ARTIFICIAL INTELLIGENCE
The Very Idea

Vasant G. Honavar
Dorothy Foehr Huck and J. Lloyd Huck Chair in Biomedical Data Sciences and Artificial Intelligence
Professor of Informatics and Intelligent Systems
Professor of Data Sciences and Artificial Intelligence Methods and Applications Undergraduate Programs
Professor of Informatics, Computer Science, Bioinformatics & Genomics and Neuroscience, and Public Health Sciences Graduate Programs
Director, Artificial Intelligence Research Laboratory
Director, Center for Artificial Intelligence Foundations and Scientific Applications
Director of Strategic Initiatives, Institute for Computational and Data Sciences
Pennsylvania State University

vhonavar@psu.edu
http://faculty.ist.psu.edu/vhonavar
http://ailab.ist.psu.edu

Agents that reason

- With rule-based knowledge representation, we can perform rule-based reasoning
- However, the rules are heuristic in nature, and the conclusions need not be logically sound
- In logic-based systems, conclusions derived from the assertions universally hold, and provably correct (if the underlying inference algorithm is sound)
- Logic based systems can be made more or less expressive based on the type of logic used

Logic and AI

- ``Civilization advances by extending the number of important operations which we can perform without thinking of them. ”

  — Alfred North Whitehead

- ``It is unworthy of excellent men to lose hours like slaves in the labor of calculation which could safely be relegated to anyone else if machines were used’’.

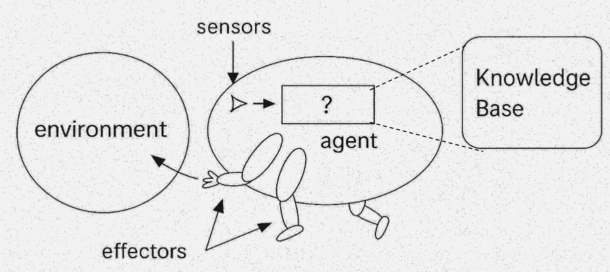
  – Gottfried Leibniz

- ``He that cannot reason is a fool. He that will not is a bigot. He that dare not is a slave.’’

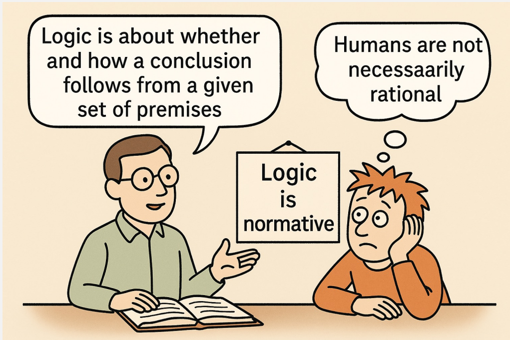
  — Andrew Carnegie

Deliberative Agents

- Can represent and reason with knowledge
- Exhibit logical rationality
- Derive conclusions
  - that logically follow from a given set of facts and
  - only those that logically follow from the facts



Structure of logical arguments



- Logic is normative
  - Logic dictates how a logically rational agent should reason,
  - Not necessarily how individuals reason

Knowledge representation (KR) is a surrogate

A declarative knowledge representation

- Encodes facts that are true in the world into sentences
- Reasoning is performed by manipulating sentences according to sound rules of inference
- The results of inference are sentences that correspond to facts that are true in the world

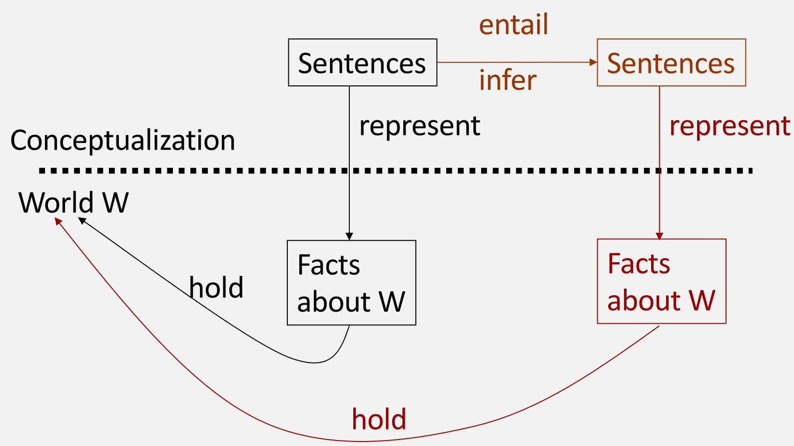
Knowledge representation (KR) is a surrogate

- The correspondence between facts that hold in the world and sentences that describe the world gives meaning to the representation
- Logic allows agents to substitute thinking for acting in the world

Known facts
- The coffee is hot;
- coffee is a liquid;
- a hot liquid will burn your tongue

Inferred fact
- Coffee will burn your tongue

The nature of representation



Logic as a Knowledge Representation Formalism

Logic is a declarative language to:

- Assert sentences representing facts that hold in a real or imagined world *W* (these sentences are given the value true)
- Deduce the true/false values of sentences representing other aspects of *W*
- We shall see that Logical reasoning = computation
- Anticipated by Leibnitz, Hilbert
  - Can all truths be reduced to calculation?
  - Is there an effective procedure for determining whether or not a conclusion is a logical consequence of a set of facts?

(Boolean) Propositional Logic - Syntax

Propositional logic is a formal language with syntax and semantics

- Syntax refers to the structure or form of the sentences
- Semantics refers to the meaning of sentences

Syntax

- Basic units – propositions, e.g., $A, B, Tall, Short, Rich, Poor$ that can be True or False
- Propositions have no intrinsic meaning
- Logical connectives
  - $\wedge$ or logical AND
  - $\vee$ or logical OR
  - $\neg$ or logical negation or NOT
  - $\equiv$ or logical equivalence
  - $\rightarrow$ or logical implication

Propositional Logic - Syntax

Valid sentences include:

