Last time

- Unification: Given $\alpha$ and $\beta$, find $\theta$ such that $\alpha \theta = \beta \theta$
- Most general unifier (MGU)

- Generalised Modus Ponens

$$p'_1, p'_2, \ldots, p'_n \ (p_1 \land p_2 \land \cdots \land p_n \Rightarrow q) \ \frac{q}{q \theta}$$

when $p'_i \theta \equiv p_i \theta \ \forall i$
Outline

- Forward chaining
- Backward chaining
- Resolution
function FOL-FC-Ask(\(\text{KB}, \alpha\)) returns a substitution or false

 inputs: \(\text{KB}\), the knowledge base, a set of first-order definite clauses
 \(\alpha\), the query, an atomic sentence

 local variables: \(\text{new}\), the new sentences inferred on each iteration

 repeat until \(\text{new}\) is empty
   \(\text{new} \leftarrow \{\}\)
   for each rule in \(\text{KB}\) do
     \((p_1 \land \ldots \land p_n \Rightarrow q) \leftarrow \text{STANDARDIZE-VARIABLES}(\text{rule})
     \) for each \(\theta\) such that \(\text{SUBST}(\theta, p_1 \land \ldots \land p_n) = \text{SUBST}(\theta, p'_1 \land \ldots \land p'_n)\)
       for some \(p'_1, \ldots, p'_n\) in \(\text{KB}\)
       \(q' \leftarrow \text{SUBST}(\theta, q)\)
       if \(q'\) does not unify with some sentence already in \(\text{KB}\) or \(\text{new}\) then
         add \(q'\) to \(\text{new}\)
         \(\phi \leftarrow \text{UNIFY}(q', \alpha)\)
         if \(\phi\) is not fail then return \(\phi\)
       add \(\text{new}\) to \(\text{KB}\)
 return false

Pattern-matching

Facts irrelevant to the goal can be generated
American(x) \land Weapon(y) \land Sells(x, y, z) \land Hostile(z) \Rightarrow Criminal(x)

Ows(Nono, M_1) and Missile(M_1)

Missile(x) \land Owns(Nono, x) \Rightarrow Sells(West, x, Nono)

Missile(x) \Rightarrow Weapon(x)

Enemy(x, America) \Rightarrow Hostile(x)

American(West) and Enemy(Nono, America)
Forward chaining proof

\[ \text{American}(x) \land \text{Weapon}(y) \land \text{Sells}(x,y,z) \land \text{Hostile}(z) \Rightarrow \text{Criminal}(x) \]
\[ \text{Owns}(\text{Nono}, M_1) \text{ and } \text{Missile}(M_1) \]
\[ \text{Missile}(x) \land \text{Owns}(\text{Nono}, x) \Rightarrow \text{Sells}(\text{West}, x, \text{Nono}) \]
\[ \text{Missile}(x) \Rightarrow \text{Weapon}(x) \]
\[ \text{Enemy}(x, \text{America}) \Rightarrow \text{Hostile}(x) \]
\[ \text{American}(\text{West}) \text{ and } \text{Enemy}(\text{Nono}, \text{America}) \]
Forward chaining proof

\[ \text{American}(x) \land \text{Weapon}(y) \land \text{Sells}(x, y, z) \land \text{Hostile}(z) \Rightarrow \text{Criminal}(x) \]
\[ \text{Owns}(\text{Nono}, M_1) \text{ and } \text{Missile}(M_1) \]
\[ \text{Missile}(x) \land \text{Owns}(\text{Nono}, x) \Rightarrow \text{Sells}(\text{West}, x, \text{Nono}) \]
\[ \text{Missile}(x) \Rightarrow \text{Weapon}(x) \]
\[ \text{Enemy}(x, \text{America}) \Rightarrow \text{Hostile}(x) \]
\[ \text{American}(\text{West}) \text{ and } \text{Enemy}(\text{Nono, America}) \]
Forward chaining proof

\[ \text{American}(x) \land \text{Weapon}(y) \land \text{Sells}(x, y, z) \land \text{Hostile}(z) \Rightarrow \text{Criminal}(x) \]

\[ \text{Owns}(\text{Nono}, M_1) \land \text{Missile}(M_1) \]

\[ \text{Missile}(x) \land \text{Owns}(\text{Nono}, x) \Rightarrow \text{Sells}(\text{West}, x, \text{Nono}) \]

\[ \text{Missile}(x) \Rightarrow \text{Weapon}(x) \]

\[ \text{Enemy}(x, \text{America}) \Rightarrow \text{Hostile}(x) \]

\[ \text{American}(\text{West}) \land \text{Enemy}(\text{Nono}, \text{America}) \]
Properties of forward chaining II

- Sound and complete for first-order definite clauses
  - Definite clause = exactly one positive literal.
- **Datalog** = first-order definite clauses + no functions
  - FC terminates for Datalog in finite number of iterations
- May not terminate in general if $\alpha$ is **not** entailed
- This is unavoidable: entailment with definite clauses is **semi-decidable**
Incremental forward chaining: no need to match a rule on iteration $k$ if a premise wasn’t added on iteration $k - 1$. 

