CS510 Software Engineering

Propositional Logic

Asst. Prof. Mathias Payer

Department of Computer Science
Purdue University

TA: Scott A. Carr
Slides inspired by Xiangyu Zhang

http://nebelwelt.net/teaching/15-CS510-SE

Spring 2015

Additional slides credit: Michael Reniers, Julia Lawall, and Neil Jones.

Motivation

Many static analysis techniques rely on proofing that some set of conditions hold. We need to come up with a way to express these conditions and reason about them.
SAT solving allows to test the satisfiablity of propositional formulas in the domain of Boolean values.
SAT solving is used for, e.g., formal equivalence checking, model checking, formal verification, automatic test pattern generation, scheduling problems, and symbolic execution.
We need to understand propositional logic and SAT solving to follow the techniques listed above.

History of Logic

- Philosophical Logic (500BC to 19th century)
- Symbolic Logic (mid to late 19th century)
- Mathematical Logic (late 19th century to mid 20th century)
- Logic in Computer Science (now)

Table of Contents

Syntax

$$F :== (P)|(\neg F)|(F \vee F)|(F \wedge F)|(F \rightarrow F)$$

$$P :== p|q|r|...$$

Propositional atoms (p, q, r, ...) are used to describe declarative sentences like "1037 is a prime number", "Every even number > 2 is the sum of two prime numbers", or "All Martians like pepperoni on their pizza" (i.e., they can be evaluated to true or false).

Connective	Symbol	Alternative Symbols
negation (not)	$\neq$	
disjunction (or)	$\vee$	$\mid$
conjunction (and)	$\wedge$	&
mplication (implies)	$\rightarrow$	$\Rightarrow$, $\supset$, $\subseteq$

Syntax for propositional logic

Binding priorities: $\neg$, $\vee$, $\wedge$, $\rightarrow$, $\leftrightarrow$
(These help reduce the amount of brackets needed. Also, outermost brackets are often omitted.)

Table of Contents

Semantics for Propositional Logic

The *meaning* of a formula depends on:

- The meaning of the propositional atoms (occurring in the formula)
- The meaning of the connectives (occurring in the formula)

Semantics: Propositional Atoms

The meaning of the propositional atoms (occurring in the formula):

- A declarative sentence is either `true` or `false`
- Captured as an assignment of truth values ($\mathbb{B} = \{T, F\}$) to the propositional atoms a *valuation* $v : P \to \mathbb{B}$

Semantics: Connectives

- The meaning of an n-ary connective $\oplus$ is captured by a function $f_{\oplus} : \mathbb{B}^n \to \mathbb{B}$
- Usually, such functions are specified by a truth table.

A	B	$\neg$ A	A $\wedge$ B	A $\vee$ B	A $\to$ B
T	T	F	T	T	T
T	F	F	F	T	F
F	T	T	F	T	T
F	F	T	F	F	T

Example: Formula Evaluation

Evaluate the following formula: $(p \to q) \wedge (q \to r) \to (p \to r)$

p	q	r	$p \to q$	$q \to r$	$\ldots \wedge \ldots$	$p \to r$	$A \wedge B \to C$
T	T	T	T	T	T	T	T
T	T	F	T	F	F	F	T
T	F	T	F	T	F	T	T
T	F	F	F	T	F	F	T
F	T	T	T	T	T	T	T
F	T	F	T	F	F	T	T
F	F	T	T	T	T	T	T
F	F	F	T	T	T	T	T

Table of Contents

Areas of Interest

- *Semantic entailment.* Many logical arguments are of the form: from the assumptions $\phi_1, \phi_2, \ldots \phi_n$ we know ψ. This is formalised by the *semantic entailment* relation $\models$. E.g., $M \models A$ describes that a situation M satisfies a formula A.
  Formally, $\phi_1, \phi_2, \ldots \phi_n \models \psi$ iff for all valuations v such that $[\![\phi_i]\!](v) = T$ for all $1 \leq i \leq n$ we have $[\![\psi]\!](v) = T$
- Validity: a formula ϕ is valid if $\models \phi$ holds.
- Satisfiability: a formula ϕ is *sat* if there exists a valuation v so that $[\![\phi]\!](v) = T$.