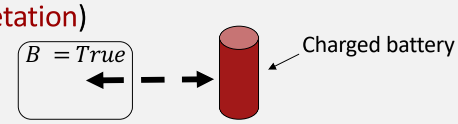
- Basic sentences
  - Propositions, e.g., $A, B, Rich, Poor$ that can be True or False
- Sentences that combine other sentences using logical connectives $\wedge$, $\vee$, $\neg$, $\rightarrow$
- If S_1 and S_2 are sentences, so are
  - $\neg S_1$, $\neg S_2$
  - $S_1 \wedge S_2$
  - $S_1 \vee S_2$
  - $S_1 \rightarrow S_2$
- We use extra-linguistic symbols like parenthesis to disambiguate e.g., $(A \wedge B) \vee (\neg B \wedge C)$

Note that this is a recursive definition

Propositional Logic - Semantics

A proposition (sentence)

- does not have intrinsic meaning
- gets its meaning from correspondence with properties of the world (interpretation)



e.g., proposition B denotes the fact that battery is charged

- There are two possible worlds – one in which battery is charged and one in which it is not
- The proposition B is True or False in a real or imagined world
- B is true in the world in which the battery is charged and false in the world in which it is not charged

Propositional Logic - Semantics

- Meaning of Logical connectives
  - $A \wedge B$ is True if both A and B are True
  - $A \vee B$ is True if A is True or B is True, or both A and B are True
  - $\neg A$ is True if and only if A is False and
  - $\neg A$ is False if and only if A is True,
  - The truth or falsity of a compound sentence is completely determined by the truth or falsity of the components of the sentence and not any other extraneous information

A	B	$A \wedge B$
False	False	False
False	True	False
True	False	False
True	True	True

A	B	$A \vee B$
False	False	False
False	True	True
True	False	True
True	True	True

A	$\neg A$
False	True
True	True

Propositional Logic - Semantics

- Meaning of Logical connectives
  - $A \rightarrow B$ is equivalent to $\neg A \vee B$
    - $A \rightarrow B$ is True whenever A is False or B is True

A	B	$\neg A$	$\neg A \vee B$	$\boldsymbol{A \rightarrow B}$
True	True	False	True	True
True	False	False	False	False
False	True	True	True	True
False	False	True	True	True

- What does $A \rightarrow B$ really mean?

What does $A \rightarrow B$ really mean?

- $A \rightarrow B$ is True whenever A is False or B is True

A	B	$\neg A$	$\neg A \vee B$	$\boldsymbol{A \rightarrow B}$
True	True	False	True	True
True	False	False	False	False
False	True	True	True	True
False	False	True	True	True

- The correct way to think about logical implication is as expression of a promise and not to causal relationship

What does $A \rightarrow B$ really mean?

- Suppose your parents promise you:
  - If you ace your exam, then we buy you a car.
- Let A denote "you ace your exam" and B denote "we buy you a car".
- Now let's see the conditions under which the promise $A \rightarrow B$ holds
  - If A is true (you ace your exam) and B is true (your parents buy you a car) then the promise $A \rightarrow B$ held or remained intact (is true).

A	B	$\neg A$	$\neg A \vee B$	$\boldsymbol{A \rightarrow B}$
True	True	False	True	True
True	False	False	False	False
False	True	True	True	True
False	False	True	True	True

What does $A \rightarrow B$ really mean?

- Suppose your parents promise you:
  - If you ace your exam, then we buy you a car.
- Let A denote "you ace your exam" and B denote "we buy you a car".
- If A is true (you ace your exam) and B is false (your parents didn't buy you a car) then the promise didn't hold so $A \rightarrow B$ is false.

A	B	$\neg A$	$\neg A \vee B$	$\boldsymbol{A \rightarrow B}$
True	True	False	True	True
True	False	False	False	False
False	True	True	True	True
False	False	True	True	True

What does $A \rightarrow B$ really mean?

- Suppose your parents promise you:
  - If you ace your exam, then we buy you a car.
- Let A denote "you ace your exam" and B denote "we buy you a car".
- If A is false (you did not ace your exam) and B is true (your parents bought you a car anyway) then the promise is intact and $A \rightarrow B$ is true
  - Because your parents never said what would happen if you did not ace your exam!
  - Buying a car even if you did not ace the exam does not violate the promise

A	B	$\neg A$	$\neg A \vee B$	$\boldsymbol{A \rightarrow B}$
True	True	False	True	True
True	False	False	False	False
False	True	True	True	True
False	False	True	True	True

What does $A \to B$ really mean?

- Suppose your parents promise you:
  - If you ace your exam, then we buy you a car.
- Let A denote ”you ace your exam” and B denote “we buy you a car”.
- Now let's see the conditions under which the promise $A \to B$ holds
  - If A is false (you did not ace your exam) and B is false (your parents didn't buy you a car) then the promise is intact and $\mathrm{A} \to B$ is true
    - Because you were promised a car if you aced your exam.
    - For the promise to hold, your parents need not buy you a car if you did not ace the exam

A	B	$\neg A$	$\neg A \vee B$	$\boldsymbol{A \to B}$
True	True	False	True	True
True	False	False	False	False
False	True	True	True	True
False	False	True	True	True

Notes about logical implication

- Unlike ∧ and ∨, → is not commutative
  - $A \rightarrow B$ is not the same as $B \rightarrow A$
- The meaning of logical implication is not quite the same as the conversational meaning we assign to implication
  - $Study \rightarrow Pass$
  - If the antecedent is true, → has the usual conversational meaning
  - If antecedent is false, then the implication is true regardless of the truth or falsity of the conclusion
  - In everyday conversation when we say A implies B we often imply a causal relationship between A and B
  - Why? Because $A \rightarrow B \equiv \neg A \vee B$

What can we infer in propositional logic?

- Propositional logic provides the machinery for us to determine
  - Whether or not some conclusion follows logically from a given set of assertions (facts or assumptions)
  - Provided both the conclusion and facts/assumptions are sentences in propositional logic
  - What does it mean for a conclusion to logically follow from a set of assertions?
- We shall see that Reasoning = computation
- Anticipated by Leibnitz, Hilbert
  - Can all truths be reduced to calculation?
  - Is there an effective procedure for determining whether or not a conclusion is a logical consequence of a set of facts?

