Notes adapted from lecture notes for CMSC 421 by B.J. Dorr

Artificial Intelligence 1: logic agents

Lecturer: Tom Lenaerts

Institut de Recherches Interdisciplinaires et de Développements en Intelligence Artificielle (IRIDIA)

Université Libre de Bruxelles

Standard Logical Equivalences

$(\alpha \wedge \beta) \equiv (\beta \wedge \alpha)$ commutativity of $\wedge$

$(\alpha \vee \beta) \equiv (\beta \vee \alpha)$ commutativity of $\vee$

$((\alpha \wedge \beta) \wedge \gamma) \equiv (\alpha \wedge (\beta \wedge \gamma))$ associativity of $\wedge$

$((\alpha \vee \beta) \vee \gamma) \equiv (\alpha \vee (\beta \vee \gamma))$ associativity of $\vee$

$\neg(\neg\alpha) \equiv \alpha$ double-negation elimination

$(\alpha \Rightarrow \beta) \equiv (\neg\beta \Rightarrow \neg\alpha)$ contraposition

$(\alpha \Rightarrow \beta) \equiv (\neg\alpha \vee \beta)$ implication elimination

$(\alpha \Leftrightarrow \beta) \equiv ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha))$ biconditional elimination

$\neg(\alpha \wedge \beta) \equiv (\neg\alpha \vee \neg\beta)$ de Morgan

$\neg(\alpha \vee \beta) \equiv (\neg\alpha \wedge \neg\beta)$ de Morgan

$(\alpha \wedge (\beta \vee \gamma)) \equiv ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma))$ distributivity of $\wedge$ over $\vee$

$(\alpha \vee (\beta \wedge \gamma)) \equiv ((\alpha \vee \beta) \wedge (\alpha \vee \gamma))$ distributivity of $\vee$ over $\wedge$

Terminology

- A sentence is valid iff its truth value is **t** in all interpretations ($\models\phi$)
  Valid sentences: <u>true</u>, ¬ <u>false</u>, P ∨ ¬ P
- A sentence is satisfiable iff its truth value is **t** in at least one interpretation
  Satisfiable sentences: P, <u>true</u>, ¬ P
- A sentence is unsatisfiable iff its truth value is **f** in all interpretations
  Unsatisfiable sentences: P ∧ ¬ P, <u>false</u>, ¬ <u>true</u>

Examples

Sentence	Valid?	
wealthy ⇒ wealthy ¬wealthy ∨wealthy	valid	
wealthy ⇒ happy	satisfiable, not valid	w = **t**, h = **f**
inverse (w ⇒ h) ⇒ (¬w ⇒¬h)	satisfiable, not valid	w=**f**, h=**t** w⇒h =**t**, ¬ w ⇒¬ h = **f**
contrapositive (w ⇒ h) ⇒ (¬h ⇒¬w)	valid	
w ∨ h ∨ (w⇒ h) w ∨ h ∨ ¬ w ∨ h	valid	

Examples

Sentence	Valid?	
wealthy ⇒ wealthy ¬wealthy ∨wealthy	valid	
wealthy ⇒ happy	satisfiable, not valid	w = **t**, h = **f**
inverse (w ⇒ h) ⇒ (¬w ⇒¬h)	satisfiable, not valid	w=**f**, h=**t** w⇒h =**t**, ¬ w ⇒¬ h = **f**
contrapositive (w ⇒ h) ⇒ (¬h ⇒¬w)	valid	
w ∨ h ∨ (w⇒ h) w ∨ h ∨ ¬ w ∨ h	valid	

Inference

- $KB \vdash_i \alpha$
- Soundness: Inference procedure i is sound if whenever $KB \vdash_i \alpha$, it is also true that $KB \models \alpha$
- Completeness: Inference procedure i is complete if whenever $KB \models \alpha$, it is also true that $KB \vdash_i \alpha$

Validity and Inference

$$((P \lor H) \land \neg H) \Rightarrow P$$

P	H	P ∨ H	(P ∨ H) ∧ ¬ H	((P ∨ H) ∧ ¬ H) ⇒ P
T	T	T	F	T
T	F	T	T	T
F	T	T	F	T
F	F	F	F	T

Rules of Inference

- $\alpha \vdash \beta$
- $\dfrac{\alpha}{\beta}$
- Valid Rules of Inference:
  - Modus Ponens
  - And-Elimination
  - And-Introduction
  - Or-Introduction
  - Double Negation
  - Unit Resolution
  - Resolution

