CS440/ECE448: Intro to Artificial Intelligence

Lecture 7: Propositional logic

Prof. Julia Hockenmaier
juliahmr@illinois.edu

http://cs.illinois.edu/fa11/cs440

Thursday's key concepts

Combining CSP search and inference:
Ordering variables (minimum remaining value, degree heuristics)
Ordering values (forward checking, MAC)

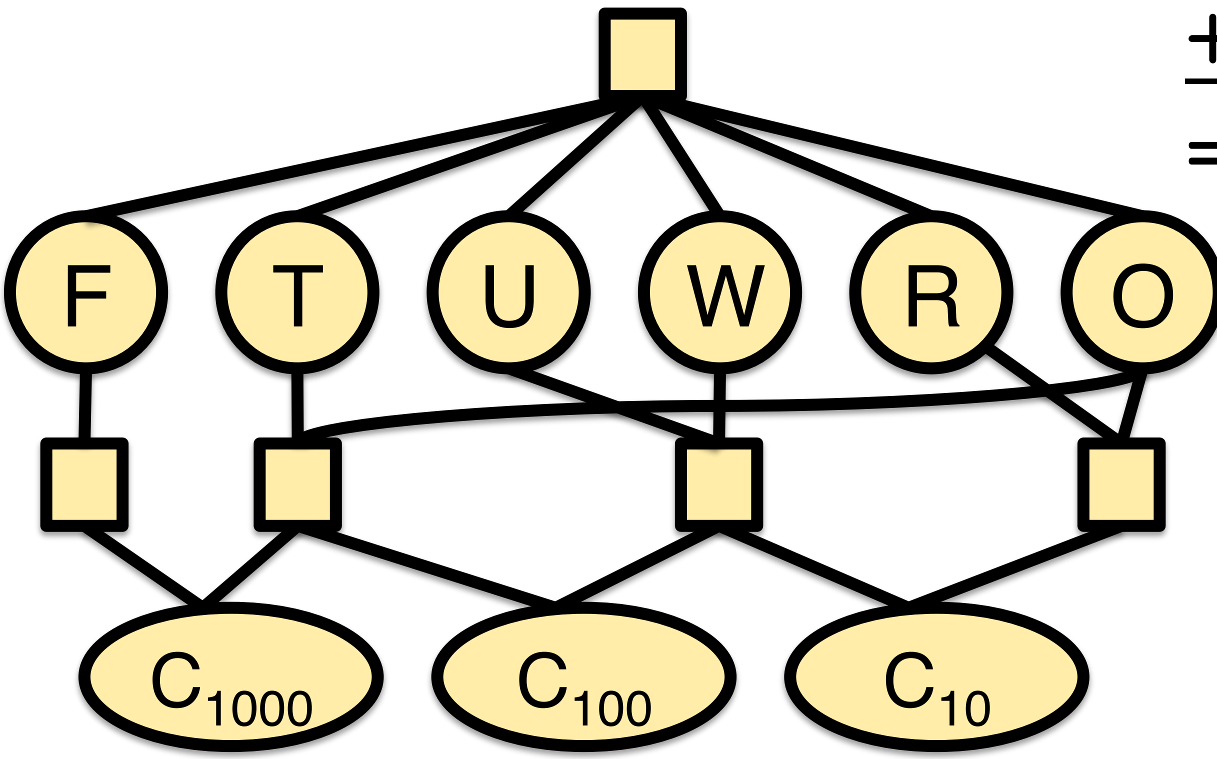
Global constraints:
Constraint hypergraph; auxiliary variables
Continuous domains:
bounds consistency

Path consistency and arc consistency

X is **arc consistent with respect to Y** if for every value of X there exists some value of Y such that C(X,Y) is satisfied.

X and Y are **path consistent with respect to Z** if for every pair of values of X and Y that satisfy C(X, Y), there exists some value of Z such that C(X,Z) and C(Y,Z) is satisfied.

Global (n-ary) constraints: Constraint Hypergraph




```
    TWO
+   TWO
-------
=  FOUR
```

Propositional logic

Propositional logic

Syntax: What is the language of well-formed formulas of propositional logic?

Semantics: What is the interpretation of a well-formed formula in propositional logic?

Inference rules and algorithms: How can we reason with propositional logic?