Model theoretic or Tarskian Semantics

- Consider a logic with only two propositions:
  - $Rich, Poor$
  - denoting "Tom is rich" and "Tom is poor" respectively
- A model M is a subset of the set A of atomic sentences or propositions in the language
- Given this logic, we have

$$A = \{Rich, Poor\}$$

- The models correspond to all possible subsets of A

$$M_0 = \{ \quad \}$$
$$M_1 = \{Rich\}$$
$$M_2 = \{Poor\}$$
$$M_3 = \{Rich, Poor\}$$

- The models denote possible worlds, that is, the possible states of affairs that one can describe or imagine in this logic

Exercise

$$M_0 = \{ \quad \}$$
$$M_1 = \{Rich\}$$
$$M_2 = \{Poor\}$$
$$M_3 = \{Rich, Poor\}$$

Identify the models where the following sentences are true

$$Rich$$
$$Rich \vee Poor$$
$$Rich \wedge Poor$$
$$Rich \Rightarrow \neg Poor$$
$$\neg Rich \vee \neg Poor$$

Exercise

$$M_0 = \{\quad\}$$
$$M_1 = \{Rich\}$$
$$M_2 = \{Poor\}$$
$$M_3 = \{Rich, Poor\}$$

Identify the models where the following sentences are true

$$\begin{array}{rl} Rich & \text{is } True \text{ in} \quad M_1, M_3 \\ Rich \vee Poor & \text{is } True \text{ in} \quad M_1, M_2, M_3 \\ Rich \wedge Poor & \text{is } True \text{ in} \quad M_3 \\ Rich \Rightarrow \neg Poor & \text{is } True \text{ in} \quad M_0, M_1, M_2 \\ \neg Rich \vee \neg Poor & \text{is } True \text{ in} \quad M_0, M_1, M_2 \end{array}$$

Model theoretic or Tarskian Semantics

- Given the set of atomic sentences $\{Rich, Poor\}$, the possible worlds are

$$M_0 = \{\ \ \}, M_1 = \{Rich\}, M_2 = \{Poor\}, M_3 = \{Rich, Poor\}$$

- By a model *M* we mean the state of affairs in the world in which
  - every atomic sentence that is in *M* is *true* and
  - every atomic sentence that is not in *M* is *false*
- In M_0 Tom is neither rich nor poor
- In M_1 Tom is rich
- In M_2 Tom is poor
- In M_3 Tom is both rich and poor

Model theoretic or Tarskian Semantics

- We have

$$A = \{Rich, Poor\}$$

- The possible worlds are

$$M_0 = \{\ \}$$
$$M_1 = \{Rich\}$$
$$M_2 = \{Poor\}$$
$$M_3 = \{Rich, Poor\}$$

- In M_0 Tom is neither rich nor poor
- In M_1 Tom is rich
- In M_2 Tom is poor
- In M_3 Tom is both rich and poor
- How could this be?

Model theoretic or Tarskian Semantics

- The possible worlds are
  $M_0 = \{\quad\}, M_1 = \{Rich\}, M_2 = \{Poor\}, M_3 = \{Rich, Poor\}$
- By a model *M* we mean the state of affairs in the world in which
  - every atomic sentence that is in *M* is *true* and
  - every atomic sentence that is not in *M* is *false*
- In M_0 Tom is neither rich nor poor $Rich$ is False and $Poor$ is False
- In M_1 Tom is rich: $Rich$ is True, $Poor$ is False
- In M_2 Tom is poor: $Poor$ is True, $Rich$ is False
- In M_3 Tom is both rich and poor: $Rich$ is True and $Poor$ is True
- How could this be?
- Because the propositions $Rich, Poor$ have no intrinsic meaning!
- They get their meaning from correspondence with the states of the world

Model theoretic or Tarskian Semantics

- The possible worlds are

$$M_0 = \{\quad\}, M_1 = \{Rich\}, M_2 = \{Poor\}, M_3 = \{Rich, Poor\}$$

- What if we wanted to ensure that the meaning of $Rich$ and *Poor* are mutually exclusive?
  - We must assert that Tom cannot be both rich and poor: $\neg(Rich \wedge Poor)$
- What if we wanted to assert that Tom has to be either rich nor poor?
  - We must assert that: $Rich \vee Poor$
- Hence, if we want to ensure that our logical assertions align with their intuitive meanings, we restrict their meanings by the additional assertions $\neg(Rich \wedge Poor)$, $Rich \vee Poor$

Some laws of propositional logic

- **Commutative law.**

  $p \wedge q \equiv q \wedge p$

  $p \vee q \equiv q \vee p$

- **Associative law**

  $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$

  $(p \vee q) \vee r \equiv p \vee (q \vee r)$

- **Distributive law**

  $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$

  $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$

- **De Morgan's Laws**

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

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

- **Identity**

  $p \wedge \top \equiv p$

  $p \vee \bot \equiv p$

- **Tautology**

  $p \vee \neg p \equiv \top$

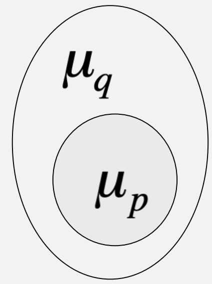
- **Contradiction**

  $p \wedge \neg p \equiv \bot$

Logical entailment

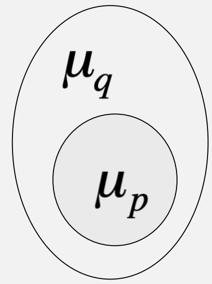
- What does it mean for a conclusion to logically follow from a given set of assertions?
- First, note that any set of assertions can be combined using $\wedge$ to obtain a single equivalent sentence
  - \`\`$A \wedge B$ is True and $\neg C \vee D$ is True '' is equivalent to
  - $(A \wedge B) \wedge (\neg C \vee D)$ is True
- Hence, it suffices to consider what it means for one sentence, say q, to logically follow from another, say, p

Logical Entailment



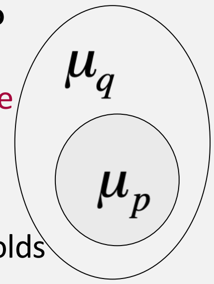
- What does it mean for q to logically follow from p?
- We say that p entails q (written as $p \vDash q$) if q holds in every model in which p holds
- Suppose
  - μ_q is the set of models in with q holds
  - μ_p is the set of models in with p holds
- Then $p \vDash q$ if it is the case that $\mu_p \subseteq \mu_q$

Logical Entailment



- What does it mean for q to logically follow from p?
- We say that p entails q (written as $p \vDash q$) if q holds in every model in which p holds, that is, $\mu_p \subseteq \mu_q$
    - Note that entailment $\vDash$ is akin to what we conversationally mean by implication which is different from logical implication ($\rightarrow$)

What does it mean to be logically rational?