Examples in Wumpus World

- Modus Ponens: $\alpha \Rightarrow \beta, \alpha \vdash \beta$
  (WumpusAhead ∧ WumpusAlive) ⇒ Shoot,
  (WumpusAhead ∧ WumpusAlive)
  ⊢ Shoot
- And-Elimination: $\alpha \land \beta \vdash \alpha$
  (WumpusAhead ∧ WumpusAlive)
  ⊢ WumpusAlive
- Resolution: $\alpha \lor \beta, \neg \beta \lor \gamma \vdash \alpha \lor \gamma$
  (WumpusDead ∨ WumpusAhead),
  (¬ WumpusAhead ∨ Shoot)
  ⊢ (WumpusDead ∨ Shoot)

$$\frac{\alpha \Rightarrow \beta \quad \alpha}{\beta}$$

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

$$\frac{\alpha \lor \beta \quad \neg \beta \lor \gamma}{\alpha \lor \gamma}$$

Proof Using Rules of Inference

Prove **A ⇒ B, (A ∧ B) ⇒ C, Therefore A ⇒ C**

- **A ⇒ B ≡ ¬ A ∨ B**
- **A ∧ B ⇒ C ≡ ¬ (A ∧ B) ∨ C ≡ ¬ A ∨ ¬ B ∨ C**
- **So ¬ A ∨ B resolves with ¬ A ∨ ¬ B ∨ C deriving ¬ A ∨ C**
- **This is equivalent to A ⇒ C**

Rules of Inference (continued)

- And-Introduction
  $$\frac{\alpha_1, \alpha_2, \ldots, \alpha_n}{\alpha_1 \land \alpha_2 \land \ldots \land \alpha_n}$$
- Or-Introduction
  $$\frac{\alpha_i}{\alpha_1 \lor \alpha_2 \lor \ldots \alpha_i \ldots \lor \alpha_n}$$
- Double Negation
  $$\frac{\neg \neg \alpha}{\alpha}$$
- Unit Resolution (special case of resolution)
  $$\frac{\alpha \lor \beta \quad \neg \beta}{\alpha}$$
  Alternatively:
  $$\frac{\neg \alpha \Rightarrow \beta \quad \neg \beta}{\alpha}$$

Wumpus World KB

- Proposition Symbols for each i,j:
  - Let $P_{i,j}$ be true if there is a pit in square i,j
  - Let $B_{i,j}$ be true if there is a breeze in square i,j
- Sentences in KB
  - "There is no pit in square 1,1"
    $R_1: \neg P_{1,1}$
  - "A square is breezy iff pit in a neighboring square"
    $R_2: B_{1,1} \Leftrightarrow (P_{1,2} \lor P_{2,1})$
    $R_3: B_{1,2} \Leftrightarrow (P_{1,1} \lor P_{1,3} \lor P_{2,2})$
  - "Square 1,1 has no breeze", "Square 1,2 has a breeze"
    $R_4: \neg B_{1,1}$
    $R_5: B_{1,2}$

Inference in Wumpus World

- Apply biconditional elimination to R_2:
  R_6: $(B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$
- Apply AE to R_6:
  R_7: $((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$
- Contrapositive of R_7:
  R_8: $(\neg B_{1,1} \Rightarrow \neg (P_{1,2} \vee P_{2,1}))$
- Modus Ponens with R_8 and R_4 $(\neg B_{1,1})$:
  R_9: $\neg(P_{1,2} \vee P_{2,1})$
- de Morgan:
  R_{10}: $\neg P_{1,2} \wedge \neg P_{2,1}$

Searching for Proofs

- Finding proofs is exactly like finding solutions to search problems.
- Can search forward (forward chaining) to derive goal or search backward (backward chaining) from the goal.
- Searching for proofs is not more efficient than enumerating models, but in many practical cases, it's more efficient because we can ignore irrelevant propositions

