

Proof Procedures

- •Interesting, proof procedures work by simply manipulating formulas. They do not know or care anything about interpretations.
- •Nevertheless they respect the semantics of interpretations!
- •We will develop a proof procedure for firstorder logic called resolution.
 - Resolution is the mechanism used by PROLOG

S3401 W 2017 Fahiem Bacchus & Yves Lesp

3

Properties of Proof Procedures

- •Before presenting the details of resolution, we want to look at properties we would like to have in a (any) proof procedure.
- •We write $KB \vdash f$ to indicate that f can be proved from KB (the proof procedure used is implicit).



Soundness

 $\blacksquare \ KB \vdash f \twoheadrightarrow \ KB \vDash f$

i.e all conclusions arrived at via the proof procedure are correct: they are logical consequences.

Completeness

 $\blacksquare \ KB \vDash f \rightarrow \ KB \vdash f$

i.e. every logical consequence can be generated by the proof procedure.

• Note proof procedures are computable, but they might have very high complexity in the worst case. So completeness is not necessarily achievable in practice.

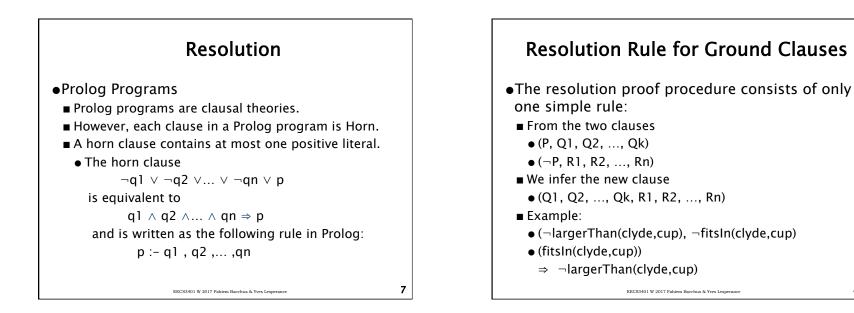
EECS3401 W 2017 Fahiem Bacchus & Yves Lesperance

Resolution

- •Clausal form.
 - Resolution works with formulas expressed in clausal form.
 - A literal is an atomic formula or the negation of an atomic formula. dog(fido), ¬cat(fido)
 - A clause is a disjunction of literals:
 - ¬owns(fido,fred) ∨ ¬dog(fido) ∨ person(fred)
 - We write (¬owns(fido,fred), ¬dog(fido), person(fred))

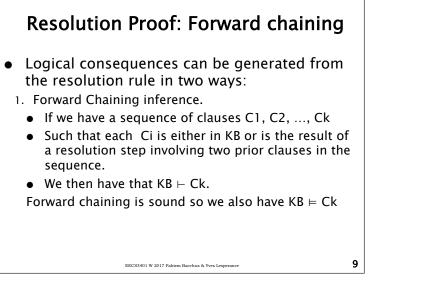
EECS3401 W 2017 Fahiem Bacchus & Yves Lesperand

■ A clausal theory is a conjunction of clauses.



5

6



Resolution Proof: Refutation proofs

2. Refutation proofs.

- We determine if $KB \vdash f$ by showing that a contradiction can be generated from $KB \land \neg f$.
- In this case a contradiction is an empty clause ().
- We employ resolution to construct a sequence of clauses C1, C2, ..., Cm such that

EECS3401 W 2017 Fahiem Bacchus & Yves Lesperance

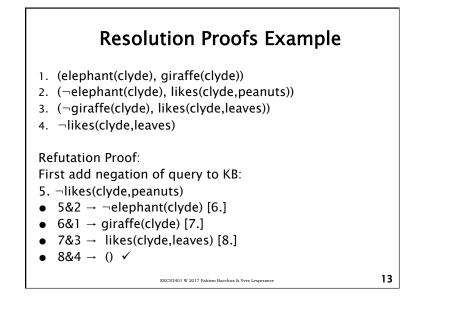
- Ci is in KB $\Lambda \neg f$, or is the result of resolving two previous clauses in the sequence.
- Cm = () i.e. its the empty clause.

