A Survey of Stålmarck's Method

Honours project
Daniel Högberg
Supervisor: Martin Henz

Stålmarck's method outlined

- A method for solving the satisfiability problem of a propositional formula F.
- As bad as any other method in the worst case but very efficient for real applications.
- Stålmarck's original patent from -94 with just implications.

Stålmarck's method outlined

- Rewrite the input formula F into a logically equivalent formula F' with a possibly less number of connectives.
- Represent the formula as triplets.
- Propagate the triplets.
- If terminal then we have proved tautology, else we make assumptions and repeat the propagation.

What the fuzz is about

- Many people have many different suggestions how to build a proof system from the method.
- Often the solutions are in an ad hoc manner.
- Theoretical reasoning and benchmarks to show properties of the method.

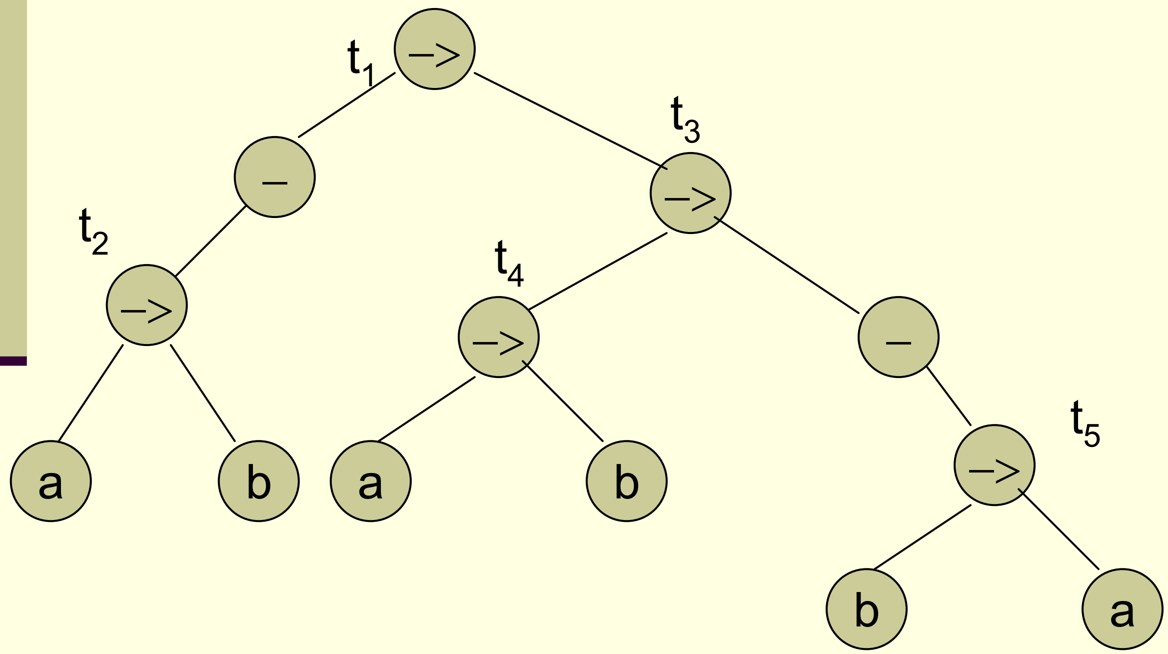
Triplets and propagation rules

- The input formula is first rewritten into a logically equivalent one.
- Example: Stålmarck's patent with just implications and negations.

$$a \wedge \neg b \rightarrow \neg(a \leftrightarrow b) = \neg(a \rightarrow b) \rightarrow ((a \rightarrow b) \rightarrow \neg(b \rightarrow a))$$

Triplets and propagation rules

- The formula F' is the represented as a set of triplets over the chosen connectives.
- Example: Stålmarck's patent again



$$t_1 = \neg t_2 \rightarrow t_3$$

$$t_2 = a \rightarrow b$$

$$t_3 = t_4 \rightarrow \neg t_5$$

$$t_4 = a \rightarrow b$$

$$t_5 = b \rightarrow a$$

Triplets and propagation rules

- This gives a number of propagation rules by hand.
- These rules can be applied to triplets and propagate values to uninstantiated variables.
- This is done repeatedly until no more rules can be applied or a terminal is found.