- Infer only those conclusions from one's knowledge base that are sanctioned by logical entailment
- To find out if $p \vDash q$, an agent can
  - Enumerate μ_p, the set of models in which p holds
  - Enumerate μ_q, the set of models in which q holds
  - Check if $\mu_p \subseteq \mu_q$

What does it mean to be logically rational?

- Infer only those conclusions from one's knowledge base that are sanctioned by logical entailment
  - Suppose you know that being human implies being mortal
  - Then you find out that you are human
  - Is it rational for you to conclude that you are mortal?

What does it mean to be logically rational?

- Suppose you know that being human implies being mortal
- Then you find out that you are human
- Is it rational for you to conclude that you are mortal?

Let us construct a logic to find out

- Let H denote being human
- Let M denote being mortal
- Knowledge base: $H \rightarrow M, H$
- We need to check whether $H \wedge (H \rightarrow M) \vDash M$

How can we tell if $H \wedge (H \rightarrow M) \vDash M$?

Enumerate the models

$M_0 = \{\ \}, M_1 = \{H\ \}\ M_2 = \{M\}, M_3 = \{H, M\}$

Let p be the sentence $H \wedge (H \rightarrow M)$ and q be the sentence M

μ_H = the set of models in which H holds = $\{M_1, M_3\}$

$\mu_{H \rightarrow M}$ = the set of models in which $H \rightarrow M$ holds

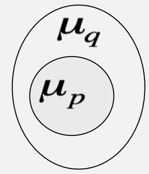
= the set of models in which $\neg H \vee M$ holds

$= \mu_{\neg H} \cup \mu_M = \{M_0, M_2\} \cup \{M_2, M_3\} = \{M_0, M_2, M_3\}$

$\mu_{H \wedge (H \rightarrow M)} = \mu_H \cap \mu_{H \rightarrow M} = \{M_1, M_3\} \cap \{M_0, M_2, M_3\} = M_3 = \mu_p$

$\mu_M = \{M_2, M_3\} = \mu_q$

How can we tell if $H \wedge (H \rightarrow M) \vDash M$?



Enumerate the models

$M_0 = \{\,\}, M_1 = \{H\}\ M_2 = \{M\}, M_3 = \{H, M\}$

Let p be the sentence $H \wedge (H \rightarrow M)$ and q be the sentence M

$\mu_p = M_3$

$\mu_q = \{M_2, M_3\}$

Clearly, $\mu_p \subseteq \mu_q$

Hence $p \vDash q$

Therefore $H \wedge (H \rightarrow M) \vDash M$

That is, given H and $H \rightarrow M$, it is logically rational to conclude M

What did we just do?

We just proved that $H \wedge (H \rightarrow M) \vDash M$

- Note that we never really made use of the fact that H and M denote being human and being mortal respectively
- So long as our knowledge base has two sentences of the form α and $\alpha \rightarrow \beta$ hold, logic permits us to conclude that β holds as well
- This yields a logically sound rule of inference that we can mechanically apply to any knowledge base:

  Given $\alpha, \alpha \rightarrow \beta$, infer β

- This is the rule called *Modus Ponens* that Aristotle had introduced but without solid justification which we now have, thanks to Tarski

Logical Rationality

- A logical agent A with a knowledge base KB_A is justified in inferring q if it is the case that $KB_A \models q$
- How can the agent A decide whether in fact $KB_A \models q$?
  - Model checking
    - Enumerate μ_{KB_A} i.e., all the models in which KB_A holds
    - Enumerate μ_q i.e., all the models in which q holds
    - Check whether $\mu_{KB_A} \subseteq \mu_q$
  - Inference algorithm based on inference rules
    - We saw one such inference rule that is provably sound:
      - Given $\alpha, \alpha \rightarrow \beta$, infer β

Proving theorems using Modus Ponens

- Suppose you are told that
  - “Today is Tuesday” and
  - “If Today is Tuesday, You have AI100 lecture today”
- Can a machine prove that “You have AI100 lecture today”?

Proving theorems using Modus Ponens

- Suppose you are told that
  - "Today is Tuesday" and
  - "If Today is Tuesday, You have AI100 lecture today"
- Can a machine prove that "You have AI100 lecture today"?
- Let P stand for "Today is Tuesday "
- Let Q stand for "You have AI100 lecture today"
- We are told
  - $P \rightarrow Q$
  - P
- Modus ponens tells us that $\{P, P \rightarrow Q\} \vDash Q$
- That concludes the proof.

Reasoning with Modus Ponens

- Given:
  - P
  - $P \rightarrow Q$
  - $Q \rightarrow R$
- Prove R
- Modus ponens (MP) tells us: $\{P, P \rightarrow Q\} \vDash Q$
- What did we do?
  - Used Modus Ponens to construct a proof
    - A sequence of applications of one or more inference rules – in this case, MP

P
$P \rightarrow Q$
$Q \rightarrow R$

MP

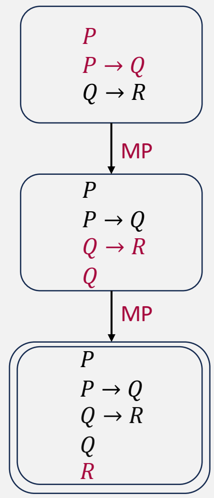
P
$P \rightarrow Q$
$Q \rightarrow R$
Q

MP

P
$P \rightarrow Q$
$Q \rightarrow R$
Q
R

Reasoning with Modus Ponens

- We can see that reasoning or theorem proving can be reduced to state space search
- Start state
  - The given set of logical assertions
- Actions or operators
  - Applications of an inference rule such as Modus Ponens
- Goal state
  - A set of logical assertions that includes the desired conclusion
- Proof (of the conclusion)
  - A sequence of actions that take you from the start state to a goal state



