An Introduction to Satisfiability Modulo Theories

Albert Oliveras and Enric Rodríguez-Carbonell

Deduction and Verification Techniques

Session 1

Fall 2009, Barcelona

Departament de Llenguatges i Sistemes Informàtics

UNIVERSITAT POLITÈCNICA DE CATALUNYA

Overview of the session

- Motivation
- SMT
- Theories of Interest
- Eager approach
- Lazy approach
  - Optimizations
  - Theory propagation
  - DPLL(T) in depth

UPC
Departament de Llenguatges i Sistemes Informatics
UNIVERSITAT POLITECNICA DE CATALUNYA

Introduction

- Historically, automated reasoning $\equiv$ uniform proof-search procedures for FO logic
- Little success: is FO logic the best compromise between expressivity and efficiency?
- Current trend is to gain efficiency by:
  - addressing only (expressive enough) decidable fragments of a certain logic
  - incorporate domain-specific reasoning, e.g:
    - arithmetic reasoning
    - equality
    - data structures (arrays, lists, stacks, ...)

UPC
Departament de Llenguatges i Sistemes Informatics
UNIVERSITAT POLITECNICA DE CATALUNYA

Introduction (2)

Examples of this recent trend:

- SAT: use propositional logic as the formalization language
  - \+ high degree of efficiency
  - \- expressive (all NP-complete) but not natural encodings
- SMT: propositional logic + domain-specific reasoning
  - \+ improves the expressivity
  - \- certain (but acceptable) loss of efficiency

> GOAL OF THIS COURSE:
> study techniques, tools and applications of SAT/SMT

UPC
Departament de Llenguatges i Sistemes Informàtics
UNIVERSITAT POLITÈCNICA DE CATALUNYA

Overview of the session

- Motivation
- SMT
- Theories of Interest
- Eager approach
- Lazy approach
  - Optimizations
  - Theory propagation
  - DPLL(T) in depth

Departament de Llenguatges i Sistemes Informàtics
UNIVERSITAT POLITÈCNICA DE CATALUNYA

Need and Applications of SMT

- Some problems are more naturally expressed in other logics than propositional logic, e.g:
  - Software verification needs reasoning about equality, arithmetic, data structures, ...
- SMT consists of deciding the satisfiability of a (ground) FO formula with respect to a background theory
- Example (Equality with Uninterpreted Functions – EUF):

$$g(a)=c \ \wedge \ (f(g(a)) \neq f(c) \vee g(a)=d) \ \wedge \ c \neq d$$

- Wide range of applications:
  - Predicate abstraction
  - Model checking
  - Equivalence checking
  - Static analysis
  - Scheduling
  - Test-case generation
  - ...

Overview of the session

- Motivation
- SMT
- Theories of Interest
- Eager approach
- Lazy approach
  - Optimizations
  - Theory propagation
  - DPLL(T) in depth

Departament de Llenguatges i Sistemes Informàtics
UNIVERSITAT POLITÈCNICA DE CATALUNYA

Theories of Interest - EUF

- Equality with Uninterpreted Functions, i.e. "$=$" is equality
- If background logic is FO with equality, EUF is empty theory
- Consider formula

$$a * (f(b) + f(c)) = d \quad \wedge \quad b * (f(a) + f(c)) \neq d \quad \wedge \quad a = b$$

- Formula is UNSAT, but no arithmetic resoning is needed
- If we abstract the formula into
$$h(a,\ g(f(b), f(c))\ ) = d \quad \wedge \quad h(b,\ g(f(a), f(c))\ ) \neq d \quad \wedge \quad a = b$$
it is still UNSAT
- EUF is used to to abstract non-supported constructions
  - Non-linear multiplication
  - ALUs in circuits

UPC
Departament de Llenguatges i Sistemes Informàtics
UNIVERSITAT POLITÈCNICA DE CATALUNYA

