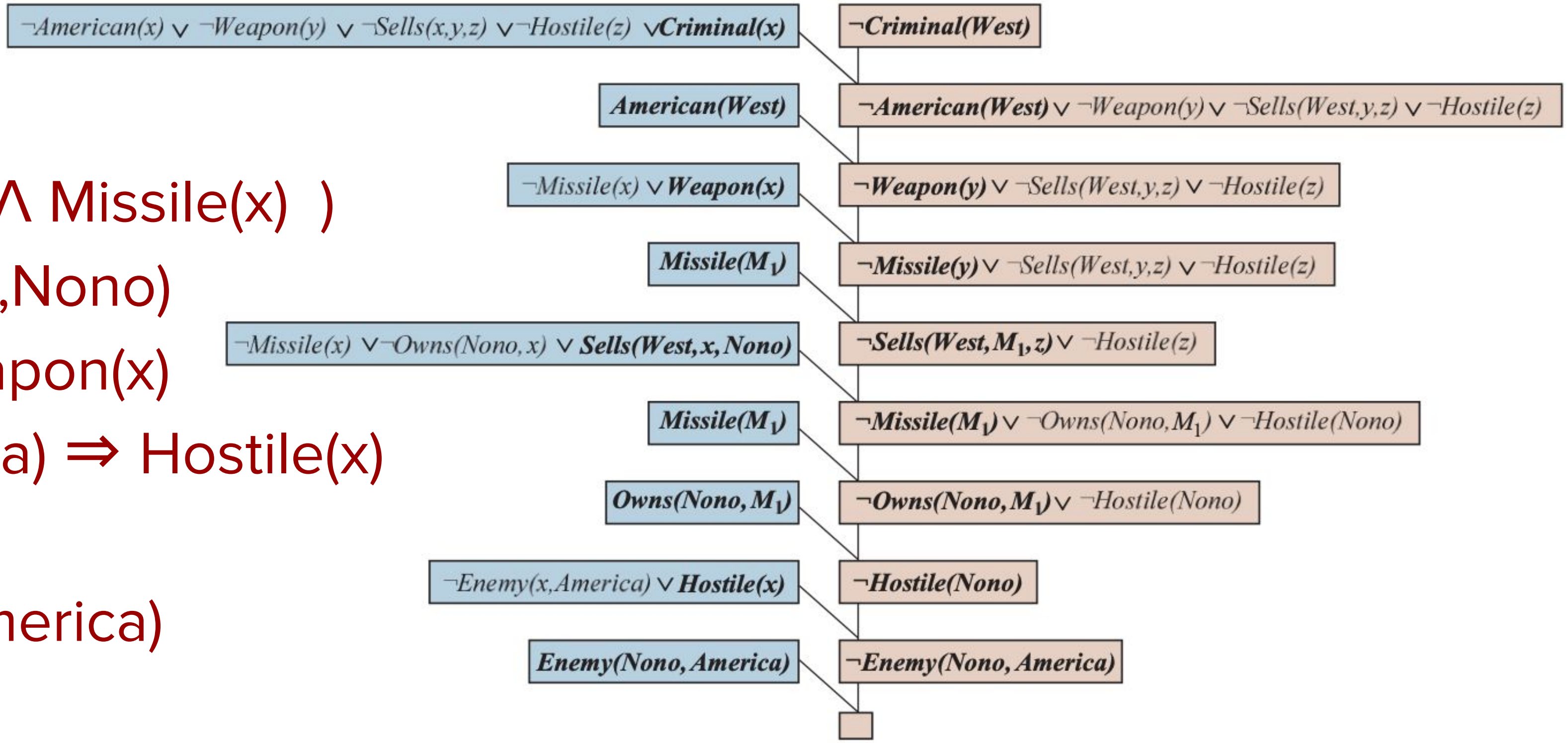
Example Resolution Proofs

- (American(x) ∧ Weapon(y) ∧ Sells(x, y, z) ∧ Hostile(z)) ⇒ Criminal(x)
- Owns(Nono, M1)
- Missile(M1)
- (Owns(Nono, x) ∧ Missile(x)) ⇒ Sells(West,x,Nono)
- Missile(x) ⇒ Weapon(x)
- Enemy(x, America) ⇒ Hostile(x)
- American(West)
- Enemy(Nono, America)



