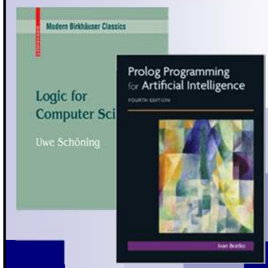


FLORIDA INTERNATIONAL UNIVERSITY

Propositional Logic (II)

Dr. Antonio L. Bajuelos
FIU School of Computing & Information Sciences

Propositional Logic.

Definition (**suitable assignment** and **model**)

- Let F be a formula and let $\mathcal{A}$ be an **assignment**, i.e. a mapping from a set of $\{A_1, A_2, \ldots\}$ to $\{0,1\}$ where $A_1, A_2, \ldots$ are atomic propositions in F.
- If $\mathcal{A}$ is defined for every atomic formula A_i occurring in F, then $\mathcal{A}$ is called **suitable** for F.
- If $\mathcal{A}$ is **suitable** for F, and if $\mathcal{A}$**(F) = 1**, then we write $\mathcal{A} \models$ **F**. In this case we say $\mathcal{A}$ **is a model for F**.
- Otherwise we write $\mathcal{A} \not\models$ **F**, and say: under the assignment $\mathcal{A}$, $\mathcal{A}$ **is not a model for F**.

Propositional Logic.

Definition (**satisfiable/unsatisfiable** and **valid** formula)

- Let F be a formula and let $\mathcal{A}$ be an assignment, i.e. a mapping from a set of $\{A_1, A_2, \ldots\}$ of F to $\{0,1\}$.
- A formula F is **satisfiable** if F has at least one model, otherwise F is called **unsatisfiable** or **contradictory**.
- A **formula** F is called **valid** (or a **tautology**) if every suitable assignment for F is a model for F. In this case we write $\models F$, and otherwise $\not\models F$.
- **Theorem:** A formula F is a **tautology** if and only if $\neg F$ is **contradictory (unsatisfiable)**.

3

Propositional Logic. Semantic

Examples:

- $\mathbf{F_1 = A \vee \neg A}$
  - $\mathbf{F_1}$ is a **valid** formula or a **tautology**
- $\mathbf{F_2 = A \wedge \neg A}$
  - $\mathbf{F_2}$ is a **contradiction (unsatisfiable)**

Exercises:

- $\mathbf{F_3 = A \rightarrow (B \rightarrow A)}$ is **???**
- $\mathbf{F_4 = A \rightarrow (A \rightarrow B)}$ is **???**
- If **F** is **valid**, then $\neg$**F** is **satisfiable** **???**

4

Propositional Logic. Semantic

Example:

- Let **F = (¬A → (A → B)).** Using truth tables we can verify that:

A	B	$\neg A$	$(A \rightarrow B)$	F
0	0	1	1	1
0	1	1	1	1
1	0	0	0	1
1	1	0	1	1

- Then F is a **valid** formula or a **tautology**

Propositional Logic. The SAT problem.

Note that...

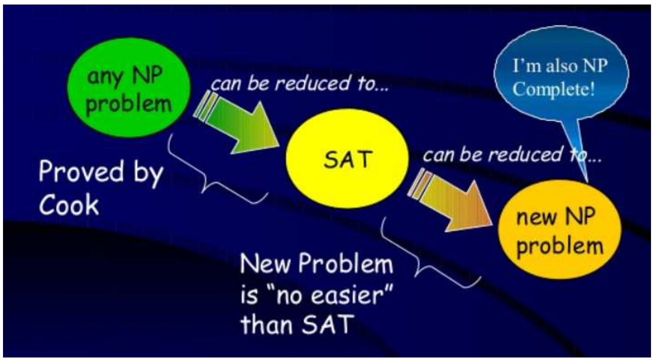
- The **truth-table method** allows us to test formulas for **satisfiability** or for **validity** in an **algorithmic way**.
- But note that the expense of this algorithm is immense: For a formula containing n atomic formulas, 2^n rows of the truth-table have to be evaluated.
- This **exponential behavior** regarding the running time of potential algorithms for the **satisfiability problem** in propositional logic does not seem to be improvable.
- **The satisfiability (SAT) problem is NP-complete***!