Theories of Interest - Arithmetic

- Very useful for obvious reasons
- Restricted fragments support more efficient methods:
  - Bounds: $x \bowtie k$ with $\bowtie \in \{<,>,\leq,\geq\}$
  - Difference logic: $x - y \bowtie k$, with $\bowtie \in \{<,>,\leq,\geq\}$
  - UTVPI: $x \pm y \bowtie k$, with $\bowtie \in \{<,>,\leq,\geq\}$
  - Linear arithmetic, e.g: $2x - 3y + 4z \leq 5$
  - Non-linear arithmetic, e.g: $2xy + 4xz^2 - 5y \leq 10$
  - Variables are either reals or integers

Theories of Interest - Arrays

- Two interpreted function symbols *read* and *write*
- Theory is axiomatized by:
  - $\forall a \forall i \forall v \; (read(write(a,i,v),i) = v)$
  - $\forall a \forall i \forall j \forall v \; (i \neq j \rightarrow read(write(a,i,v),i) = read(a,j))$
- Sometimes extensionality is added:
  - $\forall a \forall b \; (\, (\forall i (read(a,i) = read(b,i)) \,) \rightarrow a = b$
- Is the following set of literals satisfiable?

$$write(a,i,x) \neq b \quad read(b,i) = y \quad read(write(b,i,x),j) = y$$
$$a = b \qquad\qquad i = j$$

- Used for:
  - Software verification
  - Hardware verification (memories)

UPC
Departament de Llenguatges i Sistemes Informatics
UNIVERSITAT POLITECNICA DE CATALUNYA

Theories of Interest - Fixed-width bit vectors

- Constants represent vectors of bits
- Useful both for hardware and software verification
- Different type of operations:
  - String-like operations: concat, extract, ...
  - Logical operations: bit-wise not, or, and, ...
  - Arithmetic operations: add, substract, multiply, ...
- Assume bit-vectors have size 3. Is the formula SAT?

$$a[0:1] \neq b[0:1] \;\wedge\; (a|b) = c \;\wedge\; c[0] = 0 \;\wedge\; a[1] + b[1] = 0$$

UPC
Departament de Llenguatges i Sistemes Informatics
UNIVERSITAT POLITECNICA DE CATALUNYA

Theories of Interest - Combinations

- In practice, theories are not isolated
- Software verifications needs arithmetic, arrays, bitvectors, ...
- Formulas of the following form usually arise:

$$a = b + 2 \land A = write(B, a+1, 4) \land (read(A, b+3) = 2 \lor f(a-1) \neq f(b+1))$$

- The goal is to combine decision procedures for each theory

UPC
Departament de Llenguatges i Sistemes Informatics
UNIVERSITAT POLITECNICA DE CATALUNYA

Overview of the session

- Motivation
- SMT
- Theories of Interest
- Eager approach
- Lazy approach
  - Optimizations
  - Theory propagation
  - DPLL(T) in depth

Eager approach

- Methodology: translate problem into equisatisfiable propositional formula and use off-the-shelf SAT solver [Bryant, Velev, Pnueli, Lahiri, Seshia, Strichman, ...]
- Why “eager”?
  Search uses all theory information from the beginning
- Characteristics:
  - \+ Can use best available SAT solver
  - \- Sophisticated encodings are needed for each theory
- Tools: UCLID [Lahiri, Seshia and Bryant]

Overview of the session

- Motivation
- SMT
- Theories of Interest
- Eager approach
- Lazy approach
  - Optimizations
  - Theory propagation
  - DPLL(T) in depth

Departament de Llenguatges i Sistemes Informàtics
UNIVERSITAT POLITÈCNICA DE CATALUNYA

Lazy approach

Methodology:

Example: consider EUF and

$$\underbrace{g(a)=c}_{1} \;\wedge\; (\underbrace{f(g(a))\neq f(c)}_{\overline{2}} \vee \underbrace{g(a)=d}_{3}) \;\wedge\; \underbrace{c\neq d}_{\overline{4}}$$