Full Resolution Rule Revisited

- Start with Unit Resolution Inference Rule:

$$\frac{\ell_1 \vee \cdots \vee \ell_k, \qquad m}{\ell_1 \vee \cdots \vee \ell_{i-1} \vee \ell_{i+1} \vee \cdots \vee \ell_k}$$

- Full Resolution Rule is a generalization of this rule:

$$\frac{\ell_1 \vee \cdots \vee \ell_k, \qquad m_1 \vee \cdots \vee m_n}{\ell_1 \vee \cdots \vee \ell_{i-1} \vee \ell_{i+1} \vee \cdots \vee \ell_k \vee m_1 \vee \cdots \vee m_{j-1} \vee m_{j+1} \vee \cdots \vee m_n}$$

- For clauses of length two:

$$\frac{\ell_1 \vee \ell_2, \qquad \neg \ell_2 \vee \ell_3}{\ell_1 \vee \ell_3}$$

Resolution Applied to Wumpus World

- At some point we determine the absence of a pit in square 2,2:
  R_{13}: $\neg P_{2,2}$
- Biconditional elimination applied to R_3 followed by modus ponens with R_5:
  R_{15}: $P_{1,1} \vee P_{1,3} \vee P_{2,2}$
- Resolve R_{15} and R_{13}:
  R_{16}: $P_{1,1} \vee P_{1,3}$
- Resolve R_{16} and R_1:
  R_{17}: $P_{1,3}$

Resolution: Complete Inference Procedure

- Any complete search algorithm, applying only the resolution rule, can derive any conclusion entailed by any knowledge base in propositional logic.
- Refutation completeness: Resolution can always be used to either confirm or refute a sentence, but it cannot be used to enumerate true sentences.

Conjunctive Normal Form

- Conjunctive Normal Form is a disjunction of literals.
- Example:

literals

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

clause

CNF Example

Example: (A ∨ B) ⇔ (C ⇒ D)

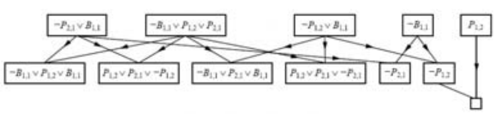
- Eliminate ⇔
  ((A ∨ B) ⇒ (C ⇒ D)) ∧ ((C ⇒ D) ⇒ (A ∨ B)
- Eliminate ⇒
  (¬ (A ∨ B) ∨ (¬ C ∨ D)) ∧ (¬ (¬ C ∨ D) ∨ (A ∨ B))
- Drive in negations
  ((¬ A ∧ ¬ B) ∨ (¬ C ∨ D)) ∧ ((C ∧ ¬ D) ∨ (A ∨ B))
- Distribute
  (¬ A ∨ ¬ C ∨ D) ∧ (¬ B ∨ ¬ C ∨ D) ∧ (C ∨ A ∨ B) ∧ (¬ D ∨ A ∨ B)

Resolution Algorithm

- To show KB ⊨ α, we show (KB ∧ ¬ α) is unsatisfiable.
- This is a proof by contradiction.
- First convert (KB ∧ ¬ α) into CNF.
- Then apply resolution rule to resulting clauses.
- The process continues until:
  - there are no new clauses that can be added (KB does not entail α)
  - two clauses resolve to yield empty clause (KB entails α)

Simple Inference in Wumpus World

- KB = $R_2 \wedge R_4 = (B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})) \wedge \neg B_{1,1}$
- Prove $\neg P_{1,2}$ by adding the negation $P_{1,2}$
- Convert KB ∧ $P_{1,2}$ to CNF



PL-RESOLUTION algorithm

Horn Clauses

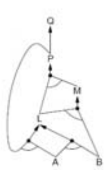
- Real World KB's are often a conjunction of Horn clauses
- Horn clause =
  - proposition symbol; or
  - (conjunction of symbols) ⇒ symbol
- Example:
  C ∧ (B ⇒ A) ∧ (C ∧ D ⇒ B)

$$\frac{\alpha_1, \ldots, \alpha_n, \qquad \alpha_1 \wedge \cdots \wedge \alpha_n \;\Rightarrow\; \beta}{\beta}$$

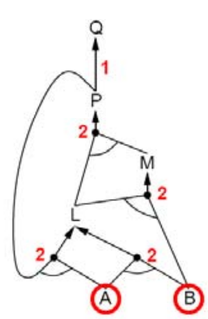
Forward Chaining

- Fire any rule whose premises are satisfied in the KB.
- Add its conclusion to the KB until query is found.