Semantic Entailment

How do we establish semantic entailment $\phi_1, \phi_2, \ldots \phi_n \models \psi$?
Option 1: Construct a truth table.
If formulas contain m propositional atoms, the truth table contains 2^m lines!
Option 2: Give a proof.
Suppose that $(p \rightarrow q) \wedge (q \rightarrow r)$. Suppose that p. Then, as $p \rightarrow q$ follows from $(p \rightarrow q) \wedge (q \rightarrow r)$, we have q. Finally, as $q \rightarrow r$ follows from $(p \rightarrow q) \wedge (q \rightarrow r)$, we have r. Thus the formula holds (i.e., there is no contradiction).

Semantic Entailment

Proof rules for inferring a conclusion ψ from a list of premises ϕ_1, ϕ_2, $\ldots \phi_n$ ($x \vdash y$ means that y is provable from x):

$$\phi_1, \phi_2, \ldots \phi_n \vdash \psi(\mathit{sequent})$$

What is a proof of a sequent $\phi_1, \phi_2, \ldots \phi_n \vdash \psi$?

- Proof rules may be instantiated: consistent replacement of variables with formulas.
- Constructing the proof is filling the gap between the premises and the conclusion by applying a suitable sequence of proof rules.

Natural Deduction: Conjunction

Proof rules for *conjunction*: proofs of $\psi \wedge \phi$ are a concatenation of proofs for ψ and proofs of ϕ.

$\wedge$ introduction: $$\frac{\psi \qquad \phi}{\psi \wedge \phi} \wedge i$$

$\wedge$ elimination: $$\frac{\psi \wedge \phi}{\psi} \wedge e_1 \qquad \frac{\psi \wedge \phi}{\phi} \wedge e_2$$

Conjunction: Exercise

Prove $(p \wedge q) \wedge r,\ s \wedge t \vdash q \wedge s$. Given that we have $(p \wedge q) \wedge r$ and $s \wedge t$ we can prove $q \wedge s$.
Linear representation:

1	$(p \wedge q) \wedge r$	premise
2	$s \wedge t$	premise
3	$p \wedge q$	$\wedge e_1$ 1
4	q	$\wedge e_2$ 3
5	s	$\wedge e_1$ 2
6	$q \wedge s$	$\wedge i$ 4,5

Conjunction: Exercise (2)

Prove $(p \wedge q) \wedge r,\ s \wedge t \vdash q \wedge s$. Given that we have $(p \wedge q) \wedge r$ and $s \wedge t$ we can prove $q \wedge s$.
Tree representation:

$$\dfrac{\dfrac{\dfrac{(p \wedge q) \wedge r}{p \wedge q} \wedge e_1}{q} \wedge e_2 \quad \dfrac{\dfrac{s \wedge t}{s} \wedge e_1}{}}{q \wedge s} \wedge i$$

Natural Deduction: Disjunction

Proof rules for *disjunction*:

$\vee$ introduction: $\dfrac{\psi}{\psi \vee \phi} \vee i_1$ $\qquad$ $\dfrac{\phi}{\psi \vee \phi} \vee i_2$

$\vee$ elminiation: $\dfrac{\phi \vee \psi \qquad \boxed{\begin{array}{c}\phi \\ \vdots \\ \chi\end{array}} \qquad \boxed{\begin{array}{c}\psi \\ \vdots \\ \chi\end{array}}}{\chi} \vee e$

Disjunction: Exercise

Prove $(p \wedge q) \vee (p \wedge r) \vdash p \wedge (q \vee r)$:

1	$(p \wedge q) \vee (p \wedge r)$	premise
2	$p \wedge q$	assumption (1)
3	p	$\wedge e_1$ 2
4	q	$\wedge e_2$ 2
5	$q \vee r$	$\vee i_1$ 4
6	$p \wedge (q \vee r)$	$\wedge i$ 3, 5
7	$p \wedge r$	assumption (2)
8	p	$\wedge e_1$ 7
9	r	$\wedge e_2$ 7
10	$q \vee r$	$\vee i_2$ 9
11	$p \wedge (q \vee r)$	$\wedge i$ 8,10
12	$p \wedge (q \vee r)$	$\vee e$ 1, 2-6, 7-11

Natural Deduction: Implication

Proof rules for *implication*:

$\rightarrow$ introduction

$$\frac{\boxed{\begin{array}{c}\phi \\ \vdots \\ \psi\end{array}}}{\phi \rightarrow \psi} \rightarrow i$$

$\rightarrow$ elminiation

$$\frac{\phi \qquad \phi \rightarrow \psi}{\psi} \rightarrow e$$

Implication: Exercise

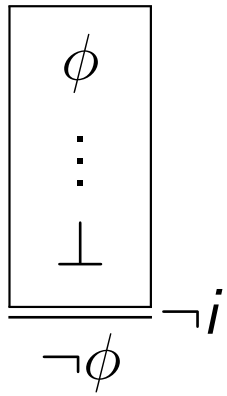
Prove $p \to q, q \to r \vdash p \to r$:

1	$p \to q$	premise
2	$q \to r$	premise
3	p	assumption
4	q	$\to e$ 1, 3
5	r	$\to e$ 2, 4
6	$p \to r$	$\to i$ 3-5

Natural Deduction: Negation

Proof rules for *negation*:

$\neg$ introduction



$\neg$ elimination:

$$\frac{\phi \qquad \neg\phi}{\bot}\neg q$$

Negation: Exercise

Prove $p \to q, p \to \neg q \vdash \neg p$:

1	$p \to q$	premise
2	$p \to \neg q$	premise
3	p	assumption
4	q	$\to e$ 1,3
5	$\neg q$	$\to e$ 2,3
6	$\bot$	$\neg e$ 4, 5
7	$\neg p$	$\neg i$ 3-6

Negation: Exercise (2)

Prove $\neg p \vee q \vdash p \rightarrow q$:

1	$\neg p \vee q$	premise
2	$\neg p$	assumption ($\vee e_1$)
3	p	assumption (contradiction)
4	$\bot$	$\neg e$ 3, 2
5	q	$\bot e$ 4
6	$p \rightarrow q$	$\rightarrow i$ 3-5
7	q	assumption ($\vee e_2$)
8	p	assumption
9	q	copy 7
10	$p \rightarrow q$	$\rightarrow i$ 8, 9
11	$p \rightarrow q$	$\vee e$ 1, 2-6, 7-10

Natural Deduction: Falsum

Proof rules for *falsum*:

$\bot$ introduction: there are no proof rules for the introduction of $\bot$

$\bot$ elimination: $$\frac{\bot}{\phi}\bot e$$

Natural Deduction: Double Negation

Proof rules for *double negation*:

$\neg\neg$ introduction: $$\frac{\phi}{\neg\neg\phi}\neg\neg i$$

$\neg\neg$ elimination: $$\frac{\neg\neg\phi}{\phi}\neg\neg e$$

Natural Deduction: Derived Rules

Modus Tollens: $\dfrac{\phi \to \psi \qquad \neg\psi}{\neg\phi}\,MT$

Reduction Ad Absurdum: $\dfrac{\boxed{\begin{array}{c}\neg\phi \\ \vdots \\ \bot\end{array}}}{\phi}\,RAA$

Tertium Non Datur: $\dfrac{}{\phi \vee \neg\phi}\,TND$

TND can also be called Law of the Excluded Middle.

Natural Deduction is Sound and Complete

Natural deduction is sound:

$$\text{if } \phi_1, \cdots \phi_n \vdash \psi, \text{ then } \phi_1, \cdots \phi_n \models \psi$$

Natural deduction is complete:

$$\text{if } \phi_1, \cdots \phi_n \models \psi, \text{ then } \phi_1, \cdots \phi_n \vdash \psi$$

Table of Contents

Validity and Satisfiability of Propositional Formulas

A formula ϕ is *valid* if for any valuations v, $[\![\phi]\!](v) = \top$
A formula ϕ is *satisfiable* if there exists a valuation v such that $[\![\phi]\!](v) = \top$

Validity and Satisfiability: Example

$p \wedge q$	satisfiable
$p \rightarrow (q \rightarrow p)$	valid(and satisfiable)
$p \wedge \neg p$	unsatisfiable

Deciding Validity

What are the means to decide whether or not a given formula ϕ is valid?

- Use techniques for semantic entailment (e.g., natural deduction)
- Use a calculus for semantical equivalence to prove that $\phi \equiv \top$.
- Transform ϕ into some normal form that is semantically equivalent and then apply dedicated (syntactic) techniques. (ϕ and ψ are *semantically equivalent* (not $\phi \equiv \psi$) iff $\phi \models \psi$ and $\psi \models \phi$.

Deciding Validity (2)

Lemma 1.41

A decision procedure for validity can be used for semantic entailment.

$$\phi_1, \cdots \phi_n \models \psi \text{ iff } \models \phi_1 \rightarrow (\phi_2 \rightarrow \cdots \rightarrow (\phi_n \rightarrow \psi))$$

Deciding Validity (3)

- If I'm wealthy, then I'm happy. I am happy.Therefore, I'm wealthy.
- If John drinks beer, he is at least 21 years old. John does not drink beer.Therefore, John is not yet 21 years old.
- If I study, then I will not fail basket weaving 101. If I do not play cards too often, then I will study. I failed basket weaving 101.Therefore, I played cards too often.