Search for proofs: inference

- An inference rule $\{\alpha, \beta\} \vdash \gamma$ consists of
  - two sentence patterns α and β called the premises and
  - one sentence pattern γ called the consequent
- Note the difference between $\vDash$ and $\vdash$
  - $\vDash$ is a semantic notion
  - $\vdash$ is a syntactic pattern matching procedure
- If α and β match two sentences of KB then
  - the corresponding sentence of the form γ can be inferred according to the rule
- Given one or more sound inference rules and a knowledge base *KB*
  - inference is the process of successively applying inference rules to *KB*
  - Each rule application adds its consequent to the *KB*

Generalized Modus Ponens

$$\{\alpha, \beta\} \vdash \gamma$$

$$\{\, \overbrace{p \rightarrow q}\,, p \,\} \vdash q$$

$$\frac{\begin{array}{c} a_1 \wedge a_2 \wedge a_3 \cdots \wedge a_{i-1} \wedge a_i \wedge a_{i+1} \cdots a_m \rightarrow q \\ a_i \end{array}}{a_1 \wedge a_2 \wedge a_3 \cdots \wedge a_{i-1} \wedge a_{i+1} \cdots a_m \rightarrow q}$$

Generalized Modus Ponens

$$\{\alpha, \beta\} \vdash \gamma$$

$$\{\overbrace{p \rightarrow q}, p\} \vdash q$$

Generalized Modus Ponens

$$\frac{\begin{array}{c} a_1 \wedge a_2 \wedge a_3 \cdots \wedge a_{i-1} \wedge a_i \wedge a_{i+1} \cdots a_m \rightarrow q \\ b_1 \wedge b_2 \wedge \cdots \wedge b_n \rightarrow a_i \end{array}}{a_1 \wedge a_2 \wedge a_3 \cdots \wedge a_{i-1} \wedge a_{i+1} \cdots a_m \wedge b_1 \wedge b_2 \wedge \cdots \wedge b_n \rightarrow q}$$

Reasoning with inference rules

- Forward chaining
  - Start with the given facts
  - Recursively apply the rule(s) of inference to derive the conclusion
- Backward chaining
  - Start with the conclusion you want to prove
  - Apply the inference rules backwards until you derive the given facts

Application of generalized modus ponens

KB:

$$BatteryOK \wedge BulbsOK \rightarrow HeadlightsWork$$

$$BatteryOK \wedge StarterOK \wedge \neg EmptyGasTank \rightarrow EngineStarts$$

$$EngineStarts \wedge \neg FlatTire \wedge HeadlightsWork \rightarrow CarOK$$

$$BatteryOK, BulbsOK, StarterOK, \neg EmptyGasTank, \neg FlatTire$$

Query:

$CarOK?$

Example: Forward-chaining using Modus Ponens

$$BatteryOK \wedge BulbsOK \rightarrow HeadlightsWork$$
$$BatteryOK, BulbsOK$$

$$HeadlightsWork$$

$$BatteryOK \wedge StarterOK \wedge \neg EmptyGasTank \rightarrow EngineStarts$$
$$BatteryOK, StarterOK, \neg EmptyGasTank$$

$$EngineStarts$$

$$EngineStarts \wedge \neg FlatTire \wedge HeadlightsWork \rightarrow CarOK$$
$$EngineStarts, \neg FlatTire, HeadlightsWork$$

$$CarOK$$

Example: Backward chaining to prove $CarOK$

KB:

$$BatteryOK \wedge BulbsOK \rightarrow HeadlightsWork$$

$$BatteryOK \wedge StarterOK \wedge \neg EmptyGasTank \rightarrow EngineStarts$$

$$EngineStarts \wedge \neg FlatTire \wedge HeadlightsWork \rightarrow CarOK$$

$$BatteryOK, BulbsOK, StarterOK, \neg EmptyGasTank, \neg FlatTire$$

Query:

$CarOK?$

Example: Backward chaining to prove $CarOK$

$CarOK$

$EngineStarts \wedge \neg FlatTire \wedge HeadlightsWork$

-

$EngineStarts \wedge \neg FlatTire \wedge HeadlightsWork$

$BatteryOK \wedge BulbsOK \rightarrow HeadlightsWork$

$EngineStarts \wedge \neg FlatTire \wedge BatteryOK \wedge BulbsOK$

$BatteryOK \wedge StarterOK \wedge \neg EmptyGasTank \rightarrow EngineStarts$

$\neg FlatTire \wedge BatteryOK \wedge BulbsOK \wedge StarterOK \wedge \neg EmptyGasTank$

$BatteryOK, BulbsOK, StarterOK , \neg EmptyGasTank, \neg FlatTire$

Forward chaining

Work forwards from premises to conclusion

- To prove q by forward chaining,
  - Put given facts into working memory
  - Apply applicable rules generate next states
  - Repeat until you end up with a state that contains q

Avoid loops

- Check if new fact is already in working memory
- If so, do not add it to working memory

We can ensure this by not applying a rule if it has been applied already

Forward chaining

- Appy any rule whose premises are satisfied in the *KB*
- Add its conclusion to the *KB*, until the conclusion is added to KB

1. $P \rightarrow Q$
2. $L \wedge M \rightarrow P$
3. $B \wedge L \rightarrow M$
4. $A \wedge P \rightarrow L$
5. $A \wedge B \rightarrow L$

A, B

↓ 5

A, B, L

↓ 3

A, B, L, M

↓ 2

A, B, L, M, P

↓

A, B, L, M, P, Q

Backward chaining

Work backwards from the goal *q*

- To prove *q* by backward chaining,
  - check if *q* is known already, or
  - prove by BC all premises of some rule concluding *q*

Avoid loops

- Check if new subgoal is already on the goal list

Avoid repeated work

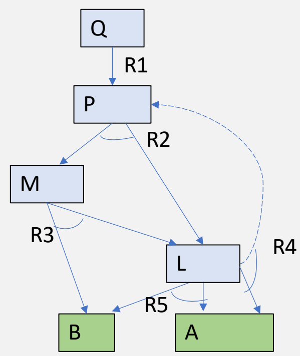
- check if new subgoal has already been proved true, or
- has already failed

Backward chaining

AND-OR search working backwards from the goal

- To prove *goal* by backward chaining,
  - check if *goal* is known already, or
  - Recursively prove by BC all premises of some rule concluding *goal*

R1. $P \rightarrow Q$
R2. $L \wedge M \rightarrow P$
R3. $B \wedge L \rightarrow M$
R4. $A \wedge P \rightarrow L$
R5. $A \wedge B \rightarrow L$



