SGGS Theorem Proving: an Exposition

Maria Paola Bonacina

Dipartimento di Informatica
Università degli Studi di Verona
Verona, Italy

July 2014

Joint work with David Plaisted
Motivation: Why SGGS?

Model representation

Inferences

Refutational Completeness

Goal Sensitivity

Discussion
Motivation

A first-order theorem-proving method simultaneously

- DPLL-style model based
- Proof confluent
- Semantically guided
- Goal sensitive
DPLL-style model based

- Derivation state includes candidate (partial) model
- Inference and search (for model) guide each other (e.g., CDCL in DPLL)
- Inference as model transformation
Proof confluent

- **Confluence**: diamond property: \( \backslash / \backslash \Rightarrow \backslash / \slash \)
- **Proof confluence**: Committing to an inference never prevent proof
- **No backtracking**
Semantically guided

- Semantic guidance by a given initial interpretation $\mathcal{I}$
- In theory and manual examples: e.g., based on sign
- In practice: problems and knowledge bases may come with it
- SGGS: semantic guidance and model-based style connected; $\mathcal{I}$ as starting point and default
Goal sensitive

- **Notion of goal:**
  - \( H \models \varphi \)
  - \( H \cup \{\neg \varphi\} \vdash \bot \)
  - \( H \cup \{\neg \varphi\} \leadsto S \) set of clauses
  - \( S = T \uplus iSOS \) where \( H \leadsto T, \{\neg \varphi\} \leadsto iSOS \)
  - \( S = T \uplus iSOS, iSOS \) input set of support

- Alternatively: \( S = T \uplus iSOS \) with \( T \) consistent, \( iSOS = S \setminus T \)
- Generate only clauses **connected** with \( iSOS \)
Motivation summary

- A first-order reasoning method with all these properties?!
- Yes!!!

**SGGS**

*Semantically Guided Goal Sensitive reasoning*
Model representation from PL to FOL:

- **DPLL:** Trail of literals $L_1, \ldots, L_n$
- **SGGS:**
  - Initial interpretation $\mathcal{I}$
  - Sequence of constrained clauses with selected literals
    $\Gamma = A_1 \triangleright C_1[L_1], \ldots, A_n \triangleright C_n[L_n]$
  - That modify $\mathcal{I}$
What does a constrained clause represent?

Its constrained ground instances (cgi’s) or ground instances satisfying the constraints

Example:

- \( x \not\equiv y \triangleright P(x, y) \)
- \( P(a, b) \in Gr(x \not\equiv y \triangleright P(x, y)) \)
- \( P(b, b) \not\in Gr(x \not\equiv y \triangleright P(x, y)) \)
Literal selection

- Every literal in sequence is either $I$-true or $I$-false
- $I$-true: all cgi’s true in $I$
- $I$-false: all cgi’s false in $I$
- Literal tells truth value of all its cgi’s
- Prefer $I$-false literals for selection:
  If clause has $I$-false literals, one is selected
Interpretation $\mathcal{I}[\Gamma]$ represented by clause sequence $\Gamma$

- Partial interpretation $\mathcal{I}^p(\Gamma|_j)$ for prefix $\Gamma|_j$
- For each clause $A_j \triangleright C_j[L_j]$ take its proper constrained ground instances (pcgi):
  - Not satisfied by $\mathcal{I}^p(\Gamma|_{j-1})$
  - Satisfiable by adding the pcgi of $L_j$
- $\mathcal{I}[\Gamma]$: complete $\mathcal{I}^p(\Gamma)$ by consulting $\mathcal{I}$ whenever $\mathcal{I}^p(\Gamma)$ does not determine truth value
- $\mathcal{I}[\Gamma]$ is $\mathcal{I}$ modified to satisfy the pcgi’s of the selected literals
Example

- $\mathcal{I}$: all negative
- Sequence $\Gamma$: $[P(x)]$, $\text{top}(y) \neq g \triangleright [Q(y)]$, $z \neq c \triangleright [Q(g(z))]$
- Interpretation $\mathcal{I}[\Gamma]$:
  - $\mathcal{I}[\Gamma] \models P(x)$
  - $\mathcal{I}[\Gamma] \models Q(t)$ for all ground terms $t$ whose top symbol is not $g$
  - $\mathcal{I}[\Gamma] \models Q(g(t))$ for all ground terms $t$ other than $c$
  - $\mathcal{I}[\Gamma] \not\models L$ for all other positive literals $L$
SGGS-Derivation

- Input set of clauses $S$
- Initial interpretation $I$
- Derivation $\Gamma_0 \vdash \Gamma_1 \vdash \ldots \Gamma_j \vdash \ldots$
- $\Gamma_0$ is empty, $I[\Gamma_0]$ is $I$
- $\Gamma_j$ generated from $\Gamma_{j-1}$, $S$, and $I$ by an SGGS inference rule
- Termination: either $\Gamma_k$ contains empty clause (refutation) or no rule applies
Assignment function: intuition