- SAT solver returns model $[1, \overline{2}, \overline{4}]$

UPC
Departament de Llenguatges i Sistemes Informatics
UNIVERSITAT POLITECNICA DE CATALUNYA

Lazy approach

Methodology:

Example: consider EUF and

$$\underbrace{g(a)=c}_{1} \;\wedge\; (\underbrace{f(g(a))\neq f(c)}_{\overline{2}} \vee \underbrace{g(a)=d}_{3}) \;\wedge\; \underbrace{c\neq d}_{\overline{4}}$$

- SAT solver returns model $[1, \overline{2}, \overline{4}]$
- Theory solver says T-inconsistent

UPC
Departament de Llenguatges i Sistemes Informatics
UNIVERSITAT POLITECNICA DE CATALUNYA

Lazy approach

Methodology:

Example: consider EUF and

$$\underbrace{g(a)=c}_{1} \;\wedge\; (\underbrace{f(g(a))\neq f(c)}_{\overline{2}} \vee \underbrace{g(a)=d}_{3}) \;\wedge\; \underbrace{c\neq d}_{\overline{4}}$$

- SAT solver returns model $[1, \overline{2}, \overline{4}]$
- Theory solver says T-inconsistent
- Send $\{1, \overline{2} \vee 3, \overline{4}, \overline{1} \vee 2 \vee 4\}$ to SAT solver

UPC
Departament de Llenguatges i Sistemes Informatics
UNIVERSITAT POLITECNICA DE CATALUNYA

Lazy approach

Methodology:

Example: consider EUF and

$$\underbrace{g(a)=c}_{1} \;\wedge\; (\underbrace{f(g(a))\neq f(c)}_{\overline{2}} \vee \underbrace{g(a)=d}_{3}) \;\wedge\; \underbrace{c\neq d}_{\overline{4}}$$

- SAT solver returns model $[1, \overline{2}, \overline{4}]$
- Theory solver says T-inconsistent
- Send $\{1, \overline{2}\vee 3, \overline{4}, \overline{1}\vee 2\vee 4\}$ to SAT solver
- SAT solver returns model $[1, 2, 3, \overline{4}]$

UPC
Departament de Llenguatges i Sistemes Informatics
UNIVERSITAT POLITECNICA DE CATALUNYA

Lazy approach

Methodology:

Example: consider EUF and

$$\underbrace{g(a)=c}_{1} \wedge (\underbrace{f(g(a))\neq f(c)}_{\overline{2}} \vee \underbrace{g(a)=d}_{3}) \wedge \underbrace{c\neq d}_{\overline{4}}$$

- SAT solver returns model $[1, \overline{2}, \overline{4}]$
- Theory solver says T-inconsistent
- Send $\{1, \overline{2} \vee 3, \overline{4}, \overline{1} \vee 2 \vee 4\}$ to SAT solver
- SAT solver returns model $[1, 2, 3, \overline{4}]$
- Theory solver says T-inconsistent

UPC
Departament de Llenguatges i Sistemes Informatics
UNIVERSITAT POLITECNICA DE CATALUNYA

Lazy approach

Methodology:

Example: consider EUF and

$$\underbrace{g(a)=c}_{1} \wedge \big(\underbrace{f(g(a))\neq f(c)}_{\overline{2}} \vee \underbrace{g(a)=d}_{3}\big) \wedge \underbrace{c\neq d}_{\overline{4}}$$

- SAT solver returns model $[1, \overline{2}, \overline{4}]$
- Theory solver says T-inconsistent
- Send $\{1, \overline{2} \vee 3, \overline{4}, \overline{1} \vee 2 \vee 4\}$ to SAT solver
- SAT solver returns model $[1, 2, 3, \overline{4}]$
- Theory solver says T-inconsistent
- SAT solver detects $\{1, \overline{2} \vee 3, \overline{4}, \overline{1} \vee 2 \vee 4, \overline{1} \vee \overline{2} \vee \overline{3} \vee 4\}$
  UNSATISFIABLE