Syntax: the building blocks

Variables: $p \mid q \mid r \mid \ldots$

Constants: $\top$ (true) , $\bot$ (false)

Unary connectives: $\neg$ (negation)

Binary connectives: $\wedge$ (conjunction)

$\vee$ (disjunction)

$\rightarrow$ (implication)

Syntax: well-formed formulas

$$
\begin{aligned}
\text{WFF} &\longrightarrow \text{Atomic} \mid \text{Complex} \\
\text{Atomic} &\longrightarrow \text{Constant} \mid \text{Variable} \\
\text{WFF'} &\longrightarrow \text{Atomic} \mid (\text{Complex}) \\
\text{Complex} &\longrightarrow \neg\ \text{WFF'} \mid \\
&\quad\ \ \text{WFF'} \wedge \text{WFF'} \mid \\
&\quad\ \ \text{WFF'} \vee \text{WFF'} \mid \\
&\quad\ \ \text{WFF'} \rightarrow \text{WFF'}
\end{aligned}
$$

Semantics: truth values

The interpretation $[\![\alpha]\!]^v$ of a well-formed formula α under a model v is a truth value:

$$[\![\alpha]\!]^v \in \{true, false\}.$$

A model (=valuation) v is a complete* assignment of truth values to variables:

$$v(p) = true \quad v(q) = false, \ \ldots$$

*each variable is either true or false

With n variables, there are 2^n different models

Models of α ('$M(\alpha)$'): set of models where α is true

Interpretation $[\![\alpha]\!]^v$ of α

Interpretation of **constants**: $[\![\top]\!]^v = \mathit{true}$, $[\![\bot]\!]^v = \mathit{false}$

Interpretation of **variables** defined by v $\quad [\![p]\!]^v = v(p)$

Interpretation of **connectives** given by truth tables

if…	*….then:*
$[\![p]\!]^v$	$[\![\neg p]\!]^v$
true	*false*
false	*true*

if…		*….then:*		
$[\![p]\!]^v$	$[\![q]\!]^v$	$[\![p \wedge q]\!]^v$	$[\![p \vee q]\!]^v$	$[\![p \rightarrow q]\!]^v$
true	*true*	*true*	*true*	*true*
true	*false*	*false*	*true*	*false*
false	*true*	*false*	*true*	*true*
false	*false*	*false*	*false*	*true*

Validity and satisfiability

α is valid in a model m ('$m \models \alpha$') iff $m \in M(\alpha)$
= the model m satisfies α
(α is true in m)

α is valid ('$\models \alpha$') iff $\forall m: m \in M(\alpha)$
(α is true in all possible models. α is a tautology.)

α is satisfiable iff $\exists m: m \in M(\alpha)$
(α is true in at least one model, $M(\alpha) \neq \varnothing$)

Inference in propositional logic

Entailment

Definition:

α entails β (‘$\alpha \models \beta$’) *iff* $M(\alpha) \subseteq M(\beta)$

Entailment is monotonic:

If $\alpha \models \beta$, then $\alpha \wedge \gamma \models \beta$ for any γ

Proof: $M(\alpha \wedge \gamma) \subseteq M(\alpha) \subseteq M(\beta)$

We also write $\alpha,\gamma \models \beta$ or $\{\alpha,\gamma\} \models \beta$ for $\alpha \wedge \gamma \models \beta$

Logical equivalence

α is equivalent to β (‘$\alpha \equiv \beta$’) *iff* $M(\alpha) = M(\beta)$

$\alpha \vee \beta$	$\equiv$	$\beta \vee \alpha$	**Commutativity**
$\alpha \wedge \beta$	$\equiv$	$\beta \wedge \alpha$	
$(\alpha \vee \beta) \vee \gamma$	$\equiv$	$\alpha \vee (\beta \vee \gamma)$	**Associativity**
$(\alpha \wedge \beta) \wedge \gamma$	$\equiv$	$\alpha \wedge (\beta \wedge \gamma)$	
$\alpha \vee (\beta \wedge \gamma)$	$\equiv$	$(\alpha \vee \beta) \wedge (\alpha \vee \gamma)$	**Distributivity**
$\alpha \wedge (\beta \vee \gamma)$	$\equiv$	$(\alpha \wedge \beta) \vee (\alpha \wedge \gamma)$	