Triplets and propagation rules

- Example: Stålmarck's patent again
- Each rule propagates values to maximum two variables
- The rules for contradiction are not written out

$$1: \frac{p \leftrightarrow (q \to 1)}{1 \leftrightarrow (q \to 1)} \, p/1$$

$$2: \frac{p \leftrightarrow (0 \to r)}{1 \leftrightarrow (0 \to r)} \, p/1$$

$$3: \frac{p \leftrightarrow (q \to q)}{1 \leftrightarrow (q \to q)} \, p/1$$

$$4: \frac{p \leftrightarrow (p \to r)}{1 \leftrightarrow (1 \to 1)} \, p/1, r/1$$

$$5: \frac{p \leftrightarrow (1 \to r)}{r \leftrightarrow (1 \to r)} \, p/r$$

$$6: \frac{p \leftrightarrow (q \to 0)}{\neg p \leftrightarrow (q \to 0)} \, p/\neg q$$

$$7: \frac{0 \leftrightarrow (q \to r)}{0 \leftrightarrow (1 \to 0)} \, q/1, r/0$$

0-Saturation

- In the 0-saturation we try to find a terminal by repeatedly applying rules.
- We start with negating the whole formula and then continuing with propagation wherever possible.
- Example: Stålmarck's patent again

$$a \wedge \neg b \rightarrow \neg(a \leftrightarrow b) = \neg(a \rightarrow b) \rightarrow ((a \rightarrow b) \rightarrow \neg(b \rightarrow a))$$

0-Saturation

$$t_1 = \neg t_2 \rightarrow t_3$$
$$t_2 = a \rightarrow b$$
$$t_3 = t_4 \rightarrow \neg t_5$$
$$t_4 = a \rightarrow b$$
$$t_5 = b \rightarrow a$$

$\longrightarrow$ $0 = \neg t_2 \rightarrow t_3$

$t_2 = a \rightarrow b$

$t_3 = t_4 \rightarrow \neg t_5$

$t_4 = a \rightarrow b$

$t_5 = b \rightarrow a$

$\longrightarrow$ $0 = \neg 0 \rightarrow 0$

$\longrightarrow$ $0 = a \rightarrow b$

$\longrightarrow$ $0 = t_4 \rightarrow \neg t_5$

$t_4 = a \rightarrow b$

$t_5 = b \rightarrow a$

$0 = \neg 0 \rightarrow 0$

$\longrightarrow$ $0 = 1 \rightarrow 0$

$0 = t_4 \rightarrow \neg t_5$

$\longrightarrow$ $t_4 = 1 \rightarrow 0$

$\longrightarrow$ $t_5 = 0 \rightarrow 1$

0-Saturation

$0 = \neg 0 \rightarrow 0$

$0 = 1 \rightarrow 0$

→ $0 = 1 \rightarrow \neg 1$

→ $1 = 1 \rightarrow 0$

→ $1 = 0 \rightarrow 1$

$0 = \neg 0 \rightarrow 0$

$0 = 1 \rightarrow 0$

$0 = 1 \rightarrow \neg 1$

$1 = 1 \rightarrow 0$

$1 = 0 \rightarrow 1$

(k+1)-Saturation

- When we end up with uninstantiated variables we have to make an assumption.
- Two derivations are performed in parallel with the assumption and its complement.

$$\frac{R}{R(A \equiv B) \quad R(A \equiv B')}$$

$$derivation \quad derivation$$

$$\frac{R_1 \qquad R_2}{R_1 \cap R_2}$$

(k+1)-Saturation

- The intersection of the two relations givs new information about the formula.
- If both of them holds a terminal we are done, if either holds a terminal we continue with the other one.
- Example:

$$(((p \to p) \to p) \to (p \to q)) \to (((p \to q) \to p) \to q)$$

(k+1)-Saturation

$t_1 = t_2 \to t_6$	$\longrightarrow$ $0 = t_2 \to t_6$	$\longrightarrow$ $0 = 1 \to 0$
$t_2 = t_3 \to t_5$	$t_2 = t_3 \to t_5$	$\longrightarrow$ $1 = t_3 \to t_5$
$t_3 = t_4 \to p$	$t_3 = t_4 \to p$	$t_3 = t_4 \to p$
$t_4 = p \to p$	$t_4 = p \to p$	$t_4 = p \to p$
$t_5 = p \to q$	$t_5 = p \to q$	$t_5 = p \to q$
$t_6 = t_7 \to q$	$t_6 = t_7 \to q$	$\longrightarrow$ $0 = t_7 \to q$
$t_7 = t_8 \to p$	$t_7 = t_8 \to p$	$t_7 = t_8 \to p$
$t_8 = p \to q$	$t_8 = p \to q$	$t_8 = p \to q$