Example Resolution Proofs

- $KB = \{ \quad P(u) \vee P(F(u)), \quad \neg P(v) \vee P(F(w)) \quad \}$

  Does $KB \models \exists x \; P(x) \wedge P(F(x))$

1. $P(u) \vee P(F(u))$
2. $\neg P(v) \vee P(F(w))$
3. $\neg P(x) \vee \neg P(F(x))$

4. $P(u) \vee P(F(w))$	BRR for (1) and (2) with $\{ v / F(u)\}$
5. $P(F(w))$	Factoring (4) with $\{v / w\}$
6. $\neg P(F(F(z)))$	BRR for (3) and (5) with $\{x / F(w), w/z\}$
7. []	BRR for (6) and (7) with $\{z/ F(w)\}$

Example Resolution Proof

- Everyone who loves all animals is loved by someone.
- Anyone who kills an animal is loved by no one.
- Jack loves all animals.
- Either Jack or Curiosity killed the cat, who is named Tuna.

- Did Curiosity kill the cat?

$$Animal(F(x)) \lor Loves(G(x), x)$$
$$\neg Loves(x, F(x)) \lor Loves(G(x), x)$$
$$\neg Loves(y, x) \lor \neg Animal(z) \lor \neg Kills(x, z)$$
$$\neg Animal(x) \lor Loves(Jack, x)$$
$$Kills(Jack, Tuna) \lor Kills(Curiosity, Tuna)$$
$$Cat(Tuna)$$
$$\neg Cat(x) \lor Animal(x)$$
$$\neg Kills(Curiosity, Tuna)$$