- Propositional clauses: $L$ and $\neg L$ are complementary. If $L$ is true in the current model, $\neg L$ is not.
  Boolean Constraint Propagation

- First-order constrained clauses: $A \triangleright [L]$ and $B \triangleright [M]$ have complementary cgi’s

- Semantic guidance: reasoning relative to $\mathcal{I}$: $L$ is $\mathcal{I}$-true and $M$ is $\mathcal{I}$-false
Assignment function: definition

- Every sequence $\Gamma$ in derivation equipped with (a set of) assignment functions (one per clause)
- Maps $\mathcal{I}$-true literal $L$ not selected in $A_i \triangleright C_i[L_i]$ to preceding clause $A_j \triangleright C_j[L_j]$ ($j < i$) with $\mathcal{I}$-false selected literal
- All cgi’s of $A_i \triangleright L$ appear negated among pcgi’s of $A_j \triangleright L_j$
- $A_i \triangleright C_i[L_i]$ depends on $A_j \triangleright C_j[L_j]$
Consider an $\mathcal{I}$-all-true clause with selected literal not assigned: $L_1 \lor \ldots \lor L_{k-1} \lor [L_k]$

By the assignment, $L_1 \ldots L_{k-1}$ are all false in $\mathcal{I}[\Gamma]$

Thus $L_k$ is implied

(like an implied literal by BCP in DPLL)
Assignment function: conflict + explanation à la CDCL

Consider an $\mathcal{I}$-all-true clause with selected literal assigned:

$$L_1 \lor \ldots \lor L_{k-1} \lor [L_k]$$

By the assignment, $L_1 \ldots L_{k-1}[L_k]$ are all false in $\mathcal{I}[\Gamma]$

Thus we have a conflict (like in DPLL-CDCL)

Explanation: by SGGS-resolution (coming soon)
Main inference mechanisms

1. **Instance generation**: extend current candidate model
2. **Resolution**: amend candidate model removing inconsistencies or generate $\bot$ if impossible
3. **Partitioning inferences**: amend candidate model pulling out duplications
   - Introduce constraints to capture different sets of ground instances
4. **Deletion** of disposable clauses (model-based redundancy)
SGGS-Extension

$\Gamma \vdash \Gamma'$

- Take input clause $C$ and find instance $E$ not satisfied by $\mathcal{I}[\Gamma]$ and such that all its literals are either $\mathcal{I}$-true or $\mathcal{I}$-false
- Find a place in $\Gamma$ where $E$ can be inserted so that the $\mathcal{I}$-true literals can be assigned properly
- $E$ satisfied by $\mathcal{I}[\Gamma']$
- Lifting Theorem:
  For all ground instance $C_\mu$ not satisfied by $\mathcal{I}[\Gamma]$, there is SGGS-extension of $\Gamma$ into $\Gamma'$ so that $C_\mu$ satisfied by $\mathcal{I}[\Gamma']$
SGGS-Resolution

- Model-based: resolution in the current candidate model
- Resolves clauses $B \triangleright D[M]$ and $A \triangleright C[L]$ in the sequence, not in the input set
- Only selected literals are resolved upon
- One $\mathcal{I}$-true and one $\mathcal{I}$-false
- $B \triangleright D[M]$ is $\mathcal{I}$-all-true and precedes $A \triangleright C[L]$
- SGGS-resolution uses matching: $L = \neg M \vartheta$ and $A \supset B \vartheta$
- Resolvent replaces $A \triangleright C[L]$
Inside SGGS-Resolution

Theorem:
Under the hypotheses of SGGS-resolution:

- $A \triangleright L$ has no pcgi’s
- The atoms of the cgi’s of $A \triangleright L$ that $A \triangleright C[L]$ would capture are covered by $B \triangleright D[M]$
- $A \triangleright C[L]$ replaced by resolvent which captures the cgi’s of $C \setminus \{L\}$
Example of SGGS-Resolution

- $\mathcal{I}$: all negative
- $\Gamma \vdash \Gamma'$
  - $\Gamma$: $[P(x)], [Q(y)], x \not\equiv c \triangleright \neg P(f(x)) \lor \neg Q(g(x)) \lor [R(x)], [\neg R(c)], \neg P(f(c)) \lor \neg Q(g(c)) \lor [R(c)]$
  - $\Gamma'$: $[P(x)], [Q(y)], x \not\equiv c \triangleright \neg P(f(x)) \lor \neg Q(g(x)) \lor [R(x)], [\neg R(c)], \neg P(f(c)) \lor [\neg Q(g(c))]]$
Assignment function + SGGS-resolution: explanation