Entailment and implication

α entails β (‘$\alpha \models \beta$’) *iff* $\alpha \rightarrow \beta$ is valid
($\models \alpha \rightarrow \beta$)

Proof:

If $v \in M(\alpha)$: $[\![\alpha]\!]^v = \mathit{true}$ by definition.
So $[\![\alpha \rightarrow \beta]\!]^v = \mathit{true}$ only if $[\![\beta]\!]^v = \mathit{true}$ $(v \in M(\beta))$
Thus, $v \in M(\alpha)$ implies $v \in M(\beta)$.

If $v \notin M(\alpha)$: $[\![\alpha]\!]^v = \mathit{false}$ by definition.
So $[\![\alpha \rightarrow \beta]\!]^v = \mathit{true}$ regardless of $[\![\beta]\!]^v$
Thus, when $v \notin M(\alpha)$, $v \in M(\beta)$ or $v \notin M(\beta)$.

More logical equivalences

$\neg(\alpha \vee \beta)$	$\equiv$	$\neg \alpha \wedge \neg \beta$	**DeMorgan**
$\neg(\alpha \wedge \beta)$	$\equiv$	$\neg \alpha \vee \neg \beta$	
$\alpha \rightarrow \beta$	$\equiv$	$\neg \alpha \vee \beta$	**Implication elimination**
$\alpha \rightarrow \beta$	$\equiv$	$\neg \beta \rightarrow \neg \alpha$	**Contraposition**

Biconditional (equivalence) $\leftrightarrow$

We can also define a binary connective $\leftrightarrow$:

$$
\begin{aligned}
\alpha \leftrightarrow \beta \;&\equiv (\alpha \rightarrow \beta) \land (\beta \rightarrow \alpha) \\
&\equiv (\neg\,\alpha \lor \beta) \land (\neg\,\beta \lor \alpha) \\
&\equiv ((\neg\,\alpha \lor \beta) \land \neg\,\beta) \\
&\quad\;\; \lor (\neg\,\alpha \lor \beta) \land \alpha) \\
&\equiv ((\neg\,\alpha \land \neg\,\beta) \lor (\beta \land \neg\,\beta)) \\
&\quad\;\; \lor ((\neg\,\alpha \land \alpha) \;\lor (\beta \land \alpha)) \\
&\equiv (\neg\,\alpha \land \neg\,\beta) \lor (\beta \land \alpha)
\end{aligned}
$$

Literals and clauses

Literal: p, $\neg p$, q, $\neg q$,
an atomic formula, or a negated atomic formula

Clause: p, $\neg p$, $p \vee q$, $\neg q \vee p$,
a literal (= unit clause), or a disjunction of literals

Normal Forms

Every formula α in propositional logic has two equivalent normal forms:

Conjunctive Normal Form (CNF)
a conjunction of clauses
$\alpha \equiv (p_{11} \vee \ldots \vee p_{1n}) \wedge (p_{21} \vee \ldots \vee p_{2m}) \wedge \ldots$

Disjunctive Normal Form (DNF)
a disjunction of conjoined literals
$\alpha \equiv (q_{11} \wedge \ldots \wedge q_{1n}) \vee (q_{21} \wedge \ldots \wedge q_{2m}) \vee \ldots$

Inference in propositional logic

We often have prior domain knowledge.

Given a knowledge base $\mathrm{KB} = \{\varphi_1, \ldots, \varphi_n\}$ (a set of formulas that are true), how do we know α is valid given KB?

Validity: KB $\models \alpha$ (shorthand for $\varphi_1 \wedge \ldots \wedge \varphi_n \models \alpha$)
Satisfiability: $\exists\, m: m \in M(\mathrm{KB}) \wedge m \in M(\alpha)$
($M(\mathrm{KB})$ shorthand for $M(\varphi_1 \wedge \ldots \wedge \varphi_n)$)

Inference in propositional logic

How do we know whether α is valid or satisfiable given KB?

Model checking: (semantic inference)
Enumerate all models for KB and α.

Theorem proving: (syntactic inference)
Use inference rules to derive α from KB.