Conjunctive Normal Form

A *literal* is either an atom p or the negation of an atom $\neg p$.
A formula ϕ is in *conjunctive normal form (CNF)* if it is a conjunction of a number of disjunctions and literals only.

$L ::= P \mid \neg P$	literal
$C ::= L \mid C \vee C$	clause
$CNF ::= C \mid CNF \wedge CNF$	CNF

CNF Examples

p, $\neg p$	CNF
$\neg\neg p$	not CNF
$p \wedge \neg p$	CNF
$(p \vee \neg r) \wedge (\neg r \vee s) \wedge q$	CNF
$(p \wedge \neg q) \vee q$	not CNF

Validity in CNF

Remember a formula is valid iff any of its equivalent formulas is valid.
Reduce the problem of determining whether *any* ϕ is valid to the problem of computing an equivalent $\psi \equiv \phi$ such that ψ is in CNF and then checking ψ.
Deciding validity in CNF $(C_1 \wedge C_2 \wedge \cdots \wedge C_n)$ is incremental: each clause C_i must be valid individually.
Each clause C_i consists of a disjunction of literals $L_1 \vee L_2 \vee \cdots \vee L_m$.
A disjunction of literals is valid iff there are $1 \leq i, j \leq m$ such that L_i is $\neg L_j$.

Validity in CNF (2)

We now have a simple way to check the validity of $\models \phi$ as long as ϕ is in CNF: inspect all conjuncts ψ_k of ϕ and search for atoms in ψ_k so that ψ_k also contains their negation. If a match is found for all conjuncts we have $\models \phi$. Otherwise (i.e., some conjunct contains no pair L_i and $\neg L_j$), ϕ is not valid.

Transformation into CNF

1. IF: Remove all occurrences of $\rightarrow$: translate $\psi \rightarrow \eta$ to $\neg\psi \vee \eta$ (in: formula, out: formula without $\rightarrow$).
2. NNF: Obtain a *negation normal form* (NNF) where only atoms are negated (in: formula without $\rightarrow$, out: formula in NNF):

$$\begin{array}{l} N ::= P | \neg P | (N \vee N) | (N \wedge N) \\ P ::= p | q | r | \cdots \end{array}$$

3. CNF: Apply distribution laws (in: formula in NNF, out: formula in CNF):

$$\text{replace } (\phi_1 \wedge \phi_2) \vee \psi \text{ by } (\phi_1 \vee \psi) \wedge (\phi_2 \vee \psi)$$

$$\text{replace } \phi \vee (\psi_1 \wedge \psi_2) \text{ by } (\phi \vee \psi_1) \wedge (\phi \vee \psi_2)$$

Therefore, $CNF(NNF(IF(\phi)))$ is in *CNF* and semantically equivalent with ϕ.

Transformation into CNF: IF algorithm

Remove implications from the formula by applying the following replacement until you reach a fix-point: $\psi \rightarrow \eta$ to $\neg\psi \vee \eta$

Inductive definition of IMPL_FREE:

$$
\begin{aligned}
IF(p) &= p \\
IF(\neg\phi) &= \neg IF(\phi) \\
IF(\phi_1 \wedge \phi_2) &= IF(\phi_1) \wedge IF(\phi_2) \\
IF(\phi_1 \vee \phi_2) &= IF(\phi_1) \vee IF(\phi_2) \\
IF(\phi_1 \rightarrow \phi_2) &= \neg IF(\phi_1) \vee IF(\phi_2)
\end{aligned}
$$

Properties of IF: it is (i) well-defined (terminates for any input), (ii) $IF(\psi) \equiv \psi$ (output of both formulas are semantically equivalent),and (iii) $IF(\psi)$ is an implication-free formula for any formula ψ.