Departament de Llenguatges i Sistemes Informatics
UNIVERSITAT POLITECNICA DE CATALUNYA

Lazy approach (2)

- Why "lazy"?
  Theory information used lazily when checking T-consistency of propositional models

- Characteristics:
  - \+ Modular and flexible
  - \- Theory information does not guide the search

- Tools:
  - Barcelogic (UPC)
  - CVC3 (Univ. New York + Iowa)
  - DPT (Intel)
  - MathSAT (Univ. Trento)
  - Yices (SRI)
  - Z3 (Microsoft)
  - ...

Lazy approach - Optimizations

Several optimizations for enhancing efficiency:

- Check T-consistency only of full propositional models

Lazy approach - Optimizations

Several optimizations for enhancing efficiency:

- ~~Check T-consistency only of full propositional models~~
- Check T-consistency of partial assignment while being built

Departament de Llenguatges i Sistemes Informatics
UNIVERSITAT POLITECNICA DE CATALUNYA

Lazy approach - Optimizations

Several optimizations for enhancing efficiency:

- ~~Check T-consistency only of full propositional models~~
- Check T-consistency of partial assignment while being built

- Given a T-inconsistent assignment M, add $\neg M$ as a clause

Departament de Llenguatges i Sistemes Informatics
UNIVERSITAT POLITECNICA DE CATALUNYA

Lazy approach - Optimizations

Several optimizations for enhancing efficiency:

- ~~Check T-consistency only of full propositional models~~
- Check T-consistency of partial assignment while being built

- ~~Given a T-inconsistent assignment M, add $\neg M$ as a clause~~
- Given a T-inconsistent assignment M, identify a T-inconsistent subset $M_0 \subseteq M$ and add $\neg M_0$ as a clause

UPC
Departament de Llenguatges i Sistemes Informatics
UNIVERSITAT POLITECNICA DE CATALUNYA

Lazy approach - Optimizations

Several optimizations for enhancing efficiency:

- ~~Check T-consistency only of full propositional models~~
- Check T-consistency of partial assignment while being built

- ~~Given a T-inconsistent assignment M, add $\neg M$ as a clause~~
- Given a T-inconsistent assignment M, identify a T-inconsistent subset $M_0 \subseteq M$ and add $\neg M_0$ as a clause

- Upon a T-inconsistency, add clause and restart

UPC
Departament de Llenguatges i Sistemes Informàtics
UNIVERSITAT POLITÈCNICA DE CATALUNYA

Lazy approach - Optimizations

Several optimizations for enhancing efficiency:

- ~~Check T-consistency only of full propositional models~~
- Check T-consistency of partial assignment while being built

- ~~Given a T-inconsistent assignment M, add $\neg M$ as a clause~~
- Given a T-inconsistent assignment M, identify a T-inconsistent subset $M_0 \subseteq M$ and add $\neg M_0$ as a clause

- ~~Upon a T-inconsistency, add clause and restart~~
- Upon a T-inconsistency, bactrack to some point where the assignment was still T-consistent

Departament de Llenguatges i Sistemes Informatics
UNIVERSITAT POLITECNICA DE CATALUNYA

Lazy approach - Important points

Important and benefitial aspects of the lazy approach:
(even with the optimizations)

- Everyone does what he/she is good at:
  - SAT solver takes care of Boolean information
  - Theory solver takes care of theory information
- Theory solver only receives conjunctions of literals
- Modular approach:
  - SAT solver and T-solver communicate via a simple API
  - SMT for a new theory only requires new T-solver
  - SAT solver can be embedded in a lazy SMT system with very few new lines of code (40?)

Lazy approach - T-propagation

- As pointed out the lazy approach has one drawback:
  - Theory information does not guide the search
- How can we improve that?