$\Rightarrow$ match each rule whose premise contains a newly added positive literal.

Matching itself can be expensive:

*Database indexing* allows $O(1)$ retrieval of known facts. 

- e.g., query $\text{Missile}(x)$ retrieves $\text{Missile}(M1)$

Forward chaining is widely used in *deductive databases*
Efficiency of forward chaining II

- **Pattern Matching**
  - For each $\theta$ such that
    \[ \text{SUBST}(\theta, p_1 \land \cdots \land p_n) = \text{SUBST}(\theta, p'_1 \land \cdots \land p'_n) \]
    for some $p'_1, \ldots, p'_n$ in KB
  - Finding all possible unifiers can be very expensive

- **Example**
  - \textit{Missile}(x) \land \textit{Owns}(Nono, x) \Rightarrow \textit{Sells}(West, x, Nono)
  - Can find each object owned by Nono in constant time and then check if it is admissible
  - But what if Nono owns many objects but very few missiles?
  - Conjunct Ordering: Better (cost-wise) to find all missiles first and then check whether they are owned by Nono
  - Optimal ordering is NP-hard. Heuristics available: e.g. MRV from CSP if each conjunct is viewed as a constraint on its vars
Every finite domain CSP can be expressed as a single definite clause + ground facts
Colourable is inferred iff the CSP has a solution
CSPs include 3SAT as a special case, hence matching is NP-hard
Backward chaining algorithm

A function that returns multiple times, each time giving one possible result

\[
\text{SUBST}(\text{COMPOSE}(\theta_1, \theta_2), p) = \text{SUBST}(\theta_2, \text{SUBST}(\theta_1, p))
\]

10/02/2017 Michael Herrmann Inf2D 12
Backward chaining example

\[\text{American}(x) \land \text{Weapon}(y) \land \text{Sells}(x, y, z) \land \text{Hostile}(z) \Rightarrow \text{Criminal}(x)\]
\[\text{Owns}(\text{Nono}, M_1) \land \text{Missile}(M_1)\]
\[\text{Missile}(x) \land \text{Owns}(\text{Nono}, x) \Rightarrow \text{Sells}(\text{West}, x, \text{Nono})\]
\[\text{Missile}(x) \Rightarrow \text{Weapon}(x)\]
\[\text{Enemy}(x, \text{America}) \Rightarrow \text{Hostile}(x)\]
\[\text{American}(\text{West}) \land \text{Enemy}(\text{Nono}, \text{America})\]

\[
\text{Criminal(West)}
\]
Backward chaining example

\[\text{American}(x) \land \text{Weapon}(y) \land \text{Sells}(x, y, z) \land \text{Hostile}(z) \Rightarrow \text{Criminal}(x)\]

\[\text{Owns}(\text{Nono}, M_1) \text{ and } \text{Missile}(M_1)\]

\[\text{Missile}(x) \land \text{Owns}(\text{Nono}, x) \Rightarrow \text{Sells}(\text{West}, x, \text{Nono})\]

\[\text{Missile}(x) \Rightarrow \text{Weapon}(x)\]

\[\text{Enemy}(x, \text{America}) \Rightarrow \text{Hostile}(x)\]

\[\text{American}(\text{West}) \text{ and } \text{Enemy}(\text{Nono}, \text{America})\]
Backward chaining example

\[ \text{American}(x) \land \text{Weapon}(y) \land \text{Sells}(x,y,z) \land \text{Hostile}(z) \Rightarrow \text{Criminal}(x) \]
\[ \text{Owns}(\text{Nono}, M_1) \text{ and } \text{Missile}(M_1) \]
\[ \text{Missile}(x) \land \text{Owns}(\text{Nono}, x) \Rightarrow \text{Sells}(\text{West}, x, \text{Nono}) \]
\[ \text{Missile}(x) \Rightarrow \text{Weapon}(x) \]
\[ \text{Enemy}(x, \text{America}) \Rightarrow \text{Hostile}(x) \]
\[ \text{American}(\text{West}) \text{ and } \text{Enemy}(\text{Nono}, \text{America}) \]
Backward chaining example