(k+1)-Saturation

	$0=1\to 0$		$0=1\to 0$		$0=1\to 0$
	$1=t_3\to t_5$		$1=t_3\to t_5$	$\longrightarrow$	$1=p\to t_5$
	$t_3=t_4\to p$	$\longrightarrow$	$t_3=1\to p$	$\longrightarrow$	$p=1\to p$
	$t_4=p\to p$	$\longrightarrow$	$1=p\to p$		$1=p\to p$
$\longrightarrow$	$t_5=p\to 0$		$t_5=p\to 0$		$t_5=p\to 0$
$\longrightarrow$	$0=1\to 0$		$0=1\to 0$		$0=1\to 0$
$\longrightarrow$	$1=t_8\to p$		$1=t_8\to p$		$1=t_8\to p$
$\longrightarrow$	$t_8=p\to 0$		$t_8=p\to 0$		$t_8=p\to 0$

(k+1)-Saturation

$0 = 1 \rightarrow 0$	$0 = 1 \rightarrow 0$	$0 = 1 \rightarrow 0$
$1 = p \rightarrow t_5$	$\longrightarrow$ $1 = p \rightarrow \neg p$	$1 = p \rightarrow \neg p$
$p = 1 \rightarrow p$	$p = 1 \rightarrow p$	$p = 1 \rightarrow p$
$1 = p \rightarrow p$	$1 = p \rightarrow p$	$1 = p \rightarrow p$
$t_5 = p \rightarrow 0$	$\longrightarrow$ $\neg p = p \rightarrow 0$	$\neg p = p \rightarrow 0$
$0 = 1 \rightarrow 0$	$0 = 1 \rightarrow 0$	$0 = 1 \rightarrow 0$
$\longrightarrow$ $1 = \neg p \rightarrow p$	$1 = \neg p \rightarrow p$	$1 = \neg p \rightarrow p$
$\longrightarrow$ $\neg p = p \rightarrow 0$	$\neg p = p \rightarrow 0$	$\neg p = p \rightarrow 0$

Equivalence classes

- We can use equivalence classes to describe this propagation of values to variables.
- Two variabels belongs to the same equivalence class, denoted a~b, iff they have the same truth value.
- A propagation rule merges two equivalence classes.
- It is enough to represent one class C without its complement class C’.

Equivalence classes

- Initially all variables belongs to their own equivalence class, these are called indeterminate classes.
- Except these indeterminate classes we also have the True-class.
- Example:

$$a \wedge \neg b \rightarrow \neg(a \leftrightarrow b) = \neg(a \rightarrow b) \rightarrow ((a \rightarrow b) \rightarrow \neg(b \rightarrow a))$$

Equivalence classes

$$\{1\} \quad \{t_1\} \quad \{t_2\} \quad \{t_3\} \quad \{t_4\} \quad \{t_5\} \quad \{a\} \quad \{b\}$$

$$\{1 \quad t_1'\} \quad \{t_2\} \quad \{t_3\} \quad \{t_4\} \quad \{t_5\} \quad \{a\} \quad \{b\}$$

$$\{1 \quad t_1' \quad t_2'\} \quad \{t_3\} \quad \{t_4\} \quad \{t_5\} \quad \{a\} \quad \{b\}$$

$$\{1 \quad t_1' \quad t_2' \quad t_3'\} \quad \{t_4\} \quad \{t_5\} \quad \{a\} \quad \{b\}$$

$$\{1 \quad t_1' \quad t_2' \quad t_3' \quad a\} \quad \{t_5\} \quad \{t_4\} \quad \{b\}$$

$$\{1 \quad t_1' \quad t_2' \quad t_3' \quad a \quad b'\} \quad \{t_4\} \quad \{t_5\}$$

$$\{1 \quad t_1' \quad t_2' \quad t_3' \quad a \quad b' \quad t_5\} \quad \{t_4\}$$

$$\{1 \quad t_1' \quad t_2' \quad t_3' \quad a \quad b' \quad t_5 \quad t_4\}$$

$$\{1 \quad t_1' \quad t_2' \quad t_3' \quad a \quad b' \quad t_5 \quad t_4 \quad t_4'\}$$