  T-Propagate :

$$M \parallel F \quad \Rightarrow \quad M\, l \parallel F \quad \textbf{if} \begin{cases} M \models_T l \\ l \text{ or } \neg l \text{ occurs in } F \text{ and not in } M \end{cases}$$

- Search guided by T-Solver by finding T-consequences, instead of only validating it as in basic lazy approach.
- Naive implementation:: Add $\neg l$. If T-inconsistent then infer l. But for efficient Theory Propagation we need:

  -T-Solvers specialized and fast in it.
  -fully exploited in conflict analysis
- This approach has been named DPLL(T)

UPC
Departament de Llenguatges i Sistemes Informàtics
UNIVERSITAT POLITÈCNICA DE CATALUNYA

DPLL(T) - Example

Consider again EUF and the formula:

$$\underbrace{g(a)=c}_{1} \;\wedge\; (\underbrace{f(g(a))\neq f(c)}_{\overline{2}} \vee \underbrace{g(a)=d}_{3}) \;\wedge\; \underbrace{c\neq d}_{\overline{4}}$$

$$\emptyset \;\|\; 1,\; \overline{2}\vee 3,\; \overline{4} \quad\Rightarrow\quad (\text{UnitPropagate})$$

UPC
Departament de Llenguatges i Sistemes Informatics
UNIVERSITAT POLITECNICA DE CATALUNYA

DPLL(T) - Example

Consider again EUF and the formula:

$$\underbrace{g(a)=c}_{1} \;\wedge\; (\underbrace{f(g(a))\neq f(c)}_{\overline{2}} \vee \underbrace{g(a)=d}_{3}) \;\wedge\; \underbrace{c\neq d}_{\overline{4}}$$

$$\emptyset \;\|\; 1,\; \overline{2}\vee 3,\; \overline{4} \quad\Rightarrow\quad (\text{UnitPropagate})$$

$$1 \;\|\; 1,\; \overline{2}\vee 3,\; \overline{4} \quad\Rightarrow\quad (\text{T-Propagate})$$

UPC
Departament de Llenguatges i Sistemes Informatics
UNIVERSITAT POLITECNICA DE CATALUNYA

DPLL(T) - Example

Consider again EUF and the formula:

$$\underbrace{g(a)=c}_{1} \;\land\; (\underbrace{f(g(a))\neq f(c)}_{\overline{2}} \lor \underbrace{g(a)=d}_{3}) \;\land\; \underbrace{c\neq d}_{\overline{4}}$$

$$\emptyset \;\|\; 1,\; \overline{2}\lor 3,\; \overline{4} \quad\Rightarrow\quad (\text{UnitPropagate})$$

$$1 \;\|\; 1,\; \overline{2}\lor 3,\; \overline{4} \quad\Rightarrow\quad (\text{T-Propagate})$$

$$1\,2 \;\|\; 1,\; \overline{2}\lor 3,\; \overline{4} \quad\Rightarrow\quad (\text{UnitPropagate})$$

UPC
Departament de Llenguatges i Sistemes Informatics
UNIVERSITAT POLITECNICA DE CATALUNYA

DPLL(T) - Example

Consider again EUF and the formula:

$$\underbrace{g(a)=c}_{1} \;\wedge\; (\underbrace{f(g(a))\neq f(c)}_{\overline{2}} \vee \underbrace{g(a)=d}_{3}) \;\wedge\; \underbrace{c\neq d}_{\overline{4}}$$

$$\begin{array}{rcl} \emptyset \parallel 1,\ \overline{2}\vee 3,\ \overline{4} & \Rightarrow & (\text{UnitPropagate}) \\ 1 \parallel 1,\ \overline{2}\vee 3,\ \overline{4} & \Rightarrow & (\text{T-Propagate}) \\ 1\ \overline{2} \parallel 1,\ \overline{2}\vee 3,\ \overline{4} & \Rightarrow & (\text{UnitPropagate}) \\ 1\ \overline{2}\ 3 \parallel 1,\ \overline{2}\vee 3,\ \overline{4} & \Rightarrow & (\text{T-Propagate}) \end{array}$$