Transformation into CNF: NNF algorithm

Simplify formula into negation normal form by repeatedly applying pattern rewriting rules:

$\neg\neg\phi$	replace by	ϕ
$\neg(\phi \wedge \psi)$	replace by	$\neg\phi \vee \neg\psi$
$\neg(\phi \vee \psi)$	replace by	$\neg\phi \wedge \neg\psi$

Inductive definition of NFF:

$$
\begin{aligned}
NNF(p) &= p \\
NNF(\neg p) &= \neg p \\
NNF(\neg\neg\phi) &= NNF(\phi) \\
NNF(\neg(\phi \wedge \psi)) &= NNF(\neg\phi) \vee NNF(\neg\psi) \\
NNF(\neg(\phi \vee \psi)) &= NNF(\neg\phi) \wedge NNF(\neg\psi) \\
NNF(\phi \wedge \psi) &= NNF(\phi) \wedge NNF(\psi) \\
NNF(\phi \vee \psi) &= NNF(\phi) \vee NNF(\psi)
\end{aligned}
$$

Transformation into CNF: NNF algorithm (2)

Properties of *NNF*: it is (i) well-defined (terminates for any input), (ii) $NNF(\psi) \equiv \psi$ (output of both formulas are semantically equivalent),and (iii) $NNF(\psi)$ is a negation-free formula for any formula ψ.

Transformation into CNF: CNF algorithm

Simplify formula into conjunctive normal form (CNF) by repeatedly applying pattern rewriting rules:

$(\phi_1 \wedge \phi_2) \vee \psi$ replace by $(\phi_1 \vee \psi) \wedge (\phi_2 \vee \psi)$

$\phi \vee (\psi_1 \wedge \psi_2)$ replace by $(\phi \vee \psi_1) \wedge (\phi \vee \psi_2)$

Transformation into CNF: CNF algorithm (2)

Inductive definition of CNF:

$$
\begin{aligned}
CNF(p) &= p \\
CNF(\neg p) &= \neg p \\
CNF(\phi_1 \wedge \phi_2) &= CNF(\phi_1) \wedge CNF(\phi_2) \\
CNF(\phi_1 \vee \phi_2) &= D(CNF(\phi_1), CNF(\phi_2)) \\
D(\phi_1, \phi_2) &= \begin{cases} D(\phi_{11}, \phi_2) \wedge D(\phi_{12}, \phi_2) & \phi_1 = \phi_{11} \wedge \phi_{12} \\ D(\phi_1, \phi_{21}) \wedge D(\phi_1, \phi_{22}) & \phi_2 = \phi_{21} \wedge \phi_{22} \\ \phi_1 \vee \phi_2 & \textit{otherwise} \end{cases}
\end{aligned}
$$

Properties of *CNF* and *D*: *CNF* and *D* are (i) well-defined (terminate for any input), (ii) $D(\phi, \psi) \equiv \phi \vee \psi$ and $CNF(\phi) \equiv \phi$ (output of both formulas are semantically equivalent),and (iii) $CNF(\phi)$ is in CNF for any formula ϕ in *NNF* and $D(\phi, \psi)$ is in CNF for any formulas ϕ and ψ in CNF.

CNF: Example

Find a CNF for $p \vee \neg q \rightarrow r$:

$p \vee \neg q \rightarrow r$	premise
$\neg(p \vee \neg q) \vee r$	apply IMPL_FREE
$(\neg p \wedge \neg\neg q) \vee r$	apply NNF
$(\neg p \wedge q) \vee r$	apply NNF
$(\neg p \vee r) \wedge (q \vee r)$	apply CNF

Table of Contents

SAT Solver

Find satisfying valuations to a propositional formula.
Develop a systematic approach to *test* all possible valuations to find a satisfiable valuation.
SAT solving is NP-complete, so the worst-case complexity will always be exponential.But good heuristics exist.

Forcing Laws: Negation

ϕ	$\neg\phi$
T	F
F	T

$$\frac{\neg \quad o}{T \iff F}$$

$$\frac{\neg \quad o}{F \iff T}$$

Forcing Laws: Conjunction

ϕ	ψ	$\phi \wedge \psi$
T	T	T
T	F	F
F	T	F
F	F	F

$$\begin{array}{lll} \wedge & \text{———} & \phi, \psi \\ T & \Longrightarrow & T, T \end{array}$$

$$\begin{array}{lll} \phi, \psi & \text{———} & \wedge \\ T, T & \Longrightarrow & T \\ ?, F & \Longrightarrow & F \\ F, ? & \Longrightarrow & F \end{array}$$

$$\begin{array}{lll} \wedge, \phi & \text{———} & \psi \\ F, T & \Longrightarrow & F \end{array}$$

$$\begin{array}{lll} \wedge, \psi & \text{———} & \phi \\ F, T & \Longrightarrow & F \end{array}$$

Forcing Laws: Completeness

Is this enough?We now have $\neg$ and $\wedge$. We can convert any propositional formula (without loss of generality) to a formula that only contains $\neg$ and $\wedge$.