$P \Rightarrow Q$
$L \wedge M \Rightarrow P$
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$
$A \wedge B \Rightarrow L$
A
B

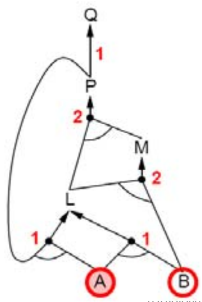


Forward Chaining Example



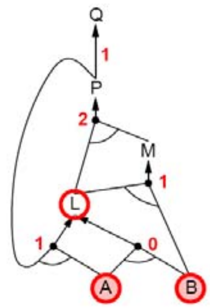
P ⇒ Q
L ∧ M ⇒ P
B ∧ L ⇒ M
A ∧ P ⇒ L
A ∧ B ⇒ L
A
B

Forward Chaining Example



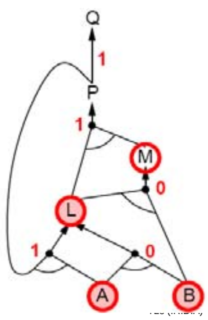
$P \Rightarrow Q$
$L \wedge M \Rightarrow P$
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$
$A \wedge B \Rightarrow L$
A
B

Forward Chaining Example



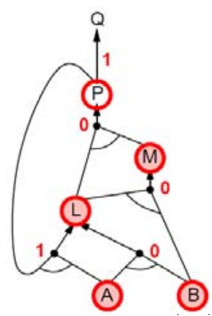
$P \Rightarrow Q$
$L \wedge M \Rightarrow P$
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$
$A \wedge B \Rightarrow L$
A
B

Forward Chaining Example



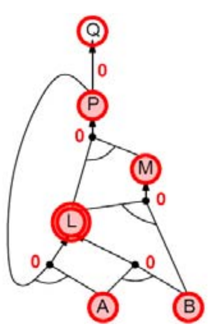
$P \Rightarrow Q$
$L \wedge M \Rightarrow P$
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$
$A \wedge B \Rightarrow L$
A
B

Forward Chaining Example



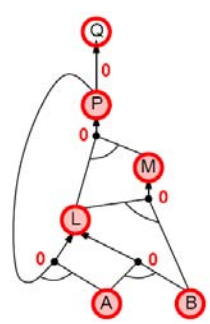
$P \Rightarrow Q$
$L \wedge M \Rightarrow P$
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$
$A \wedge B \Rightarrow L$
A
B

Forward Chaining Example



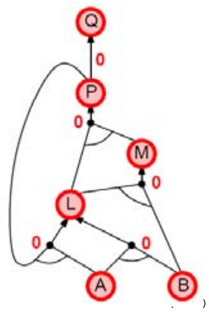
$P \Rightarrow Q$
$L \wedge M \Rightarrow P$
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$
$A \wedge B \Rightarrow L$
A
B

Forward Chaining Example



$P \Rightarrow Q$
$L \wedge M \Rightarrow P$
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$
$A \wedge B \Rightarrow L$
A
B

Forward Chaining Example

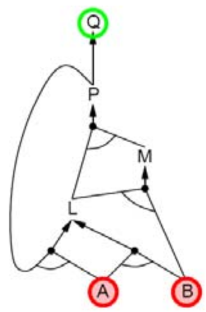


$P \Rightarrow Q$
$L \wedge M \Rightarrow P$
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$
$A \wedge B \Rightarrow L$
A
B

Backward Chaining

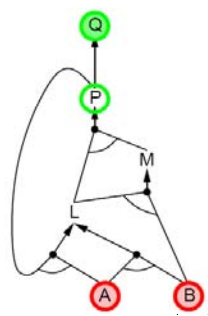
- Motivation: Need goal-directed reasoning in order to keep from getting overwhelmed with irrelevant consequences
- Main idea:
  - Work backwards from query q
  - To prove q:
    - Check if q is known already
    - Prove by backward chaining all premises of some rule concluding q