Example Resolution Proof

$Animal(F(x)) \vee Loves(G(x), x)$

$\neg Loves(x, F(x)) \vee Loves(G(x), x)$

$\neg Loves(y, x) \vee \neg Animal(z) \vee \neg Kills(x, z)$

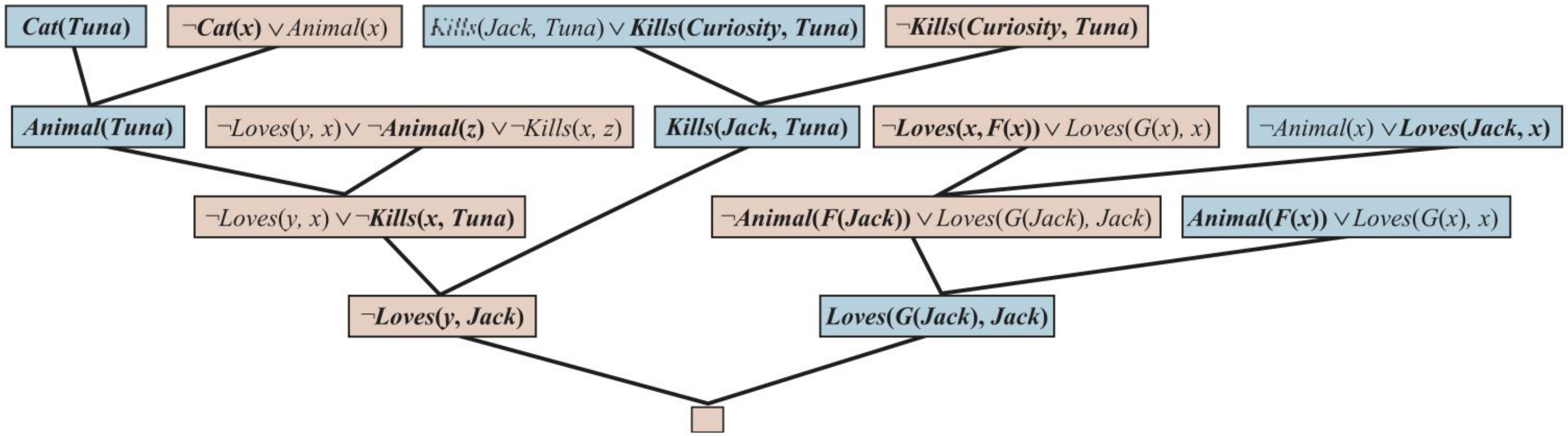
$\neg Animal(x) \vee Loves(Jack, x)$

$Kills(Jack, Tuna) \vee Kills(Curiosity, Tuna)$

$Cat(Tuna)$

$\neg Cat(x) \vee Animal(x)$

$\neg Kills(Curiosity, Tuna)$



Equality

- Some axioms about equality
  - $\forall x \ x = x$
  - $\forall x \forall y \ (x = y) \Rightarrow (y = x)$
  - $\forall x \forall y \forall z \ (x = y \wedge y = z) \Rightarrow (y = x)$
  - $\forall x \forall y \ (x = y) \Rightarrow (\boldsymbol{\alpha} \Leftrightarrow \mathrm{SUB}(x, y, \boldsymbol{\alpha}))$
    - where SUB(t, t', $\boldsymbol{\alpha}$) is obtained by replacing some/all occurrences of the term t in $\boldsymbol{\alpha}$ by t'

- BRR + FR + Above axioms are complete for First Order logic with equality
  - But above axioms produce lot of unwanted conclusions
  - Can we have rules instead?

Rules for Equality : Demodulation

- Example:
  - x = G(y) P(F(u)) V Q(u, z)
    ——————————————————————————
  - P(F(G(y))) V Q(x, z) or P(F(G(y))) V Q(G(y), z)

  - F(x) = G(y) P(F(u)) V Q(u, z)
    ——————————————————————
  - P(G(y)) V Q(x, z)
    can we have: P(G(y)) V Q(..., z) where ... is something other than x?

- s = t m_1 V ... Vm_n
  —————————————————————————————————————— (Demodulation Rule)
  SUB(SUBST(θ,s), SUBST(θ,s), SUBST(θ, m_1 V ... Vm_n))

  such that r occurs in some m_i and Unify(s,r) = θ

Rules for Equality : Paramodulation

- Example:
  - P(G(x)) V x = G(y)   P(F(u)) V Q(u, z)
    ——————————————————————
    P(G(x)) V P(F(G(y))) V Q(x, z)
  - R(H(F(x), y)) V F(x) = G(y)   P(F(u)) V Q(u, z)
    ——————————————————————
  - R(H(F(x), y)) V P(G(y)) V Q(x, z)

- $$\frac{\ell_1 \vee \ldots \vee \ell_k \vee s = t \qquad m_1 \vee \ldots \vee m_n}{\mathrm{SUBST}(\theta, \ell_1 \vee \ldots \vee \ell_k) \vee m'_1 \vee \ldots \vee m'_n} \text{ (Paramodulation Rule)}$$

such that r occurs in some m_i and Unify(s,r) = θ and m_j' is of the form SUB(SUBST(θ,s), SUBST(θ,t), SUBST(θ, m_j))

Rules for Equality : Paramodulation

- R(y) V F(A, y) = y P(F(x,B), x) V Q(x)

  —---

  R(B) V P(B, A) V Q(A)

- P(F(x)) V F(x) = G(y) P(F(u))

  —---

  P(F(x)) V P(G(y))

- P(F(u)) V F(x) = G(y) P(F(u))

  —---

  P(F(x)) V P(G(y))

- With Paramodulation we have complete inference procedure for First Order Logic with equality