Simplify formula into $\neg$, $\wedge$

$$
\begin{aligned}
T(p) &= p \\
T(\neg\phi) &= \neg T(\phi) \\
T(\phi \wedge \psi) &= T(\phi) \wedge T(\psi) \\
T(\phi \vee \psi) &= \neg(\neg T(\phi) \wedge \neg T(\psi)) \\
T(\phi \rightarrow \psi) &= \neg(T(\phi) \wedge \neg T(\psi))
\end{aligned}
$$

This translation results in a linear growth in the formula size.

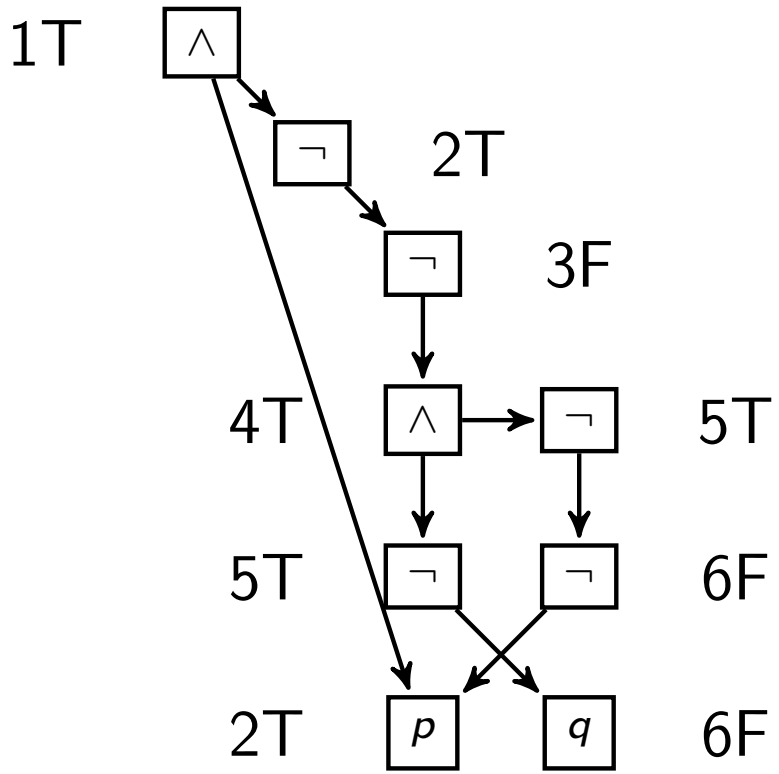
SAT Solving

1. Convert formula to $\neg$ and $\wedge$
2. Translate the formula to a DAG, sharing common subterms.
3. Set the root to T and apply the forcing rules.

The formula is satisfiable iff all nodes are consistently annotated.

Example: Satisfiability

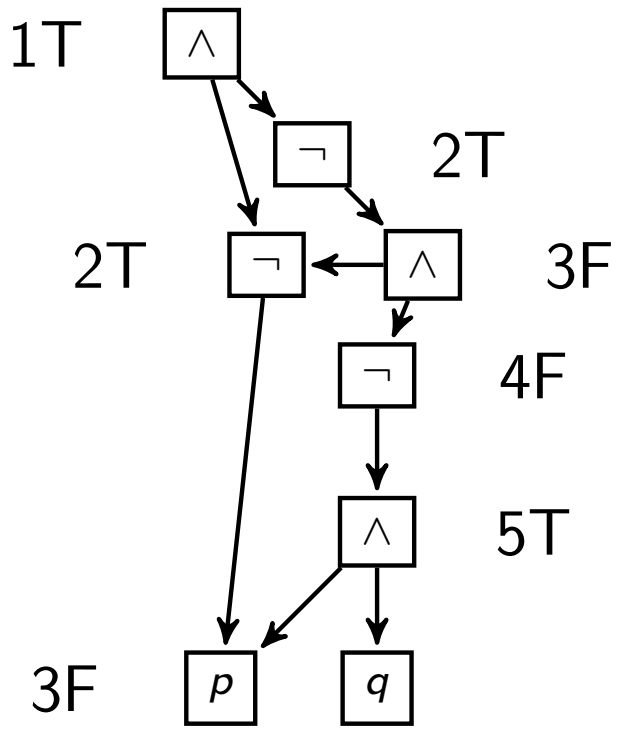
Formula: $p \wedge \neg(q \vee \neg p) \equiv p \wedge \neg\neg(\neg q \wedge \neg\neg p)$:



Is the formula satisfiable? Yes: $p = T, q = F$ is a witness.

