Variability Management

Prof. Dr. Christoph Weidenbach
Automation of Logic

Application

Logic

Calculus

Implementation

Automatic Computation of Properties
Examples by PhD Thesis

• Dr. Matthias Horbach: second-order logic decidability
• Dr. Carsten Ihlemann: local theory extensions
• Tinxiang Lu: verifying correctness of PASTRY
• Arnaud Fietzke: combining first-order and prob. reasoning
• Patrick Wischnewski: reasoning in large ontologies
• ?: variability management (PROSTEP, Siemens)
Reasoning in Large Ontologies

develop “semantic” GOOGLE
Configuration Today

The car industry:

Opel Corsa
Today’s Architecture

Sales Views

Logistics Views

Engineering Views

... Views

Sales Software

MM Software

CAX Software

Product

Application

Application + DB

Paper, People
Future’s Architecture

Sales Views

Logistics Views

Engineering Views

XXX Views

Product Interface

DB

Product Specification

Product Reasoning

Application

Application
Scientific View

- Sales Views
- Logistics Views
- Engineering Views
- XXX Views

Reasoning Interface

- Logic Formulas
- Theorem Provers
- DB

Application
Concrete Example

- v.control
- v.Control SPASS Interface
- Propositional Logic
- DB
- SPASS
- Application

Application
Opel Corsa

in cooperation with
Prof. Dr. Georg Rock, Uni App Sc Trier, PROSTEP IMP
Daniel Doenigus, PROSTEP IMP
Propositional Logic

• Language: propositional variables can be true (1) or false (0)
• Connectives: ⇒ implication, ¬ negation, ∨ disjunction, ∧ conjunction
• Clause: disjunction of variables or their negations (literal)
• Validity: a formula is valid iff it is true for all possible assignments
• Assignment: setting all propositional variables 1 or 0, can also be expressed by showing the true literals
• we write \( M \models C \) if the clause \( C \) is true by assignment \( M \)
• SAT: propositional satisfiability, find an assignment such that for a set of clauses all clauses are valid in the assignment
Unit Propagation

\[ \text{UProp}(N, M) \]
\[
\text{while (there is a clause } C' \lor L \in N \text{ such that } M \models \neg C' \text{ and } L \notin M \text{ and } \neg L \notin M) \]
\[
M := M \cup \{L\};
\]
return \(M\);

\[ \text{UProp}\left(\{\neg A \lor \neg B \lor E, \neg A \lor B, \neg E, D, A\}, \emptyset\right) \]
\[
\rightarrow M = \emptyset
\]
\[
\rightarrow M = \{\neg E\}
\]
\[
\rightarrow M = \{\neg E, D\}
\]
\[
\rightarrow M = \{\neg E, D, A\}
\]
\[
\rightarrow M = \{\neg E, D, A, B\}\]
DPLL Procedure

\[
\text{DPLL}(N, M) \\
\text{if for all } C \in N \text{ we have } M \models C \text{ return true;} \\
\text{if there is some } C \in N \text{ with } M \models \neg C \text{ return false;} \\
\text{select a variable } P \text{ occuring in } N \text{ but not in } M; \\
\text{if (DPLL}(N, \text{UProp}(N, M \cup \{P\}))) \text{ then} \\
\text{ return true;} \\
\text{else} \\
\text{return } \text{DPLL}(N, \text{UProp}(N, M \cup \{\neg P\}));
\]

\[
\neg A \lor \neg B \lor E \\
\neg A \lor B \\
\neg E \\
A \lor D \\
\text{DPLL}(N, \emptyset) \\
\text{DPLL}(N, \text{UProp}(N,\{A\})) \\
\text{DPLL}(N, \text{UProp}(N,\{\neg A\})) \\
\text{DPLL}(N, \{A, B, \neg E\}) \\
\text{DPLL}(N, \{\neg A, D, \neg E\})
\]

DPLL is sound and complete and terminating for SAT.
Propositional Logic Formulas

Corsa ⇒ Wheels ∧ Engines

4-Holes ⇒ Wheels
5-Holes ⇒ Wheels
4-Holes ⇒ ¬5-Holes
5-Holes ⇒ ¬4-Holes

Diesel ⇒ Engines
Gasoline ⇒ Engines
Diesel ⇒ ¬Gasoline
Gasoline ⇒ ¬Diesel

Diesel ⇒ ¬4-Holes

Reasoning:
Corsa ⇒ Wheels, Engines
4-Holes ⇒ ¬5-Holes, ¬Diesel, Gasoline
Gasoline ⇒ ¬Diesel
Challenge: Scalability

• worst case SAT searches $2^n$ nodes
• before 2009: approx. 1500 nodes
• in 2009: v.control + SPASS approx. 3000 nodes
• in x years: for a reasonable product approx. 60000 nodes
• SAT Seminar:
• contact us on student assistant jobs, bachelor-master-PhD thesis

Thank you for your attention