Backward Chaining Example



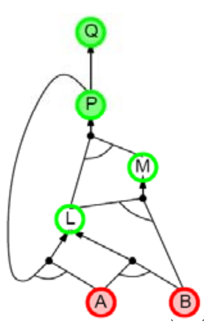
$P \Rightarrow Q$
$L \wedge M \Rightarrow P$
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$
$A \wedge B \Rightarrow L$
A
B

Backward Chaining Example



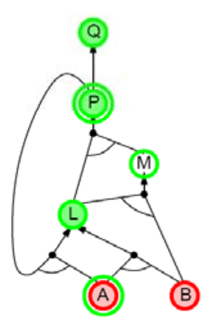
$P \Rightarrow Q$
$L \wedge M \Rightarrow P$
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$
$A \wedge B \Rightarrow L$
A
B

Backward Chaining Example



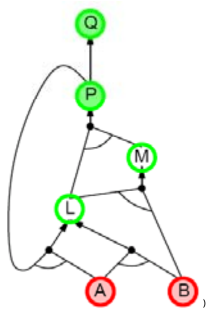
$P \Rightarrow Q$
$L \wedge M \Rightarrow P$
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$
$A \wedge B \Rightarrow L$
A
B

Backward Chaining Example



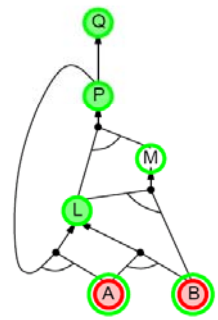
$P \Rightarrow Q$
$L \wedge M \Rightarrow P$
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$
$A \wedge B \Rightarrow L$
A
B

Backward Chaining Example



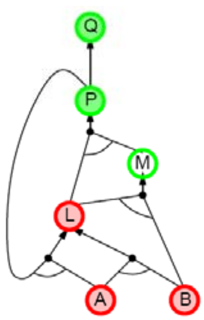
$P \Rightarrow Q$
$L \wedge M \Rightarrow P$
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$
$A \wedge B \Rightarrow L$
A
B

Backward Chaining Example



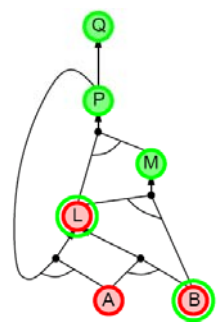
$P \Rightarrow Q$
$L \wedge M \Rightarrow P$
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$
$A \wedge B \Rightarrow L$
A
B

Backward Chaining Example



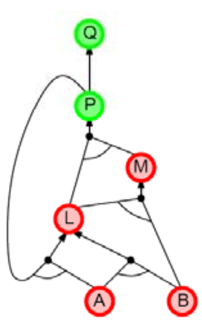
$P \Rightarrow Q$
$L \wedge M \Rightarrow P$
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$
$A \wedge B \Rightarrow L$
A
B

Backward Chaining Example



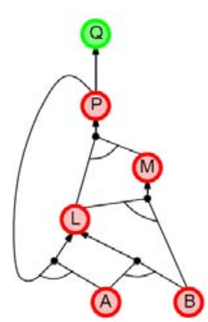
$P \Rightarrow Q$
$L \wedge M \Rightarrow P$
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$
$A \wedge B \Rightarrow L$
A
B

Backward Chaining Example



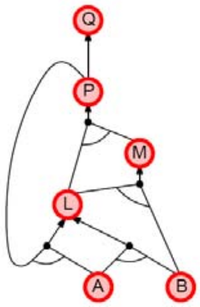
$P \Rightarrow Q$
$L \wedge M \Rightarrow P$
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$
$A \wedge B \Rightarrow L$
A
B

Backward Chaining Example



$P \Rightarrow Q$
$L \wedge M \Rightarrow P$
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$
$A \wedge B \Rightarrow L$
A
B

Backward Chaining Example



P ⇒ Q
L ∧ M ⇒ P
B ∧ L ⇒ M
A ∧ P ⇒ L
A ∧ B ⇒ L
A
B

Forward Chaining vs. Backward Chaining

- FC is data-driven—it may do lots of work irrelevant to the goal
- BC is goal-driven—appropriate for problem-solving