Example: Validity

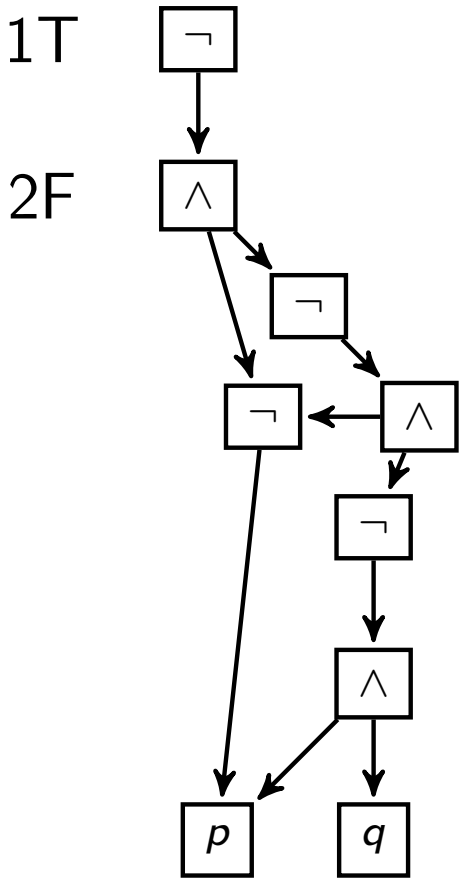
Show the validity of $(p \vee (p \wedge q)) \rightarrow p$.
This formula is valid if $\neg((p \vee (p \wedge q)) \rightarrow p)$ is not satisfiable.Translated formula: $\neg(\neg p \wedge \neg(p \wedge q)) \wedge \neg p$.



Contradiction!

Example: Satisfiability

Formula: $(p \vee (p \wedge q)) \rightarrow p \equiv \neg((p \vee (p \wedge q)) \rightarrow p)$



We have an unsatisfiable formula. Now what?

Limitation of the SAT solver algorithm

Fails for all formulas of the form $\neg(\phi_1 \wedge \phi_2)$.
Yet, some are valid and thus satisfiable:

$$\top \equiv p \rightarrow p \equiv \neg(p \wedge \neg p)$$

Some are not valid and thus not satisfiable:

$$\bot \equiv \neg\top \equiv \neg(\top \wedge \top) \equiv \neg(p \rightarrow p \wedge p \rightarrow p) \equiv \neg(\neg(p \wedge \neg p) \wedge \neg(p \wedge \neg p))$$