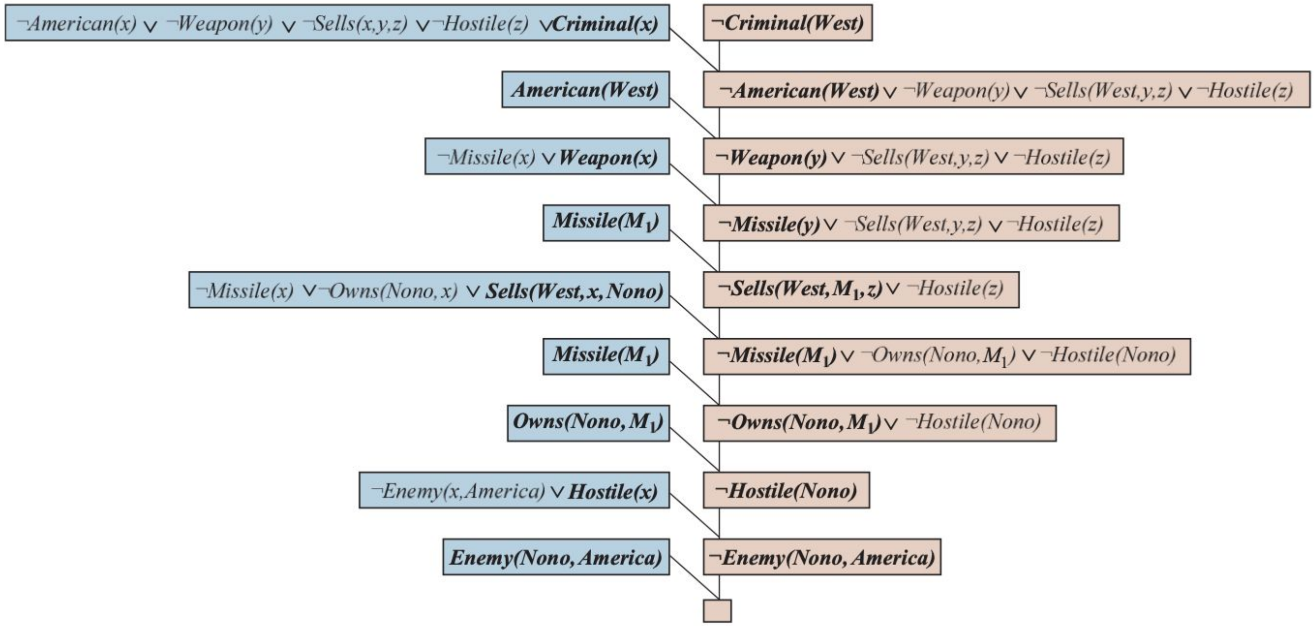
Resolution Strategies in First Order Logic

- Unit Preference: Resolve unit clauses first
  - Produces shorter clauses
  - Unit Resolution is complete for Horn Clauses

- Set of Support : Pick a set of clauses and try to ensure every resolution rule using one clause from this set.
  - New inferred clauses are also added to S
  - Reduces the search space
  - To ensure completeness, ensure that the remaining sentences are satisfiable
  - Candidate : S = clauses of the negated goal (assuming that KB is satisfiable)

Resolution Strategies in First Order Logic

- Input resolution: Every rule uses at least one clause from the input (either KB or negated goal)
  - The space of proof trees are smaller
  - Not complete in general, but complete for Horn clauses (think of Forward chaining)



Resolution Strategies in First Order Logic

- Subsumption: Eliminate all clauses that are subsumed
  - No point in adding P(John) if we already have P(x) in the KB
  - No point in adding P(x) V Q(x) if we already have P(x) in the KB

- Learning : Train a ML model to pick which clauses to resolve
  - DeepHOL

Inference in First Order Logic

- Theorem provers:
  - Used not only in AI but also in Verification and Synthesis

  - Cannot be fully automated in general
    - We do not even have a fully automated theorem provers for real world scenarios
    - Typically it is guided by humans: Occasionally it asks what rule should I use next?

- Decidable Fragments:
  - We cannot have a terminating algorithm for checking $KB \models \alpha$ in general
  - Can we say if we restrict the form of formulas then we can have such an algorithm?
    - Example : Definite clauses
  - Such restrictions are called decidable fragments
    - If we use only two variables / Guarded Fragments / Unary predicates and No functions / ...