\[
\text{American}(x) \land \text{Weapon}(y) \land \text{Sells}(x, y, z) \land \text{Hostile}(z) \Rightarrow \text{Criminal}(x) \\
\text{Owns}(\text{Nono}, M_1) \text{ and } \text{Missile}(M_1) \\
\text{Missile}(x) \land \text{Owns}(\text{Nono}, x) \Rightarrow \text{Sells}(\text{West}, x, \text{Nono}) \\
\text{Missile}(x) \Rightarrow \text{Weapon}(x) \\
\text{Enemy}(x, \text{America}) \Rightarrow \text{Hostile}(x) \\
\text{American}(\text{West}) \text{ and } \text{Enemy}(\text{Nono, America})
\]
Backward chaining example

\[
\text{American}(x) \land \text{Weapon}(y) \land \text{Sells}(x, y, z) \land \text{Hostile}(z) \Rightarrow \text{Criminal}(x)
\]
\[
\text{Owns}(\text{Nono}, M_1) \land \text{Missile}(M_1)
\]
\[
\text{Missile}(x) \land \text{Owns}(\text{Nono}, x) \Rightarrow \text{Sells}(\text{West}, x, \text{Nono})
\]
\[
\text{Missile}(x) \Rightarrow \text{Weapon}(x)
\]
\[
\text{Enemy}(x, \text{America}) \Rightarrow \text{Hostile}(x)
\]
\[
\text{American}(\text{West}) \text{ and } \text{Enemy}(\text{Nono, America})
\]
Backward chaining example

\[
\text{American}(x) \land \text{Weapon}(y) \land \text{Sells}(x, y, z) \land \text{Hostile}(z) \Rightarrow \text{Criminal}(x) \\
\text{Owns}(\text{Nono}, M_1) \text{ and } \text{Missile}(M_1) \\
\text{Missile}(x) \land \text{Owns}(\text{Nono}, x) \Rightarrow \text{Sells}(\text{West}, x, \text{Nono}) \\
\text{Missile}(x) \Rightarrow \text{Weapon}(x) \\
\text{Enemy}(x, \text{America}) \Rightarrow \text{Hostile}(x) \\
\text{American}(\text{West}) \text{ and } \text{Enemy}(\text{Nono, America})
\]
Backward chaining example

\[\text{American}(x) \land \text{Weapon}(y) \land \text{Sells}(x, y, z) \land \text{Hostile}(z) \Rightarrow \text{Criminal}(x)\]
\[\text{Owns}(\text{Nono}, \text{M}_1) \text{ and } \text{Missile}(\text{M}_1)\]
\[\text{Missile}(x) \land \text{Owns}(\text{Nono}, x) \Rightarrow \text{Sells}(\text{West}, x, \text{Nono})\]
\[\text{Missile}(x) \Rightarrow \text{Weapon}(x)\]
\[\text{Enemy}(x, \text{America}) \Rightarrow \text{Hostile}(x)\]
\[\text{American}(\text{West}) \text{ and } \text{Enemy}(\text{Nono, America})\]
Properties of backward chaining

- Depth-first recursive proof search: space is linear in size of proof
- Incomplete due to infinite loops
  - partial fix by checking current goal against every goal on stack
- Inefficient due to repeated subgoals (both success and failure)
  - fix using caching of previous results (extra space)
- Widely used for logic programming
A method for telling whether a propositional formula is satisfiable and for proving that a first-order formula is unsatisfiable.

Yields a complete inference algorithm

If a set of clauses is unsatisfiable, then the resolution closure of those clauses contains the empty clause (propositional logic)
Ground Binary Resolution

\[
\frac{C \lor P \quad D \lor \neg P}{C \lor D}
\]

Soundness:

\[C \lor P \text{ iff } \neg C \Rightarrow P\]
\[D \lor \neg P \text{ iff } P \Rightarrow D\]

Therefore, \( \neg C \Rightarrow D \),

which is equivalent to \( C \lor D \)

Note: if both \( C \) and \( D \) are empty then resolution deduces the empty clause, i.e. false.
Non-Ground Binary Resolution

\[
\frac{C \vee P}{(C \vee D) \theta} \quad \frac{D \vee \neg P'}{(C \vee D) \theta}
\]

where \( \theta \) is the MGU of \( P \) and \( P' \)

- The two clauses are assumed to be standardized apart so that they share no variables.

- **Soundness**: apply \( \theta \) to premises then appeal to ground binary resolution.

\[
\frac{C \theta \vee P \theta}{(C \theta \vee D \theta)} \quad \frac{D \theta \vee \neg P \theta}{(C \theta \vee D \theta)}
\]
$$\neg Rich(x) \lor Unhappy(x) \quad Rich(Ken)$$

Unhappy(Ken)

with $\theta = \{x/\text{Ken}\}$
Factoring

\[
\frac{A \lor B}{A \lor A} \quad \frac{A \lor \neg B}{A \lor A}
\]

More generally

\[
C \lor P_1 \lor \ldots \lor P_m \quad \frac{(C \lor P_1) \theta}{(C \lor P_1) \theta}
\]

where \( \theta \) is the MGU of the \( P_i \);