Extended Algorithm

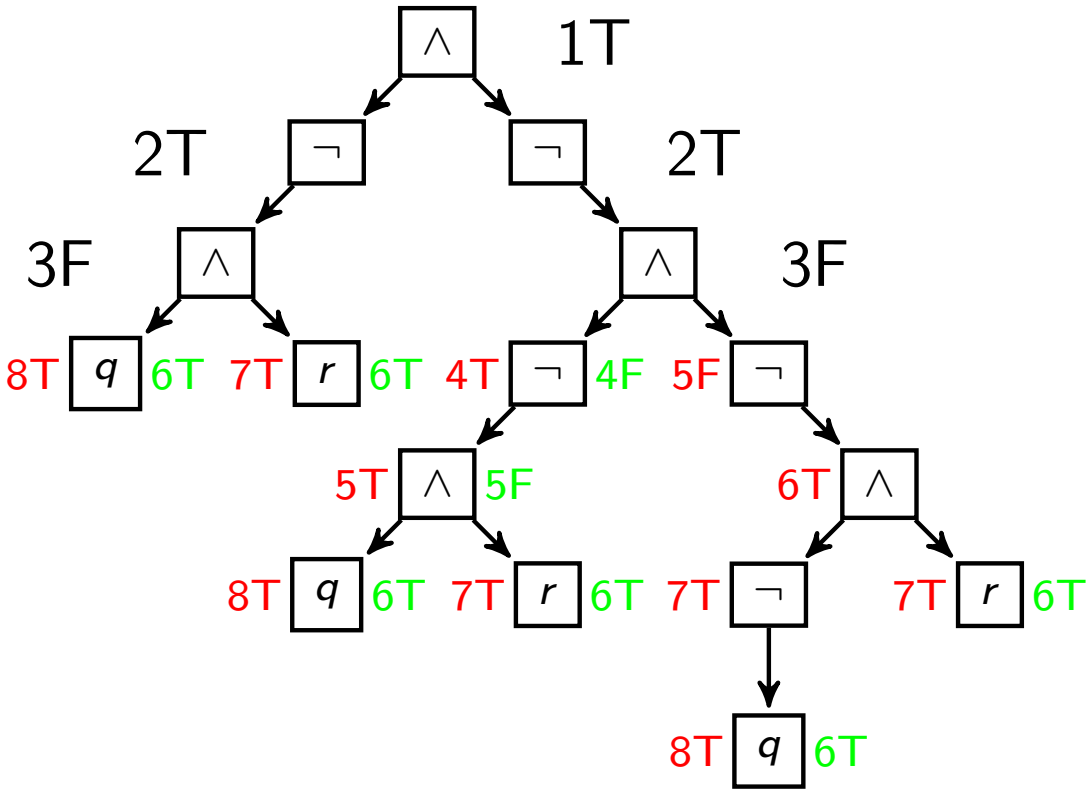
SAT Checking

1. Pick an unmarked node and add temporary T and F marks.
2. Use the forcing rules to propagate both marks.
3. If both marks lead to a contradiction, report a contradiction.
4. If both marks lead to some node having the same value, permanently assign the node that value.
5. Erase the remaining temporary marks and continue.

Complexity: $O(n^3)$: (i) testing each unmarked node $O(n)$, (ii) testing a given unmarked node $O(n)$, (iii) repeating the process when a new node is marked $O(n)$.

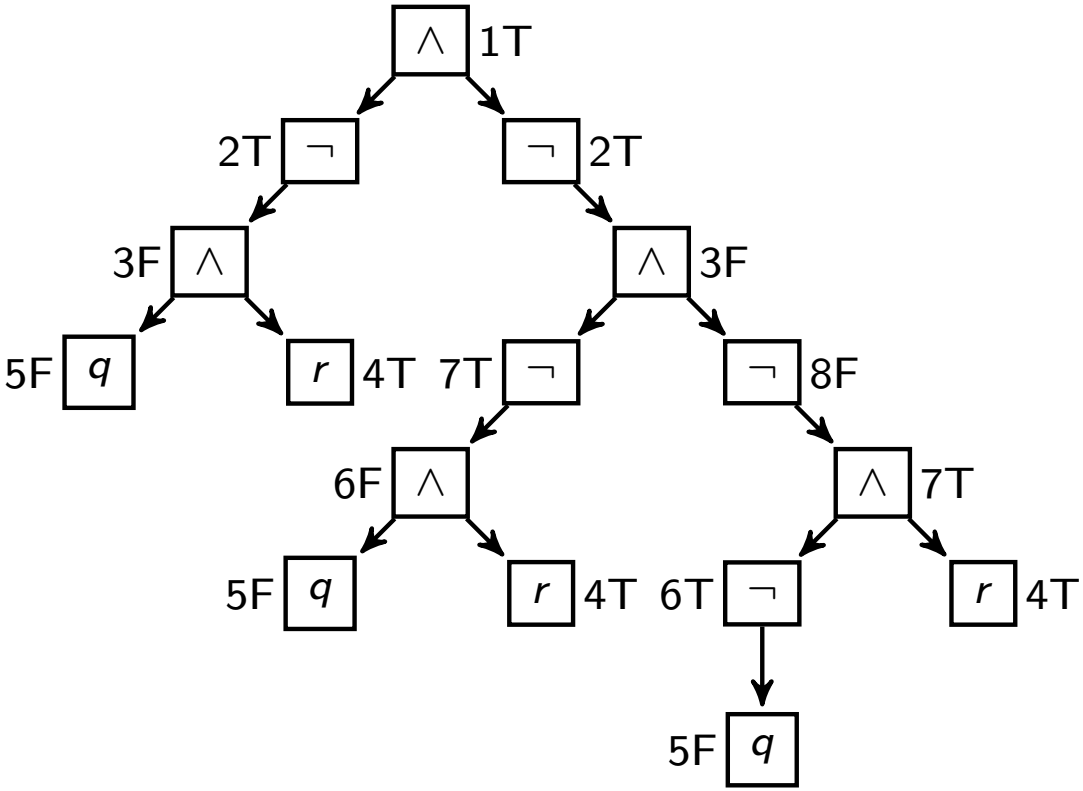
Extended Algorithm: Example

Formula: $\neg(q \wedge r) \wedge \neg(\neg(q \wedge r) \wedge \neg(\neg q \wedge r))$:



r is true in both cases. Fix r to T.

Extended Algorithm: Example (2)



Satisfiable!

Questions?

?