- Recall that an $I$-all-true clause with selected literal assigned is a conflict clause:
  \[ L_1 \lor \ldots \lor L_{k-1} \lor [L_k] \]

- It moves to the left so that $L_k$ enters $I[\Gamma]$: model fixing

- Then it SGGS-resolves with following clause replacing it by SGGS-resolvent amending the model further
Partitioning inferences

- Replace a clause by its partition
- Partition of a clause: a set of clauses that capture the same cgi’s, and have disjoint selected literals (no cgi’s with the same atoms)

- Clause: \( \text{true} \supset P(x, y) \) (or simply \( P(x, y) \))
- Partition: \( \text{true} \supset P(f(z), y) \), \( \text{top}(x) \neq f \supset P(x, y) \)
Example of partitioning inference

- $\Gamma \vdash \Gamma'$
- $\Gamma$: $[P(x)], [Q(y)], \ x \not\equiv c \triangleright \neg P(f(x)) \lor \neg Q(g(x)) \lor [R(x)], [\neg R(c)], \neg P(f(c)) \lor [\neg Q(g(c))]$
- $\Gamma'$:
  $[P(x)], \ top(y) \not\equiv g \triangleright [Q(y)], \ z \not\equiv c \triangleright [Q(g(z))], [Q(g(c))], \ x \not\equiv c \triangleright \neg P(f(x)) \lor \neg Q(g(x)) \lor [R(x)], [\neg R(c)], \neg P(f(c)) \lor [\neg Q(g(c))]$
Deletion of disposable clauses

- **pcgi’s**: cgi’s of selected literal that can be added to current candidate model
- **ccgi’s**: cgi’s of selected literal that contradict current candidate model:
  - cgi of clause not satisfied by induced partial interpretation
  - cgi of selected literal appears negated in induced partial interpretation
- A clause with neither is **useless for model search** in SGGS
- **Disposable**: (non-empty) clause with neither pcgi’s nor ccgi’s
- When deleted, all clauses depending on it also deleted
Inference control

➢ Bundled derivations: all inferences are bundled
➢ Bundled inferences: macro-inferences, e.g.: an SGGS-extension followed by a series of SGGS-resolutions until an $\mathcal{I}$-all-true resolvent is generated
➢ Recall that an $\mathcal{I}$-all-true clause gives us either a lemma (implied literal) or a conflict
Refutational completeness

- $S$: input set of clauses
- $S$ unsatisfiable: any fair SGGS-derivation terminates with refutation
- $S$ satisfiable: derivation may be infinite; its limiting sequence represents model
Proof of refutational completeness: building blocks

- A convergence ordering $>^c$ on clause sequences: ensures that there is no infinite descending chain of sequences of bounded length
- A notion of fairness for SGGS-derivations: ensures that the procedure does not get stuck working on longer prefixes when shorter ones can be reduced
- A notion of limiting sequence for SGGS-derivations: every prefix stabilizes eventually
Convergence and decreasingness theorems

- **Convergence theorem:** A derivation that is a non-ascending chain admits limiting sequence
- **Decreasingness theorem:** A bundled derivation forms a non-ascending chain
Theorem:
For all initial interpretations $\mathcal{I}$ and sets $S$ of first-order clauses, if $S$ is unsatisfiable, any fair bundled SGGS-derivation is a refutation

Idea of proof:
If not, infinitely many irredudant SGGS-extensions apply; infinite derivation with infinite limiting sequence, that gets reduced in a finite prefix that had already converged: contradiction
Goal sensitivity I

▫ \( \mathcal{I} \models T \) and \( \mathcal{I} \nvDash iSOS \)

▫ Two ground clauses connected: complementary literals

▫ Goal-relevant clauses: closure of the set of ground instances of clauses in \( iSOS \) wrt connection and resolution

▫ \( \Gamma \) is goal-relevant if all ground instances of all its clauses are
Goal sensitivity II

**Theorem:** SGGS only generates goal-relevant clause sequences

**Idea of proof:**
use assignments of $\mathcal{I}$-true literals to $\mathcal{I}$-false ones to connect literals
Summary

SGGS is simultaneously

- First order
- DPLL-style model based
- Proof confluent
- Semantically guided
- Refutationally complete
- Goal sensitive
Future work

- SGGS as an abstract transition system
- Practical inference control (e.g., partitioning inferences)
- Implementation
- Non-trivial initial interpretations
- SGGS for model building and decision procedures
- Extension to equality and theory reasoning

Towards a semantically-oriented style of theorem proving which may pay off for hard problems or new domains
References

- Constraint manipulation in SGGS. 28th Workshop on Unification (UNIF), Vienna, July 2014.