Inference rules

Modus ponens

$$\frac{\alpha \rightarrow \beta \qquad \alpha}{\beta}$$

And-elimination

$$\frac{\alpha \wedge \beta}{\beta}$$

Inference rules: equivalences

$\alpha \lor \beta$	$\equiv$	$\beta \lor \alpha$	**Commutativity**
$\alpha \land \beta$	$\equiv$	$\beta \land \alpha$	

As inference rules:

$$\frac{\alpha \lor \beta}{\beta \lor \alpha} \qquad \frac{\beta \lor \alpha}{\alpha \lor \beta} \qquad \frac{\alpha \land \beta}{\beta \land \alpha} \qquad \frac{\beta \land \alpha}{\alpha \land \beta}$$

Theorem proving as search

Proving α from KB:

States: sets of formulas that are true.

Initial state: KB

Goal state: any state that contains α

Actions: a set of inference rules

Inference procedures

A procedure P that derives α from KB…

$$KB \vdash_P \alpha$$

…is **sound** if it only derives valid sentences:

if $KB \vdash_P \alpha$, then $KB \models \alpha$ (soundness)

…is **complete** if it derives any valid sentence:

if $KB \models \alpha$, then $KB \vdash_P \alpha$ (completeness)

The resolution rule

Unit resolution:

$$\frac{p_1 \vee \dots \vee p_{i-1} \vee \mathbf{p_i} \vee p_{i+1} \vee \dots \vee p_n \qquad \neg\, \mathbf{p_i}}{p_1 \vee \dots \vee p_{i-1} \vee p_{i+1} \vee \dots \vee p_n}$$

Full resolution:

$$\frac{\mathbf{p_1 \vee \dots \vee p_i \vee \dots \vee p_n} \qquad \mathbf{q_1 \vee \dots \vee \neg\, p_i \vee \dots \vee q_m}}{\mathbf{p_1 \vee \dots \vee p_n} \vee \mathbf{q_1 \vee \dots \vee q_m}}$$

Final step: factoring (remove any duplicate literals from the result $A \vee A \equiv A$)

Proof by contradiction

How do we prove that $\alpha \models \beta$?

α entails β ('$\alpha \models \beta$') *iff* $\alpha \wedge \neg \beta$ not satisfiable.

Proof:

$\alpha \wedge \neg \beta$ not satisfiable *iff* $\models \neg(\alpha \wedge \neg \beta)$

Assume $\models \neg(\alpha \wedge \neg \beta)$.

$\models \neg \; \alpha \vee \beta)$

$\models \alpha \rightarrow \beta$.

Thus, $\neg(\alpha \wedge \neg \beta) \equiv \alpha \rightarrow \beta$.

A resolution algorithm

Goal: prove $\alpha \vDash \beta$' by showing that $\alpha \wedge \neg \beta$ is not satisfiable (*false*)

Observation:
Resolution derives a contradiction (*false*) if it derives the empty clause:

$$\frac{p_i \qquad \neg p_i}{\emptyset}$$

```
function PLresolution(α, β)
    input: formula α, // knowledge base
           formula β // query
    clauses := CNF(α ∧¬ β)
    new := {}
    while true:
        for each c1, c2 in clauses do
            resolvents :=  resolve(c1, c2)
            if ∅ in resolvents then return true;
            new := new ∪ resolvents
        if new ⊆ clauses then return false;
        clauses := clauses ∪ new
```

Completeness of Resolution

Resolution closure RC(S):
The set of all clauses that can be derived by resolution from a set of clauses S.
If S is finite, RC(S) is finite.

Ground resolution theorem:
If RC(S) contains ∅, S is not satisfiable.
If RC(S) does not contain ∅, S is satisfiable.

If RC(S) doesn't contain ∅…

…S is satisfiable, because we can build a model for its variables $p_1 \ldots . p_n$:

For i from 1….n:

 if a clause in RC(S) contains $\neg p_i$
  and all its other literals are false,
 then assign false to p_i
 otherwise assign true to p_i

Today's key concepts

Syntax of propositional logic:
- propositional variables, connectives, well-formed formulas

Semantics of propositional logic:
- interpretations, models, truth tables

Inference with propositional logic:
- model-checking, resolution

Your tasks

Reading:
7.3-7.5.2

Compass quiz:
due Thursday at 2am.