UPC
Departament de Llenguatges i Sistemes Informatics
UNIVERSITAT POLITECNICA DE CATALUNYA

DPLL(T) - Example

Consider again EUF and the formula:

$$\underbrace{g(a)=c}_{1} \;\wedge\; (\,\underbrace{f(g(a))\neq f(c)}_{\overline{2}} \vee \underbrace{g(a)=d}_{3}\,) \;\wedge\; \underbrace{c\neq d}_{\overline{4}}$$

$$\emptyset \parallel 1,\ \overline{2}\vee 3,\ \overline{4} \quad\Rightarrow\quad (\text{UnitPropagate})$$

$$1 \parallel 1,\ \overline{2}\vee 3,\ \overline{4} \quad\Rightarrow\quad (\text{T-Propagate})$$

$$1\,2 \parallel 1,\ \overline{2}\vee 3,\ \overline{4} \quad\Rightarrow\quad (\text{UnitPropagate})$$

$$1\,2\,3 \parallel 1,\ \overline{2}\vee 3,\ \overline{4} \quad\Rightarrow\quad (\text{T-Propagate})$$

$$1\,2\,3\,4 \parallel 1,\ \overline{2}\vee 3,\ \overline{4} \quad\Rightarrow\quad (\text{Fail})$$

DPLL(T) - Example

Consider again EUF and the formula:

$$\underbrace{g(a)=c}_{1} \;\wedge\; (\underbrace{f(g(a))\neq f(c)}_{\overline{2}} \vee \underbrace{g(a)=d}_{3}) \;\wedge\; \underbrace{c\neq d}_{\overline{4}}$$

$$\begin{array}{rcl} \emptyset \parallel 1,\ \overline{2}\vee 3,\ \overline{4} & \Rightarrow & (\text{UnitPropagate}) \\ 1 \parallel 1,\ \overline{2}\vee 3,\ \overline{4} & \Rightarrow & (\text{T-Propagate}) \\ 1\,2 \parallel 1,\ \overline{2}\vee 3,\ \overline{4} & \Rightarrow & (\text{UnitPropagate}) \\ 1\,2\,3 \parallel 1,\ \overline{2}\vee 3,\ \overline{4} & \Rightarrow & (\text{T-Propagate}) \\ 1\,2\,3\,4 \parallel 1,\ \overline{2}\vee 3,\ \overline{4} & \Rightarrow & (\text{Fail}) \\ \mathit{fail} & & \end{array}$$

UPC

Departament de Llenguatges i Sistemes Informatics

UNIVERSITAT POLITECNICA DE CATALUNYA

DPLL(T) - Overall algorithm

High-levew view gives the same algorithm as a CDCL SAT solver:

```
while(true){

    while (propagate_gives_conflict()){
        if (decision_level==0) return UNSAT;
        else analyze_conflict();
    }

    restart_if_applicable();
    remove_lemmas_if_applicable();

    if (!decide()) returns SAT; // All vars assigned
}
```

Differences are in:

- `propagate_gives_conflict`
- `analyze_conflict`

UPC
Departament de Llenguatges i Sistemes Informatics
UNIVERSITAT POLITECNICA DE CATALUNYA

DPLL(T) - Propagation

```
propagate_gives_conflict( ) returns Bool

    do {

      // unit propagate
      if ( unit_prop_gives_conflict() ) then return  false

      // check T-consistency of the model
      if ( solver.is_model_inconsistent() ) then return  false

      // theory propagate
      solver.theory_propagate()

    } while  (someTheoryPropagation)
```

DPLL(T) - Propagation (2)

- Three operations:
  - Unit propagation (SAT solver)
  - Consistency checks (T-solver)
  - Theory propagation (T-solver)
- Cheap operations are computed first
- If theory is expensive, calls to T-solver are sometimes skipped
- For completeness, only necessary to call T-solver at the leaves
  (i.e. when we have a full propositional model)
- Theory propagation is not necessary for completeness