Partial solutions
Q
P
M&L
B&L
A&B, ~~A&P~~
■

Forward chaining (FC) vs. backward chaining (BC)

- FC is data-driven, automatic, unconscious processing
- May do lots of work that is irrelevant to the conclusion
- BC is goal-driven, appropriate for problem-solving,
- Complexity of BC can be much less than linear in size of KB

Not all inference rules are sound

- *Modus ponens*

  $\{\alpha \rightarrow \beta, \alpha\} \vdash \beta$

Modus ponens derives only inferences sanctioned by entailment

Modus ponens is sound

- *Loony tunes*

  $Friday \vdash \beta$

Loony tunes can derive inferences that are not sanctioned by entailment

Loony tunes is not sound

Soundness and Completeness of an inference rule $\vdash$

- We write $p \vdash q$ to denote that that p can be inferred from q using the inference rule $\vdash$

An inference rule $\vdash$ is said to be

- Sound if whenever $p \vdash q$, it is also the case that $p \models q$
  - That is, the inference rule yields only those conclusions that are sanctioned by entailment
- Complete if whenever $p \models q$, it is also the case that $p \vdash q$
  - That is, the inference rule can be used to derive all the conclusions that are sanctioned by entailment
- Ideally, we want inference rules that are both sound and complete
- Logical rationality requires inference rules that are sound
- We may settle for sound inference rules that are not complete

Soundness and Completeness of Modus Ponens

- We can show that *modus ponens* is sound, but *not* complete
  - unless the KB is *Horn* i.e., the KB can be written as a collection of sentences of the form
- $a_1 \wedge a_2 \wedge a_3 \ldots a_{i-1} \wedge a_i \wedge a_{i+1} \wedge a_{i+2} \ldots \wedge a_m \rightarrow b$
- Where each a_i and b are atomic sentences

Unsound inference rules are not necessarily useless!

Abduction (Charles Peirce) is not sound, but useful in diagnostic reasoning or hypothesis generation

$$\frac{\begin{array}{c} p \rightarrow q \\ q \end{array}}{p}$$

$$\frac{\begin{array}{c} BlockedArtery \rightarrow HeartAttack \\ HeartAttack \end{array}}{BlockedArtery}$$

Validity and satisfiability, equivalence

- A sentence is valid if it is true in *all* models,
  - e.g., *True*, $A \vee \neg A$, $A \rightarrow A$, $(A \wedge (A \rightarrow B)) \rightarrow B$
- A sentence is satisfiable if it is true in *some* model
  - e.g., $A \vee B, C$
- A sentence is unsatisfiable if it is true in *no* models
  - e.g., $A \wedge \neg A$
- A useful result for proof by contradiction
  - $KB \models s$ if and only if $(KB \wedge \neg s)$ is unsatisfiable
- Two sentences are logically equivalent iff they are true in same set of models or $\alpha \equiv \beta$ iff $\alpha \models \beta$ and $\beta \models \alpha$.

Constructing proofs

- Finding proofs can be cast as a search problem
- Search can involve
  - forward chaining to derive *goal* from *KB*
  - or backward chaining from the *goal* to facts
- Searching for proofs
  - Involves repeated application of applicable inference rules.
  - Can be more efficient than enumerating models
- Propositional logic is monotonic
  - Inference steps can only add inferred facts
  - An inferred fact once added is never deleted
  - A theorem once proven can never be disproven (barring error in proof)

Soundness and Completeness of a logical reasoner

- An logical reasoner starts with the KB and applies applicable inference rules until the desired conclusion is reached
- A logical reasoner is sound if it uses a sound inference rule
- An inference algorithm is complete if
  - It uses a complete inference rule and
  - a complete search procedure, that is one that is guaranteed to find a solution if one exists

Completeness of Modus Ponens for Propositional Logic

- Modus Ponens is not complete for Propositional Logic
- Suppose that all classes at some university meet either Mon/Wed/Fri or Tue/Thu.
- The AI course meets at 4 PM in the afternoon
- Jane has volleyball practice Thursdays and Fridays at that time.
- Can Jane take AI?

$$1.\ MWFAI4pm \ \vee TRAI4pm$$
$$2\ TRAI4pm \wedge JaneBusyR4pm \rightarrow JaneConflictAI$$
$$3.\ MWFAI4pm \wedge JaneBusy4pm \rightarrow JaneConflictAI$$
$$4.\ JaneBusyR4pm$$
$$5.\ JaneBusyF4pm$$

Completeness of Modus Ponens for Propositional Logic

- Modus Ponens is not complete for Propositional Logic
- Can Jane take AI?

$$1. MWFAI4pm \lor TRAI4pm$$
$$2\ TRAI4pm \land JaneBusyR4pm \rightarrow JaneConflictAI$$
$$3. MWFAI4pm \land JaneBusyF4pm \rightarrow JaneConflictAI$$
$$4. JaneBusyR4pm$$
$$5. JaneBusyF4pm$$

- Of course not!
- Try proving this using Modus Ponens
- You can't!
- Why?

Completeness of Modus Ponens for Propositional Logic

$$1. MWFAI4pm \lor TRAI4pm$$
$$2\ TRAI4pm \land JaneBusyR4pm \rightarrow JaneConflictAI$$
$$3. MWFAI4pm \land JaneBusyF4pm \rightarrow JaneConflictAI$$
$$4. JaneBusyR4pm$$
$$5. JaneBusyF4pm$$

We can use Modus Ponens to establish

$$2\&4: TRAI4pm \rightarrow JaneConflictAI$$
$$3\&4: MWFAI4pm \rightarrow JaneConflictAI$$

But Modus Ponens can't take us further to conclude $JaneConflictAI$!