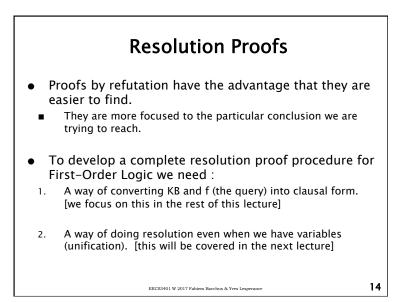
Resolution Proof: Refutation proofs

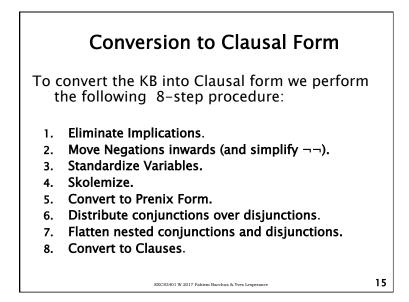
- •If we can find such a sequence C1, C2, ..., Cm=(), we have that
 - KB \vdash f.
 - Furthermore, this procedure is sound so • KB ⊨ f
- •And the procedure is also complete so it is capable of finding a proof of any f that is a logical consequence of KB. I.e.
 - \bullet If KB $\vDash f$ then we can generate a refutation from KB A $\neg f$

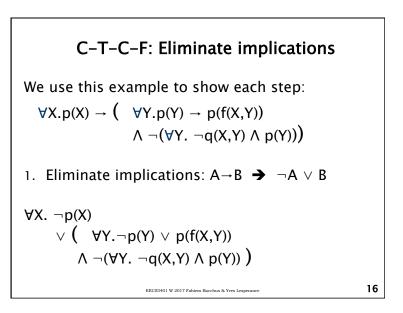
11

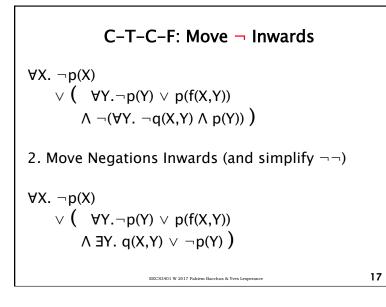
Resolution Proofs Example Want to prove likes(clyde, peanuts) from: 1. (elephant(clyde), giraffe(clyde)) 2. (\neg elephant(clyde), likes(clyde, peanuts)) 3. (\neg giraffe(clyde), likes(clyde, leaves)) 4. \neg likes(clyde, leaves) **Forward Chaining Proof:** 0. $3\&4 \rightarrow \neg$ giraffe(clyde) [5.] 0. $5\&1 \rightarrow$ elephant(clyde) [6.] 0. $6\&2 \rightarrow$ likes(clyde, peanuts) [7.] \checkmark