DPLL(T) - Conflict Analysis

Remember conflict analysis in SAT solvers:

C`:= conflicting clause`

while C `contains more than one lit of last DL`

    l`:=last literal assigned in` C
    C`:=Resolution(`C`,reason(`l`))`

end while

```
// let C = C' v l where l is UIP
backjump(maxDL(C'))
add l to the model with reason C
learn(C)
```

UPC
Departament de Llenguatges i Sistemes Informàtics
UNIVERSITAT POLITÈCNICA DE CATALUNYA

DPLL(T) - Conflict Analysis (2)

Conflict analysis in DPLL(T):

if `boolean conflict` **then** C`:= conflicting clause`
else C`:=`$\neg$`( solver.explain_inconsistency() )`

while C `contains more than one lit of last DL`

    l`:=last literal assigned in` C
    C`:=Resolution(`C`,reason(`l`))`

end while

```
// let C = C′ v l where l is UIP
backjump(maxDL(C′))
add l to the model with reason C
learn(C)
```

UPC
Departament de Llenguatges i Sistemes Informàtics
UNIVERSITAT POLITÈCNICA DE CATALUNYA

DPLL(T) - Conflict Analysis (3)

What does `explain_inconsistency` return?

- A (small) conjuntion of literals $l_1 \wedge \ldots \wedge l_n$ such that:
  - They were in the model when T-inconsistency was found
  - It is T-inconsistent

What is now $reason(l)$?

- If l was unit propagated $\longrightarrow$ clause that propagated it
- If l was T-propagated?
  - T-solver has to provide an explanation for l, i.e. a (small) set of literals $l_1, \ldots, l_n$ such that:
    - They were in the model when l was T-propagated
    - $l_1 \wedge \ldots \wedge l_n \models_T l$
  - Then $reason(l)$ is $\neg l_1 \vee \ldots \vee \neg l_n \vee l$

UPC
Departament de Llenguatges i Sistemes Informàtics
UNIVERSITAT POLITÈCNICA DE CATALUNYA

DPLL(T) - Conflict Analysis (4)

Let M be of the form $N, c = b, f(a) \neq f(b)$ and let F contain

$$a = b \ \vee \ g(a) \neq g(b), \qquad h(a) = h(c) \ \vee \ p, \qquad g(a) = g(b) \ \vee \ \neg p$$

Take the following sequence:

1. Decide $h(a) \neq h(c)$
2. T-Propagate $a \neq b$ (due to $h(a) \neq h(c)$ and $c = b$)
3. UnitPropagate $g(a) \neq g(b)$
4. UnitPropagate p
5. Conflicting clause $g(a) = g(b) \ \vee \ \neg p$

Explain($a \neq b$) is $\{h(a) \neq h(c), c = b\}$

$$\dfrac{h(a) = h(c) \vee c \neq b \vee a \neq b \qquad \dfrac{a = b \vee g(a) \neq g(b) \qquad \dfrac{h(a) = h(c) \vee p \qquad g(a) = g(b) \vee \neg p}{h(a) = h(c) \vee g(a) = g(b)}}{h(a) = h(c) \vee a = b}}{h(a) = h(c) \ \vee \ c \neq b}$$

Departament de Llenguatges i Sistemes Informàtics
UNIVERSITAT POLITÈCNICA DE CATALUNYA

DPLL(T) - Some final remarks

- Completing a partial model is no longer a trivial task
  (no proper solution found so far)
- What about T-based decision heuristics?
  (no successful alternative found so far)
- What about producing proofs? 3 things to check
  - Explanations of inconsistencies are T-tautologies
  - Explanations of T-propagations are T-tautologies
  - Resolution propositional proof is correct
- See `http://www.smtlib.org` for benchmarks, theories, ...
- See `http://www.smtcomp.org` for competition results

Departament de Llenguatges i Sistemes Informatics
UNIVERSITAT POLITECNICA DE CATALUNYA