- Modus Ponens is not complete for Propositional Logic (except in the restricted case when the KB is Horn)
- However, we can generalize Modus Ponens to obtain a sound and complete inference rule for Propositional Logic

Soundness and Completeness of Forward Chaining

- An inference algorithm starts with the KB and applies applicable inference rules until the desired conclusion is reached
- An inference algorithm is sound if it uses a sound inference rule
- An inference algorithm is complete if
  - It uses a complete inference rule and
  - a complete search procedure
- Forward chaining using Modus Ponens is sound and complete for Horn knowledge bases (i.e., knowledge bases that contain only Horn clauses)

Forward vs. backward chaining

- FC is data-driven, automatic, unconscious processing,
  - Akin to day dreaming…
- May do lots of work that is irrelevant to the goal
- BC is goal-driven, appropriate for problem-solving
  - e.g., Where are my keys? How do I get into a PhD program?
- The run time of FC is linear in the size of the KB.
- The run time of BC can be, in practice, much less than linear in size of *KB*

Resolution principle

Any set of propositional logic sentences can be expressed in a standard form, CNF, conjunction of disjunctions

Resolution is sound and complete for propositional *KB*

Given

$\neg a_1 \lor \ldots \lor \neg a_{i-1} \lor \neg a_i \lor \neg a_{i+1} \lor \ldots \lor \neg a_M \lor q_1 \lor q_2 \ldots \lor q_N$

$b_1 \lor \ldots \lor b_L \lor c_1 \lor \ldots \lor c_{j-1} \lor c_j \lor c_{j+1} \ldots \lor c_K$

If $a_i = c_j$ then we can conclude:

$\neg a_1 \ldots a_{i-1} \lor \neg a_{i+1} \lor \ldots \lor \neg a_M \lor q_1 \lor q_2 \ldots \lor q_N \lor b_1 \lor \ldots \lor b_L \lor c_1 \lor \ldots \lor c_{j-1} \lor c_{j+1} \ldots \lor c_K$

Transformation to Clause Form (CNF)

Example: (A ∨ ¬B) →(C ∧ D)

1. Eliminate → using $p \rightarrow q \equiv \neg p \vee q$ to get
   ¬(A ∨ ¬B) ∨ (C ∧ D)
2. Reduce scope of ¬ using De Morgan's laws (¬A ∧ B) ∨ (C ∧ D)
3. Distribute ∨ over ∧
   (¬A ∨ (C ∧ D)) ∧ (B ∨ (C ∧ D)) to get
   (¬A ∨ C) ∧ (¬A ∨ D) ∧ (B ∨ C) ∧ (B ∨ D)
4. Break up the conjunction into individual sentences to get a set of clauses or conjunction of disjunctions (CNF):
   {¬A ∨ C , ¬A ∨ D , B ∨ C , B ∨ D}

De Morgan's Laws

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

Distributive law

$$p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$$
$$p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$$

Exercise: Transformation to Clause Form (CNF)

Example: $(A \vee B \vee \neg C) \rightarrow (D \wedge E \wedge F)$

1. Eliminate $\rightarrow$ using $p \rightarrow q \equiv \neg p \vee q$ to get
2. Reduce scope of $\neg$ using De Morgan's laws to get
3. Distribute $\vee$ over $\wedge$ to get

Break up the conjunction into individual sentences to get a set of clauses or conjunction of disjunctions (CNF):

De Morgan's Laws

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

Distributive law

$$p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$$
$$p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$$

Exercise: Transformation to Clause Form (CNF)

Example: $(A \vee B \vee \neg C) \rightarrow (D \wedge E \wedge F)$

1. Eliminate $\rightarrow$ using $p \rightarrow q \equiv \neg p \vee q$ to get
   $\neg(A \vee B \vee \neg C) \vee (D \wedge E \wedge F)$
2. Reduce scope of $\neg$ using De Morgan's laws: $(\neg A \wedge \neg B \wedge C) \vee (D \wedge E \wedge F)$
3. Distribute $\vee$ over $\wedge$ to get
   $(\neg A \vee (D \wedge E \wedge F)) \wedge (\neg B \vee (D \wedge E \wedge F)) \wedge (C \vee (D \wedge E \wedge F))$
   which is equivalent to
   $(\neg A \vee D) \wedge (\neg A \vee E) \wedge (\neg A \vee F) \wedge$
   $(\neg B \vee D) \wedge (\neg B \vee E) \wedge (\neg B \vee F) \wedge$
   $(C \vee D) \wedge (C \vee E) \wedge (C \vee F)$

4. Break up the conjunction into individual sentences to get a set of clauses or conjunction of disjunctions (CNF):
   $\{\neg A \vee D, \neg A \vee E, \neg A \vee F, \neg B \vee D, \neg B \vee E, \neg B \vee F, C \vee D, C \vee E, C \vee F\}$

Applying resolution

- Transform *KB* into an equivalent Conjunctive normal form (CNF)
  - Each sentence in *KB* is a disjunction of literals (elementary propositions) or their negations using known logical equivalences
  - *KB* is a conjunction of disjunctions
- Any propositional KB can be converted into CNF

Proof

- The proof of a sentence α from a set of sentences *KB* is the derivation of α obtained through a series of applications of sound inference rules to *KB*
- *KB* ⊨ α if and only if {*KB*,¬α} is unsatisfiable

  {*KB*,¬α} ⊨ contradiction ($T \rightarrow F$, ■, empty sentence)
- Proving α from *KB* is equivalent to deriving a contradiction from *KB* augmented with the negation of α
- The above strategy is called resolution by refutation
- Automated theorem provers for propositional logic use resolution by refutation to construct proofs for CNF formulas with thousands of variables and clauses

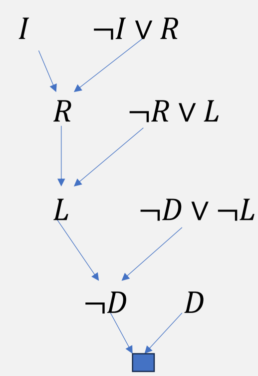
Resolution by refutation

- To establish that a conclusion follows from a set of premises
  - Negate the conclusion and add it to the premises
  - Convert the resulting set of propositional logic sentences into clause normal form (CNF)
  - Start with the clause(s) resulting from the negated conclusion and successively resolve them with other clauses until an empty clause results
  - An empty clause corresponds to a logical contradiction in propositional logic
  - If the premises together with a negated conclusion yield a contradiction, the conclusion must follow from the premises

Example: Applying resolution

- Given KB: I, D, $\neg R \vee L$, $\neg D \vee \neg L$
- To prove: $I \wedge \neg R$
- Negated theorem: $\neg(I \wedge \neg R) \equiv \neg I \vee R$

Proof



Example

- Show that R follows from the KB: $P, P \rightarrow Q, P \wedge Q \rightarrow R$
- Convert the given sentences into CNF and negate the theorem