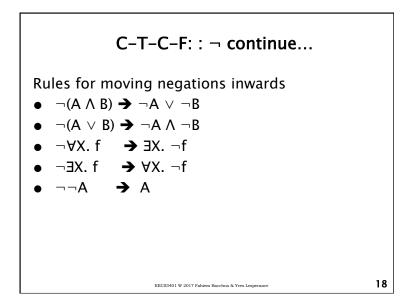




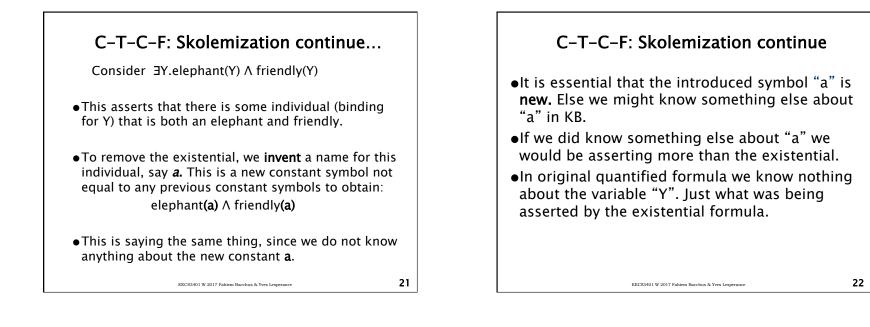








C-T-C-F: Standardize Variables $\forall X. \neg p(X)$ $\lor (\forall Y. \neg p(Y) \lor p(f(X,Y))$ $\land \exists Y. q(X,Y) \lor \neg p(Y))$ 3. Standardize Variables (Rename variables so that each quantified variable is unique) $\forall X. \neg p(X)$ $\lor (\forall Y. (\neg p(Y) \lor p(f(X,Y)))$ $\land \exists Z. q(X,Z) \lor \neg p(Z))$ $\begin{array}{l} \textbf{C-T-C-F: Skolemize} \\ \forall X. \neg p(X) \\ & \lor \left(\quad \forall Y. \neg p(Y) \lor p(f(X,Y)) \\ & \land \exists Z.q(X,Z) \lor \neg p(Z) \right) \\ \hline \\ \textbf{4. Skolemize (Remove existential quantifiers by introducing new function symbols).} \\ \forall X. \neg p(X) \\ & \lor \left(\forall Y. \neg p(Y) \lor p(f(X,Y)) \\ & \land q(X,g(X)) \lor \neg p(g(X)) \right) \end{array}$



C-T-C-F: Skolemization continue

Now consider $\forall X \exists Y$. loves(X,Y).

- •This formula claims that for every X there is some Y that X loves (perhaps a different Y for each X).
- Replacing the existential by a new constant won't work
 ∀X.loves(X.a).

Because this asserts that there is a **particular** individual "a" loved by every X.

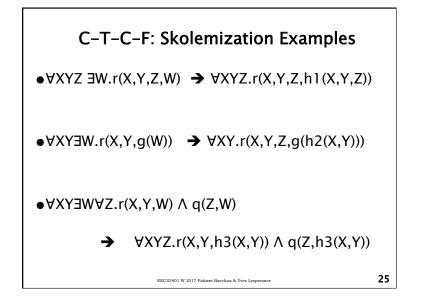
•To properly convert existential quantifiers scoped by universal quantifiers we must use **functions** not just constants.

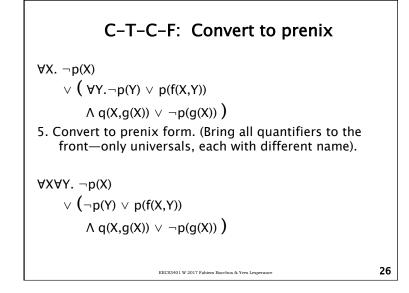
EECS3401 W 2017 Fahiem Bacchus & Yves Lespera

23

C-T-C-F: Skolemization continue

- •We must use a function that mentions every universally quantified variable <u>that scopes the existential</u>.
- In this case X scopes Y so we must replace the existential Y by a function of X
 ∀X. loves(X,g(X)).
 where g is a new function symbol.
- This formula asserts that for every X there is some individual (given by g(X)) that X loves. g(X) can be different for each different binding of X.