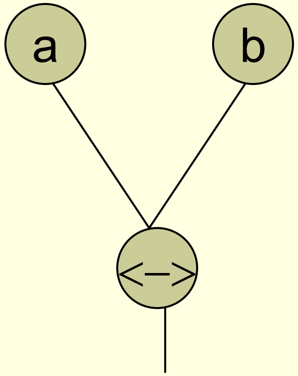
Properties and improvements

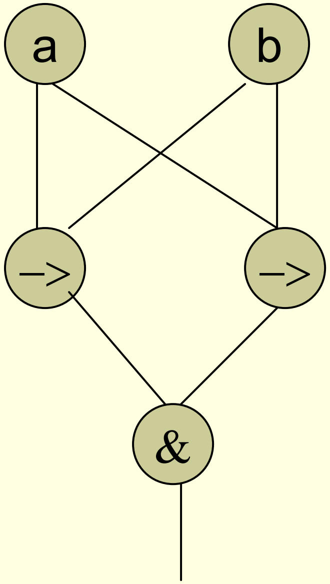
- Initially we rewrite the formula F into a logically equivalent formula F’ to minimize the number of propagation rules.
  - Even if they are logically equivalent it does not mean that they preserves equivalence class properties.

Properties and improvements

- Example: Assume a~b’

$$a \leftrightarrow b \qquad (a \rightarrow b) \wedge (b \rightarrow a)$$





Properties and improvements

- Instead of just propagating on a single triplet we could use pairs of triplets to derive new information.
  - Example: Assume a~c and b~d over the same connective, and consider the following triplets. We can derive that t_1~t_2.

$$t_1 = a \oplus b$$

$$t_2 = c \oplus d$$

Properties and improvements

- There are a number of strategies of how to improve the saturation.
- Branching is the expensive operation so we would like to branch over a suitable variable and branch with a suitable value.
  - In Stålmarck’s patent he branches with truth-falsity.
  - But it maybe is a good idea to branch with arbitrary variables. E.g. the variable that belongs to the largest equivalence class, we tries to merge the largest classes.
  - If we have a choice of which variable to branch over, we might consider to branch over a variable that is involved in the largest number of triplets.

Properties and improvements

- Associativity of connectives is also a problem when representing the formula as set of triplets.
  - Example: Assume that a~c', we want to represent the associativity property of AND. We can do this by adding more triplets to the set of triplets.

$$a \wedge (b \wedge c) = (a \wedge b) \wedge c = (a \wedge b \wedge c) = a \wedge b \wedge c$$

$$t_1 = a \wedge t_2 \qquad t_1 = b \wedge t_2 \qquad t_1 = c \wedge t_2$$

$$t_2 = b \wedge c \qquad t_2 = a \wedge c \qquad t_2 = a \wedge b$$

Short about implementation

- The proof system involves a number of steps:
  - Parsing of the input formula F.
  - Rewriting the formula F into formula F' with a possibly reduced number of connectives.
  - Build a triplet representation.
  - Assume the formula false and try to find a contradiction.
  - Saturate until terminal found or we can not prove the tautology of the formula.

Short about implementation

- We use two important data structures:
  - Set of triplets
  - Equivalence class relation
- The triplets are only place holders for the actual values which are given by the equivalence class relation.
- The equivalence class relation holds information about which variables are related to which equivalence class.

Short about implementation

- The equivalence class relation is implemented as a disjoint set and must be efficient with respect to a number of operations:
  - eqClassOf
  - branchIntersect
  - Union
- For every variable we keep a list of triplet it is involved in.

Short about implementation

- A triplet is called triggable if it is subject to propagation. Triggable triplets are kept in a data structure we call pool, and preserves a number of invariants:
  - Either the triplet has one variable in the True/False-class or it has two variables in the same indeterminate class.
  - If a triplet has at most two distinct variables either it can propagate or is useless.
  - A triplet with only one unassigned variable is either a subject to propagation or is useless.
  - A triplet is propagated on at most two times.

Questions

- Questions?