1. P
2. $P \rightarrow Q \equiv \neg P \vee Q$
3. $P \wedge Q \rightarrow R \equiv (\neg P \vee \neg Q) \vee R \equiv \neg P \vee \neg Q \vee R$
4. $\neg R$ (negated theorem)

Resolve 3,4 to get 5. $\neg P \vee \neg Q$

Resolve 1 and 5 to get 6. $\neg Q$

Resolve 6 and 2 to get 7. $\neg P$

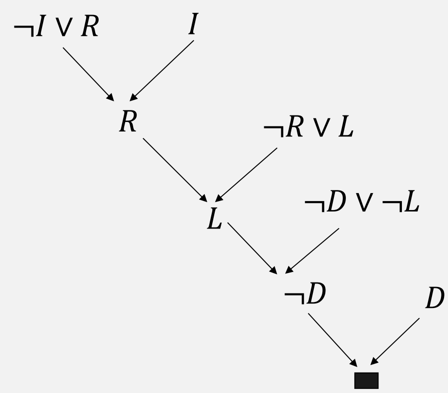
Resolve 7 and 1 to get ■ (empty clause), thus proving that the KB entails R

Some useful tricks

- Set of support strategy
  - Start with the negated theorem
  - At each step, use a clause derived from resolving the negated theorem or one of its descendants with some other clause
  - Negated theorem must play a role in a resolution by refutation proof
- Unit clause strategy
  - All things being equal, choose a clause with a single literal to resolve with a clause from the set of support
  - Helps us get to the empty clause (contradiction) quicker

Set of Support Strategy in Action

Given: $I, D, \neg R \vee L, \neg D \vee \neg L$
To prove: $I \wedge \neg R$.
Negated theorem: $\neg I \vee R$



Example: Applying Resolution

- Suppose that all classes at some university meet either Mon/Wed/Fri or Tue/Thu.
- The AI course meets at 4 PM in the afternoon
- Jane has volleyball practice Thursdays and Fridays at 4pm.
- Does Jane have a conflict with AI? Assume not.

$$1.\ MWFAI4pm \ \vee TRAI4pm$$
$$2\ TRAI4pm \wedge JaneBusyR4pm \rightarrow JaneConflictAI$$
$$3.\ MWFAI4pm \wedge JaneBusyF4pm \rightarrow JaneConflictAI$$
$$4.\ JaneBusyR4pm$$
$$5.\ JaneBusyF4pm$$
$$6.\ \neg JaneConflictAI$$

Example: Applying Resolution

- Transform to CNF and add negated theorem:

$$1. MWFAI4pm \lor TRAI4pm$$
$$2\ \neg TRAI4pm \lor \neg JaneBusyR4pm \lor JaneConflictAI$$
$$3. \neg MWFAI4pm \lor \neg JaneBusyF4pm \lor JaneConflictAI$$
$$4. JaneBusyR4pm$$
$$5. JaneBusyF4pm$$
$$6. \neg JaneConflictAI$$

Example: Applying Resolution

1. $MWFAI4pm \vee TRAI4pm$
2 $\neg TRAI4pm \vee \neg JaneBusyR4pm \vee JaneConflictAI$
3. $\neg MWFAI4pm \vee \neg JaneBusyF4pm \vee JaneConflictAI$
4. $JaneBusyR4pm$
5. $JaneBusyF4pm$
6. $\neg JaneConflictAI$

Resolution by refutation proof

2: $\neg TRAI4pm \vee \neg JaneBusyR4pm \vee JaneConflictAI$ 6: $\neg JaneConflictAI$

7: $\neg TRAI4pm \vee \neg JaneBusyR4pm$ 1: $MWFAI4pm \vee TRAI4pm$

8: $\neg JaneBusyR4pm \vee MWFAI4pm$ 4: $JaneBusyR4pm$

9: $MWFAI4pm$ 3: $\neg MWFAI4pm \vee \neg JaneBusyF4pm \vee JaneConflictAI$

10: $\neg JaneBusyF4pm \vee JaneConflictAI$ 5: $JaneBusyF4pm$

11: $JaneConflictAI$ 6: $\neg JaneConflictAI$

■

Exercise: Prove $CarOK$ using resolution

KB:

$$BatteryOK \wedge BulbsOK \rightarrow HeadlightsWork$$

$$BatteryOK \wedge StarterOK \wedge \neg EmptyGasTank \rightarrow EngineStarts$$

$$EngineStarts \wedge \neg FlatTire \wedge HeadlightsWork \rightarrow CarOK$$

$$BatteryOK, BulbsOK, StarterOK, \neg EmptyGasTank, \neg FlatTire$$

Theorem:

$CarOK$?

Beyond propositional logic

- Propositional logic
  - assumes the world can be represented using propositions
  - has limited expressive power
- First-order predicate logic (like natural language)
  - assumes the world contains
    - Objects:
      - people, flowers, houses, numbers, students,
    - Relations:
      - red, round, prime, brother of, bigger than, part of
    - Functions:
      - father of, best friend of, plus, …
  - Allows one to talk about some or all of the objects

Sentences in first-order logic

- Variables and quantifiers for all (∀) and there exists (∃) can be used to express statements that describe all or some individuals
  - All humans are mortal

$$\forall x \; Human(x) \rightarrow Mortal(x)$$

  - There exist students in AI100 who are smart

$$\exists x \; Student(x, AI100) \wedge Smart(x)$$

  - There are no pigs that fly

$$\neg\exists x \; [Pig(x) \wedge Flies(x)]$$

First order logic

- We can define the semantics of first order logic by noting that
  - The semantics of for all (∀) and there exists (∃) naturally follows from the semantics of logical AND and logical OR
  - We can extend the inference rules to predicate logic
    - $\forall x\ Human(x) \rightarrow Mortal(x)$
    - $Human(Joe)$
    - Prove $Mortal(Joe)$

Beyond propositional logic

- First order logic allows quantification over variables
- Modal logics of knowledge support reasoning about knowledge like
  - Sam knows that Joe did not do his homework
  - John knows that everyone knows that Joe did not do his homework
  - All of these logics generalize propositional logic so most
- Detailed discussion of first order and higher order logics is beyond the scope of this course