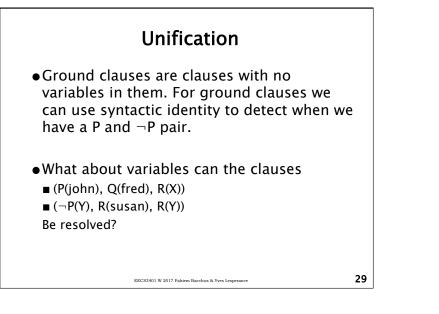


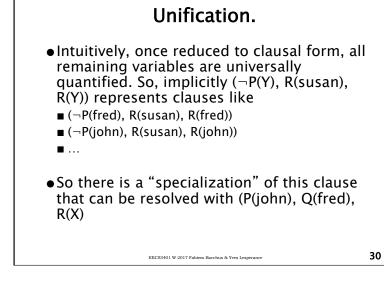


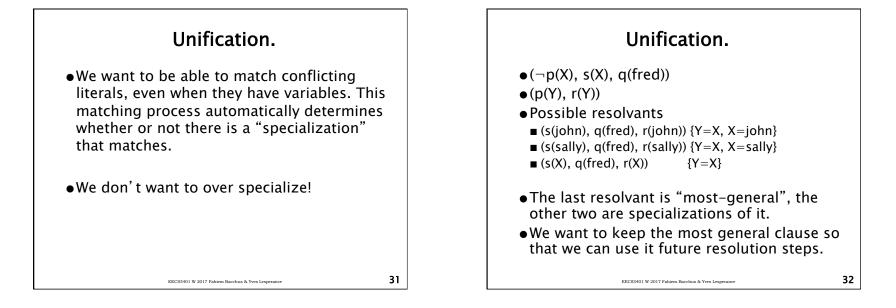
C-T-C-F: Conjunctions over disjunctions $\forall X \forall Y. \neg p(X)$ $\lor (\neg p(Y) \lor p(f(X,Y)))$ $\land q(X,g(X)) \lor \neg p(g(X)))$ 6. Conjunctions over disjunctions $\land \lor (B \land C) \Rightarrow (A \lor B) \land (A \lor C)$ $\forall XY. \neg p(X) \lor \neg p(Y) \lor p(f(X,Y))$ $\land \neg p(X) \lor q(X,g(X)) \lor \neg p(g(X))$

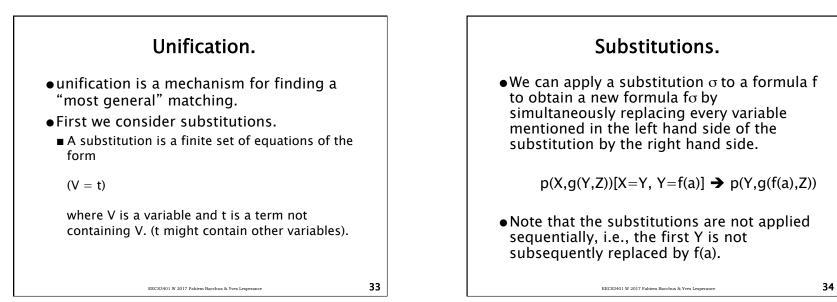
27

C-T-C-F: flatten & convert to clauses 7. Flatten nested conjunctions and disjunctions. $(A \lor (B \lor C)) \Rightarrow (A \lor B \lor C)$ 8. Convert to Clauses (remove quantifiers and break apart conjunctions). $\forall XY. \neg p(X) \lor \neg p(Y) \lor p(f(X,Y))$ $\Lambda \neg p(X) \lor q(X,g(X)) \lor \neg p(g(X))$ a) $\neg p(X) \lor \neg p(Y) \lor p(f(X,Y))$ $b) \neg p(X) \lor q(X,g(X)) \lor \neg p(g(X))$









Substitutions.

• We can compose two substitutions. θ and σ to obtain a new substition $\theta\sigma.$

Let $\theta = \{X_1 = s_1, X_2 = s_2, ..., X_m = s_m\}$ $\sigma = \{Y_1 = t_1, Y_2 = t_2, ..., Y_k = s_k\}$

To compute $\theta\sigma$

1.
$$S = \{X_1 = s_1\sigma, X_2 = s_2\sigma, ..., X_m = s_m\sigma, Y_1 = t_1, Y_2 = t_2, ..., Y_k = s_k\}$$

we apply σ to each RHS of θ and then add all of the equations of $\sigma.$

CS3401 W 2017 Fahiem Bacchus & Yves Le

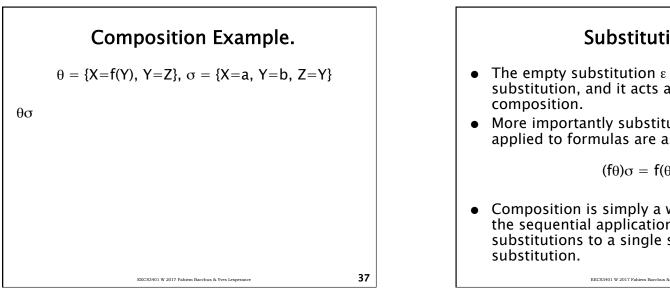
35

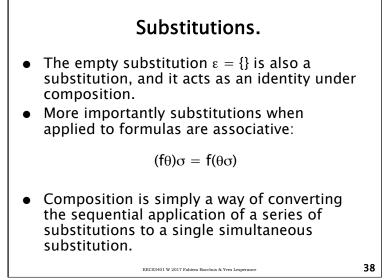
Substitutions.