*the most notable (informal) property of **NP-complete problems** is that **no fast solution** to them is known. That is, the time required to solve the problem using any currently known algorithm increases very quickly as the size of the problem grows.

Propositional Logic. The SAT problem.

Note that…

- **The satisfiability (SAT) problem is NP-complete*!**
- How to prove NP-completeness?

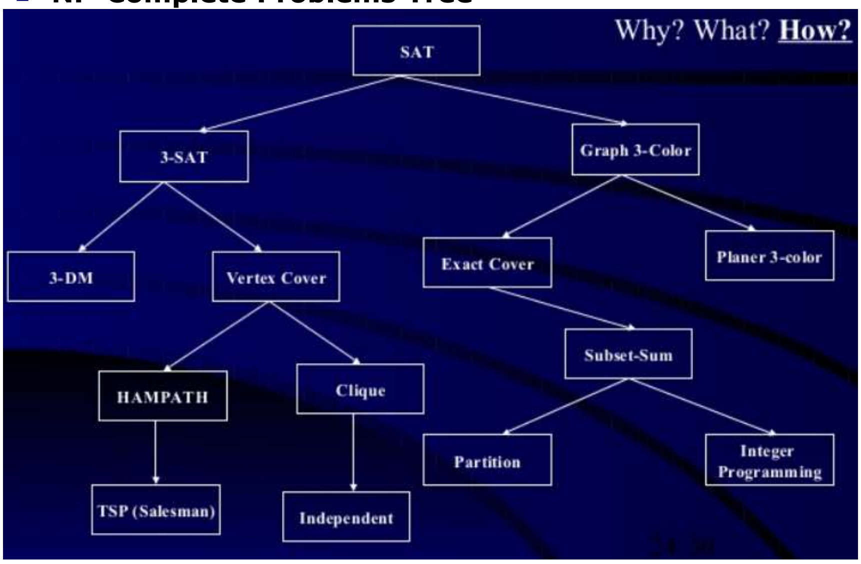


(https://www.slideshare.net/GeneMooLee/introduction-to-np-completeness)

7

Propositional Logic. The SAT problem.

- **NP-Complete Problems Tree**



(https://www.slideshare.net/GeneMooLee/introduction-to-np-completeness)

Propositional Logic. Semantic

Exercises:

- Construct truth-tables for each of the following formulas and indicate if the formula is a **tautology**.
  - $A \leftrightarrow (B \rightarrow C)$
  - $(A \wedge \neg A) \rightarrow B$

$\mathcal{A}(F)$	$\mathcal{A}(G)$	$\mathcal{A}((F \leftrightarrow G))$
0	0	1
0	1	0
1	0	0
1	1	1

- Let $\mathcal{A}(A)= 1$, $\mathcal{A}(B)= 0$, $\mathcal{A}(C)= 0$, and $\mathcal{A}(D)= 1$. Verify if the following formula is **true** or **false**.
  - $F= C \vee (D \vee (A \wedge B))$

Propositional Logic. Semantic

Homework:

- Construct truth-tables for each of the following formulas.

$$((A \wedge B) \wedge (\neg B \vee C))$$

$$\neg(\neg A \vee \neg(\neg B \vee \neg A))$$

$$(A \leftrightarrow (B \leftrightarrow C))$$

Equivalence

Definition

- Two formulas **F** and **G** are (semantically/logically) **equivalent** if for every assignment $\mathcal{A}$ that is suitable for both F and G, $\mathcal{A}$**(F) =** $\mathcal{A}$**(G)**. Symbolically we denote this by **F ≡ G**.