Knowledge Representation

- For Wumpus world, whether we used propositional Logic or First order Logic did not matter much

- But how should Google store its Knowledge Base?

- How to handle the facts about worlds? (Irrespective of the logic used)
    - Represent / Modify / Infer / ....
    - Real world data involves Events, Time, Objects, Beliefs ..
    - Ontological Engineering : How to represent these abstract concepts

Exceptions

- Should we tell the knowledge base :

  "Birds fly" / "All tomatoes are red"

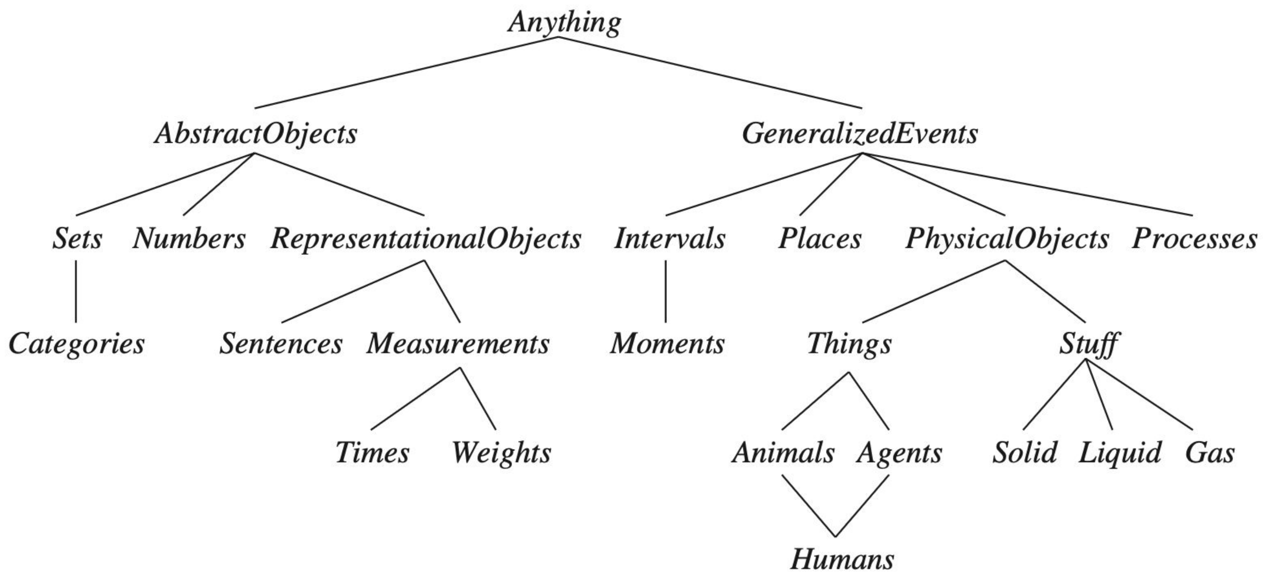
- Real world data always has exceptions

- First we will see how to represent General Knowledge
  - Deal with exceptions later

Upper Ontology

- General framework of concepts : Similar to Class design in Object Oriented Programming and/or Schema in Relational Databases

- Allows us to make simplifying assumptions



General Purpose Ontology

- We can keep generalizing Upper Ontologies
  - Making it cover more and more scenarios
  - Does it mean we have have a “most general Ontology” that can model everything?

- A general purpose Ontology should be applicable in almost all domains

- We need this if we want our AI systems to know about “everything”

- No success till now. Though there are some attempts:
  - DBPedia : Built from Wikipedia data
  - TextRunner : Built by reading large corpus of Web pages
  - OpenMind : Commonsense Knowledge

  - Google knowledge Graph uses semi-structured content from Wikipedia
    - Has over 70 billion facts (in 2014)
    - Answers about 33% of Google searches