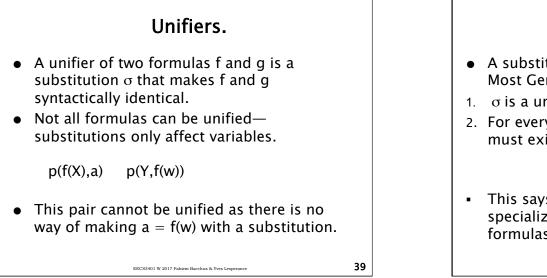
- 1. $S = \{X_1 = s_1\sigma, X_2 = s_2\sigma, ..., X_m = s_m\sigma, Y_1 = t_1, Y_2 = t_2, ..., Y_k = s_k\}$
- 2. Delete any identities, i.e., equations of the form V=V.
- 3. Delete any equation $Y_i = s_i$ where Y_i is equal to one of the X_i in θ .

EECS3401 W 2017 Fahiem Bacchus & Yves Le

The final set S is the composition $\theta\sigma$.

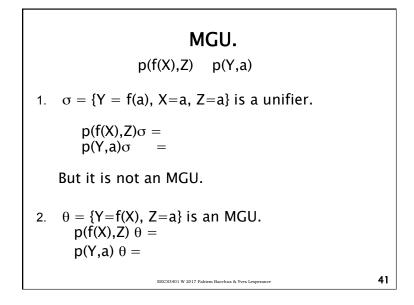


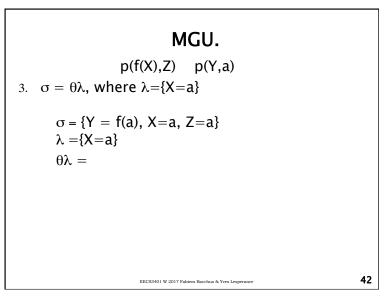




MGU.

- A substitution σ of two formulas f and g is a Most General Unifier (MGU) if
- 1. σ is a unifier.
- 2. For every other unifier θ of f and g there must exist a third substitution λ such that $\theta = \sigma \lambda$
- This says that every other unifier is "more specialized than σ . The MGU of a pair of formulas f and g is unique up to renaming.





MGU.

- The MGU is the "least specialized" way of making clauses with universal variables match.
- We can compute MGUs.
- Intuitively we line up the two formulas and find the first sub-expression where they disagree. The pair of subexpressions where they first disagree is called the disagreement set.
- The algorithm works by successively fixing disagreement sets until the two formulas become syntactically identical.

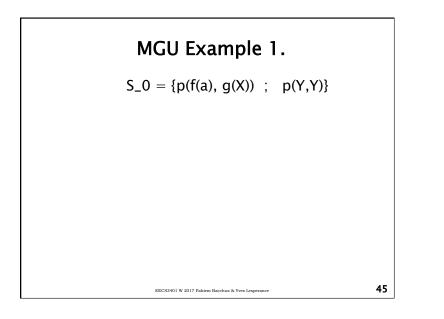
43

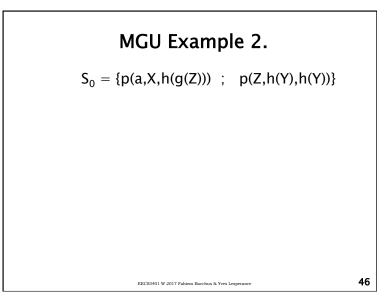
MGU.

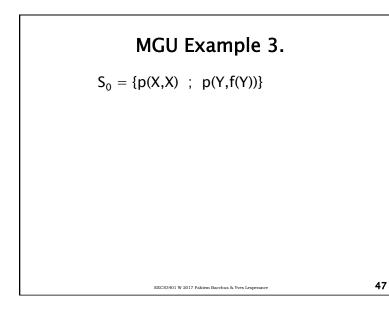
To find the MGU of two formulas f and g.