Equivalence (cont)

Theorem

- For all formulas F, G, and H, the following equivalences hold.

$$
\begin{array}{rcll}
(F \wedge F) & \equiv & F & \\
(F \vee F) & \equiv & F & \text{(Idempotency)} \\[1ex]
(F \wedge G) & \equiv & (G \wedge F) & \\
(F \vee G) & \equiv & (G \vee F) & \text{(Commutativity)} \\[1ex]
((F \wedge G) \wedge H) & \equiv & (F \wedge (G \wedge H)) & \\
((F \vee G) \vee H) & \equiv & (F \vee (G \vee H)) & \text{(Associativity)} \\[1ex]
(F \wedge (F \vee G)) & \equiv & F & \\
(F \vee (F \wedge G)) & \equiv & F & \text{(Absorption)} \\[1ex]
(F \wedge (G \vee H)) & \equiv & ((F \wedge G) \vee (F \wedge H)) & \\
(F \vee (G \wedge H)) & \equiv & ((F \vee G) \wedge (F \vee H)) & \text{(Distributivity)} \\[1ex]
\neg\neg F & \equiv & F & \text{(Double Negation)}
\end{array}
$$

Equivalence (cont)

Theorem (cont…)

- For all formulas F, G, and H, the following equivalences hold.

$$\begin{array}{rcll} \neg(F \wedge G) & \equiv & (\neg F \vee \neg G) & \\ \neg(F \vee G) & \equiv & (\neg F \wedge \neg G) & \text{(deMorgan's Laws)} \\ (F \vee G) & \equiv & F\text{, if } F \text{ is a tautology} & \\ (F \wedge G) & \equiv & G\text{, if } F \text{ is a tautology} & \text{(Tautology Laws)} \\ (F \vee G) & \equiv & G\text{, if } F \text{ is unsatisfiable} & \\ (F \wedge G) & \equiv & F\text{, if } F \text{ is unsatisfiable} & \text{(Unsatisfiability Laws)} \end{array}$$

13

Equivalence (cont)

Theorem (cont…)

- **Proof.**

  All equivalences can be shown easily using the **semantic definition of propositional logi**c. Also, we can verify them using truth tables. As an example we show this for the **first absorption law**.

$\mathcal{A}(F)$	$\mathcal{A}(G)$	$\mathcal{A}((F \vee G))$	$\mathcal{A}((F \wedge (F \vee G)))$
0	0	0	0
0	1	1	0
1	0	1	1
1	1	1	1

  The first column and the fourth column coincide. Therefore, it follows:

$$(F \wedge (F \vee G)) \equiv F$$

■

14

Equivalence (cont)

Example:

- Using the above equivalences and the substitution theorem (ST) we can prove that:

$$((A \vee (B \vee C)) \wedge (C \vee \neg A)) \equiv ((B \wedge \neg A) \vee C)$$

because we have:

$((A \vee (B \vee C)) \wedge (C \vee \neg A))$	
$\equiv \; (((A \vee B) \vee C) \wedge (C \vee \neg A))$	Associativity
$\equiv \; ((C \vee (A \vee B)) \wedge (C \vee \neg A))$	Commutativity
$\equiv \; (C \vee ((A \vee B) \wedge \neg A))$	Distributivity
$\equiv \; (C \vee (\neg A \wedge (A \vee B))$	Commutativity
$\equiv \; (C \vee ((\neg A \wedge A) \vee (\neg A \wedge B))$	Distributivity
$\equiv \; (C \vee (\neg A \wedge B))$	Unsatisfibility Law
$\equiv \; (C \vee (B \wedge \neg A))$	Commutativity
$\equiv \; ((B \wedge \neg A) \vee C)$	Commutativity

15

Equivalence (cont)

Important remarks:

- The associativity law gives us the justification for a certain freedom in writing down formulas. For example, the notation:

$$F = A \wedge B \wedge C \wedge D$$

refers to an arbitrary formula from the following list.

$$(((A \wedge B) \wedge C) \wedge D)$$
$$((A \wedge B) \wedge (C \wedge D))$$
$$((A \wedge (B \wedge C)) \wedge D)$$
$$(A \wedge ((B \wedge C) \wedge D))$$
$$(A \wedge (B \wedge (C \wedge D)))$$

- Since all these formulas are equivalent to each other, from the semantic viewpoint it does not matter which of the formulas is referred to.

