{\text{Inf}}, F_{\text{Red}})\) is statically complete.

**Theorem**

\((S_{\text{Inf}}, S_{\text{Red}})\) is statically complete
(and hence also dynamically complete).
Local completeness

Definition

\((\mathcal{N}_i)_i\) is \textit{locally fair} if either

1. \(\bot \in \bigcup_i \mathcal{N}_i\) or
2. exists \(J \models (\mathcal{N}_\infty)_J\) such that \(\text{FlInf}(\mathcal{N}_\infty)_J \subseteq \bigcup_i \text{FRed}_1((\mathcal{N}_i)_J)\)

Theorem

If \((\mathcal{N}_i)_i\) is locally fair and \(\mathcal{N}_0 \models \{\bot\}\), then \(\bot \in \bigcup_i \mathcal{N}_i\).
Model-guided prover

▶ Derivations \((\mathcal{J}_0, \mathcal{N}_0) \xrightarrow{\text{MG}} (\mathcal{J}_1, \mathcal{N}_1) \xrightarrow{\text{MG}} \cdots\)

▶ Transition rules:

**DERIVE** \((\mathcal{J}, \mathcal{N} \cup \mathcal{M}) \xrightarrow{\text{MG}} (\mathcal{J}, \mathcal{N} \cup \mathcal{M}')\) if \(\mathcal{M} \subseteq SRed_F(\mathcal{N} \cup \mathcal{M}')\)

**SWITCH** \((\mathcal{J}, \mathcal{N}) \xrightarrow{\text{MG}} (\mathcal{J}', \mathcal{N})\) if \(\mathcal{J}' \models \mathcal{N}_\bot\)

**UNSAT** \((\mathcal{J}, \mathcal{N}) \xrightarrow{\text{MG}} (\mathcal{J}, \mathcal{N} \cup \{\bot\})\) if \(\mathcal{N}_\bot \models \{\bot\}\)
**Topology on interpretations**

- Equip the set of propositional interpretations with the product topology
  - “topology of pointwise convergence”

- Clearly homeomorphic to the Cantor space $2^\omega$
  - Complete metric space
  - Compact

- Every sequence $(\mathcal{I}_i)_i$ has a convergent subsequence $(\mathcal{I}'_i)_i$
- We call $\lim_{i \to \infty} \mathcal{I}'_i$ a limit point

- Evaluating assertions is continuous

---

1 in analogy to other “limits” of clause sets
Motivation

Saturation and redundancy

Splitting calculus

Local fairness and saturation

**Local redundancy and locking**

Given-clause procedure

Conclusion
Local redundancy

- $\mathcal{C} \in SRed_F(\mathcal{M})$ captures “global” redundancy
  - can be removed permanently
  - does not depend on model

- $\lfloor \mathcal{C} \rfloor \in FRed_F(\mathcal{M}_I)$ captures “local” redundancy
  - can only be removed \textit{temporarily}
  - relative to current model
Extra set for A-formulas that are locally redundant depending on some finite subset of the model

\[
\begin{align*}
\text{LIFT} & \quad (\mathcal{J}, \mathcal{N}, \mathcal{L}) \xrightarrow{L} (\mathcal{J}, \mathcal{N}' \cup \|\mathcal{U}\|, \mathcal{L} \setminus \mathcal{U}) \\
& \quad \text{if } (\mathcal{J}, \mathcal{N}) \xrightarrow{\text{MG}} (\mathcal{J}', \mathcal{N}') \text{ and } \mathcal{U} = \{(B, C \leftarrow A) \in \mathcal{L} \mid B \not\subseteq \mathcal{J}' \text{ and } A \subseteq \mathcal{J}'\}
\end{align*}
\]

\[
\begin{align*}
\text{LOCK} & \quad (\mathcal{J}, \mathcal{N} \cup \{C \leftarrow A\}, \mathcal{L}) \xrightarrow{L} (\mathcal{J}, \mathcal{N}, \mathcal{L} \cup \{(B, C \leftarrow A)\}) \\
& \quad \text{if } B \subseteq \mathcal{J} \text{ and } C \in FRed_F(\mathcal{N}_{\mathcal{J}'}) \text{ for all } \mathcal{J}' \supseteq A \cup B
\end{align*}
\]
Counterexamples
Completeness
Motivation
Saturation and redundancy
Splitting calculus
Local fairness and saturation
Local redundancy and locking

**Given-clause procedure**

Conclusion
Required and allowed inferences
Strongly finitary functions
Conditions
Motivation

Saturation and redundancy

Splitting calculus

Local fairness and saturation

Local redundancy and locking

Given-clause procedure

Conclusion
Conclusion

▶ Completeness of splitting provers depends on subtle details
  ▶ Clause-selection strategies that are complete for nonsplitting provers are not necessarily complete for splitting provers
  ▶ Fairness not only requires a minimum of inferences but also a maximum

▶ Completeness theorem for an Avatar-like given-clause procedure
  ▶ Requires a very very strong restriction on locking
  ▶ No restriction on the models

▶ Can we reduce the restriction on locking by requiring “regular” sequences of propositional models?