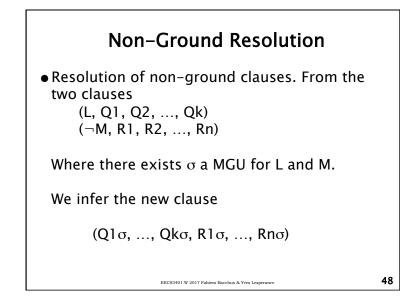
1.
$$k = 0; \sigma_0 = \{\}; S_0 = \{f,g\}$$

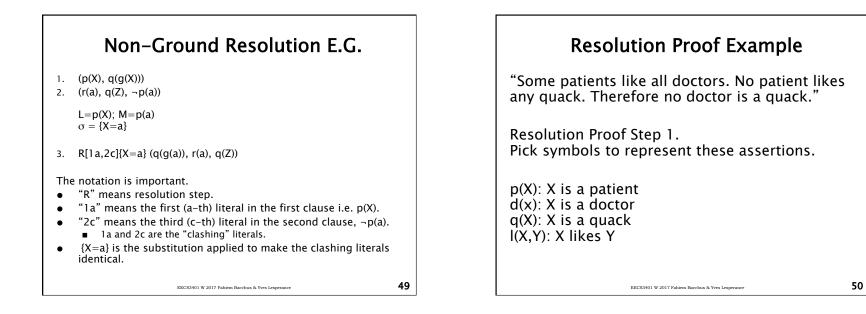
- 2. If S_k contains an identical pair of formulas stop, and return σ_k as the MGU of f and g.
- 3. Else find the disagreement set $D_k = \{e_1, e_2\}$ of S_k
- 4. If e₁ = V a variable, and e₂ = t a term not containing V (or vice-versa) then let σ_{k+1} = σ_k {V=t} (Compose the additional substitution) S_{k+1} = S_k{V=t} (Apply the additional substitution) k = k+1 GOTO 2
 5. Else stop, f and g cannot be unified.











51

Resolution Proof Example

EECS3401 W 2017 Fahiem Bacchus & Yves Lesp

Resolution Proof Step 2. Convert each assertion to a first-order formula.

1. Some patients like all doctors.

F1.

Resolution Proof Example

EECS3401 W 2017 Fahiem Bacchus & Yves Lesperand

2. No patient likes any quack

F2.

3. Therefore no doctor is a quack. Query.

Resolution	Proof	Examp	e
------------	-------	-------	---

EECS3401 W 2017 Fahiem Bacchus & Yves Lesperanc

Resolution Proof Step 3. Convert to Clausal form.

F1.

F2.

Negation of Query.

53

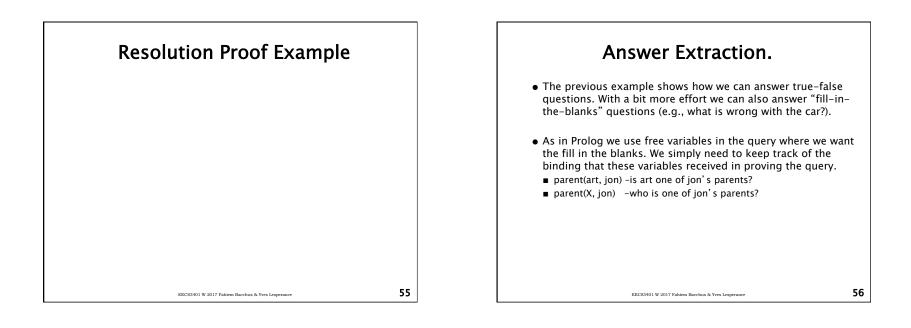
Resolution Proof Example

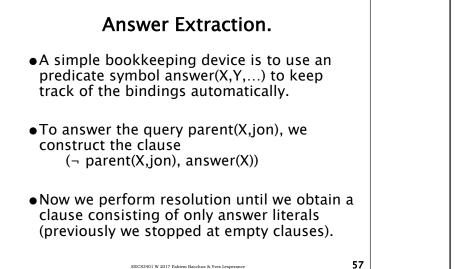
EECS3401 W 2017 Fahiem Bacchus & Yves Lesperance

Resolution Proof Step 4. Resolution Proof from the Clauses. 1. p(a)

- 2. (¬d(Y), l(a,Y))
- 3. $(\neg p(Z), \neg q(R), \neg I(Z,R))$
- 4. d(b)
- 5. q(b)







Answer Extraction: Example 1