16

Equivalence (cont)

Exercise/Homework:

- Exclusive disjunction (or **exclusive or**) essentially means "**either one, but not both**". The exclusive disjunction can be expressed in terms of the **conjunction** ($\wedge$), the **disjunction** ($\vee$), and the **negation** ($\neg$) as follows:

$$\mathbf{A \oplus B \equiv (A \vee B) \wedge \neg(A \wedge B)}$$

Using the theorem of equivalences and the substitution theorem prove that:

$$\mathbf{A \oplus B \equiv \neg(A \leftrightarrow B)}$$

17

Normal Forms

Definition (normal forms):

- A **literal** is an **atomic formula** or the **negation** of an **atomic formula**.
- A formula F is in **Conjunctive Normal Form** (**CNF**) if it is a conjunction of disjunctions of literals, i.e.

$$F = (\bigwedge_{i=1}^{n} (\bigvee_{j=1}^{m_i} L_{i,j})) \ ,$$

$$\text{where } L_{i,j} \in \{A_1, A_2, \ldots\} \cup \{\neg A_1, \neg A_2, \ldots\}$$

- **Example:**
  - **(A $\vee$ ¬C $\vee$ ¬D) $\wedge$ (¬B $\vee$ ¬C $\vee$ ¬D) is in CNF**

18

Normal Forms

Definition (normal forms):

- A formula F is in **Disjunctive Normal Form** (**DNF**) if it is a disjunction of conjunctions of literals, i.e.

$$F = (\bigvee_{i=1}^{n} (\bigwedge_{j=1}^{m_i} L_{i,j})) ,$$

$$\text{where } L_{i,j} \in \{A_1, A_2, \ldots\} \cup \{\neg A_1, \neg A_2, \ldots\}$$

- **Example:**
  - **$(A \wedge \neg D) \vee (B \wedge \neg C \wedge \neg D)$ is in DNF**

Normal Forms. Reduction

Theorem (Normal Form reduction):

- For every formula F there is an equivalent formula F_1 in **CNF** and an equivalent formula F_2 in **DNF**.

(see Proof in Schöning's book, Section 1.2)

Normal Forms. Reduction

Algorithm (to transform a formula into equivalent **CNF**):

- *Given:* a formula F.
  - Substitute in F every occurrence of a subformula of the form
    - $(A \rightarrow B)$ by $(\neg A \vee B)$
    - $(A \leftrightarrow B)$ by $((A \wedge B) \vee (\neg A \wedge \neg B))$

    until no such subformulas occur.
  - Substitute in F every occurrence of a subformula of the form
    - $\neg\neg A$ by A
    - $\neg(A \wedge B)$ by $(\neg A \vee \neg B)$
    - $\neg(A \vee B)$ by $(\neg A \wedge \neg B)$

    until no such subformulas occur.
  - Substitute in F every occurrence of a subformula of the form
    - $(A \vee (B \wedge C))$ by $((A \vee B) \wedge (A \vee C))$
    - $((A \wedge B) \vee C)$ by $((A \vee C) \wedge (B \vee C))$

    until no such subformulas occur.

Normal Forms. Reduction

Example (to transform a formula into equivalent **CNF**):

- *Given* a formula **F =** $\mathbf{\neg(\neg A \vee B) \vee (C \rightarrow \neg D)}$
  - Substitute in F every occurrence of a subformula of the form
    - $(A \rightarrow B)$ by $(\neg A \vee B)$

    We obtain $\mathbf{\neg(\neg A \vee B) \vee (\neg C \vee \neg D)}$
  - Substitute in F every occurrence of a subformula of the form
    - $\neg\neg A$ by A
    - $\neg(A \vee B)$ by $(\neg A \wedge \neg B)$

    We obtain $\mathbf{(A \wedge \neg B) \vee (\neg C \vee \neg D)}$
  - Substitute in F every occurrence of a subformula of the form
    - $((A \wedge B) \vee C)$ by $((A \vee C) \wedge (B \vee C))$

    We obtain $\mathbf{(A \vee \neg C \vee \neg D) \wedge (\neg B \vee \neg C \vee \neg D)}$ **in CNF!**

Normal Forms. Reduction

Exercise/Homework

- Reduce to **CNF** the formula:

$$\mathbf{F = (\neg A \rightarrow B) \rightarrow (B \rightarrow \neg C)}$$