**Soundness**: by universal instantiation and deletion of duplicates.
$C \lor P_1 \lor \ldots \lor \neg P_m \quad D \lor \neg P'_1 \lor \ldots \lor \neg P'_n$ 

$(C \lor D) \theta$

where $\theta$ is MGU of all $P_i$ and $P'_j$

- **Soundness**: by combination of factoring and binary resolution.
- To prove $\alpha$: apply resolution steps to $CNF(KB \land \neg \alpha)$
  - complete for FOL, if full resolution or binary resolution + factoring is used
Conversion to CNF

Example:
Everyone who loves all animals is loved by someone:
\(\forall x. [\forall y. \text{Animal}(y) \implies \text{Loves}(x, y)] \implies [\exists y. \text{Loves}(y, x)]\)

1. Eliminate all biconditionals and implications
\(\forall x. \neg [\forall y. \neg \text{Animal}(y) \lor \text{Loves}(x, y)] \lor [\exists y. \text{Loves}(y, x)]\)

2. Move \(\neg\) inwards, use: \(\neg \forall x. p \equiv \exists x. \neg p, \neg \exists x. p \equiv \forall x. \neg p,\) etc.
\(\forall x. [\exists y. \neg (\neg \text{Animal}(y) \lor \text{Loves}(x, y))] \lor [\exists y. \text{Loves}(y, x)]\)
\(\forall x. [\exists y. \neg \text{Animal}(y) \land \neg \text{Loves}(x, y)] \lor [\exists y. \text{Loves}(y, x)]\)
\(\forall x. [\exists y. \text{Animal}(y) \land \neg \text{Loves}(x, y)] \lor [\exists y. \text{Loves}(y, x)]\)
1. Standardize variables apart: each quantifier should use a different one
   \[ \forall x. [\exists y. \text{Animal}(y) \land \neg \text{Loves}(x, y)] \lor [\exists z. \text{Loves}(z, x)] \]

2. Skolemize: a more general form of existential instantiation
   Each existential variable is replaced by a Skolem function of the enclosing universally quantified variables:\(^1\):
   \[ \forall x. [\text{Animal}(F(x)) \land \neg \text{Loves}(x, F(x))] \lor \text{Loves}(G(x), x) \]

3. Drop universal quantifiers:
   \[ [\text{Animal}(F(x)) \land \neg \text{Loves}(x, F(x))] \lor \text{Loves}(G(x), x) \]

4. Distribute \( \lor \) over \( \land \):
   \[ [\text{Animal}(F(x)) \lor \text{Loves}(G(x), x)] \land [\neg \text{Loves}(x, F(x)) \lor \text{Loves}(G(x), x)] \]

---

^1) No enclosing universal quantifier? Just replace with Skolem constant, i.e. a function with no argument.
'West' Clauses

\[\neg \text{American}(x) \lor \neg \text{Weapon}(y) \lor \neg \text{Sells}(x, y, z) \lor \neg \text{Hostile}(z) \lor \text{Criminal}(x)\]

\[\text{Owns}(\text{Nono}, M_1) \text{ and } \text{Missile}(M_1)\]

\[\neg \text{Missile}(x) \lor \neg \text{Owns}(\text{Nono}, x) \lor \text{Sells}(\text{West}, x, \text{Nono})\]

\[\neg \text{Missile}(x) \lor \text{Weapon}(x)\]

\[\neg \text{Enemy}(x, \text{America}) \lor \text{Hostile}(x)\]

\[\text{American}(\text{West}) \text{ and } \text{Enemy}(\text{Nono}, \text{America})\]
Resolution proof: definite clauses

\neg American(x) \lor \neg Weapon(y) \lor \neg Sells(x,y,z) \lor \neg Hostile(z) \lor Criminal(x)

\neg Criminal(West)

American(West)

\neg American(West) \lor \neg Weapon(y) \lor \neg Sells(West,y,z) \lor \neg Hostile(z)

\neg Weapon(y) \lor \neg Sells(West,y,z) \lor \neg Hostile(z)

\neg Missile(x) \lor Weapon(x)

Missile(MI)

\neg Missile(y) \lor \neg Sells(West,y,z) \lor \neg Hostile(z)

\neg Sells(West,MI,z) \lor \neg Hostile(z)

\neg Missile(x) \lor \neg Owns(Nono,x) \lor Sells(West,x,Nono)

Missile(MI)

\neg Missile(MI) \lor \neg Owns(Nono,MI) \lor \neg Hostile(Nono)

\neg Owns(Nono,MI) \lor \neg Hostile(Nono)

\neg Missile(x, America) \lor Hostile(x)

\neg Hostile(Nono)

Enemy(Nono, America)

\neg Enemy(Nono, America)
Summary

- Forward chaining
- Backward chaining
- Resolution