Knowledge Representation & Reasoning

Shyamanta M Hazarika
CSE, SoE
Tezpur University

Generalizing CNF

Resolution will generalize to handling variables

Ignore = for now

But to convert wffs to CNF, we need three additional steps:

1. eliminate $\supset$ and $\equiv$
2. push $\neg$ inward using also $\neg\forall x.\alpha \twoheadrightarrow \exists x.\neg\alpha$ etc.
3. standardize variables: each quantifier gets its own variable

   e.g. $\exists x[P(x)] \wedge Q(x) \twoheadrightarrow \exists z[P(z)] \wedge Q(x)$ where z is a new variable
4. eliminate all existentials *(discussed later)*
5. move universals to the front using $(\forall x\alpha) \wedge \beta \twoheadrightarrow \forall x(\alpha \wedge \beta)$

   where β does not use x
6. distribute $\vee$ over $\wedge$
7. collect terms

Get universally quantified conjunction of disjunction of literals

then drop all the quantifiers...

First-order resolution

Main idea: a literal (with variables) stands for all its instances; so allow all such inferences

So given $[P(x,a), \neg Q(x)]$ and $[\neg P(b,y), \neg R(b,f(y))]$,
want to infer $[\neg Q(b), \neg R(b,f(a))]$ among others

since $[P(x,a), \neg Q(x)]$ has $[P(b,a), \neg Q(b)]$ and
$[\neg P(b,y), \neg R(b,f(y))]$ has $[\neg P(b,a), \neg R(b,f(a))]$

Resolution:

Given clauses: $\{\rho_1\} \cup C_1$ and $\{\overline{\rho_2}\} \cup C_2$.

Rename variables, so that distinct in two clauses.

For any θ such that $\rho_1\theta = \rho_2\theta$, can infer $(C_1 \cup C_2)\theta$.

We say that ρ_1 unifies with ρ_2 and that θ is a unifier of the two literals

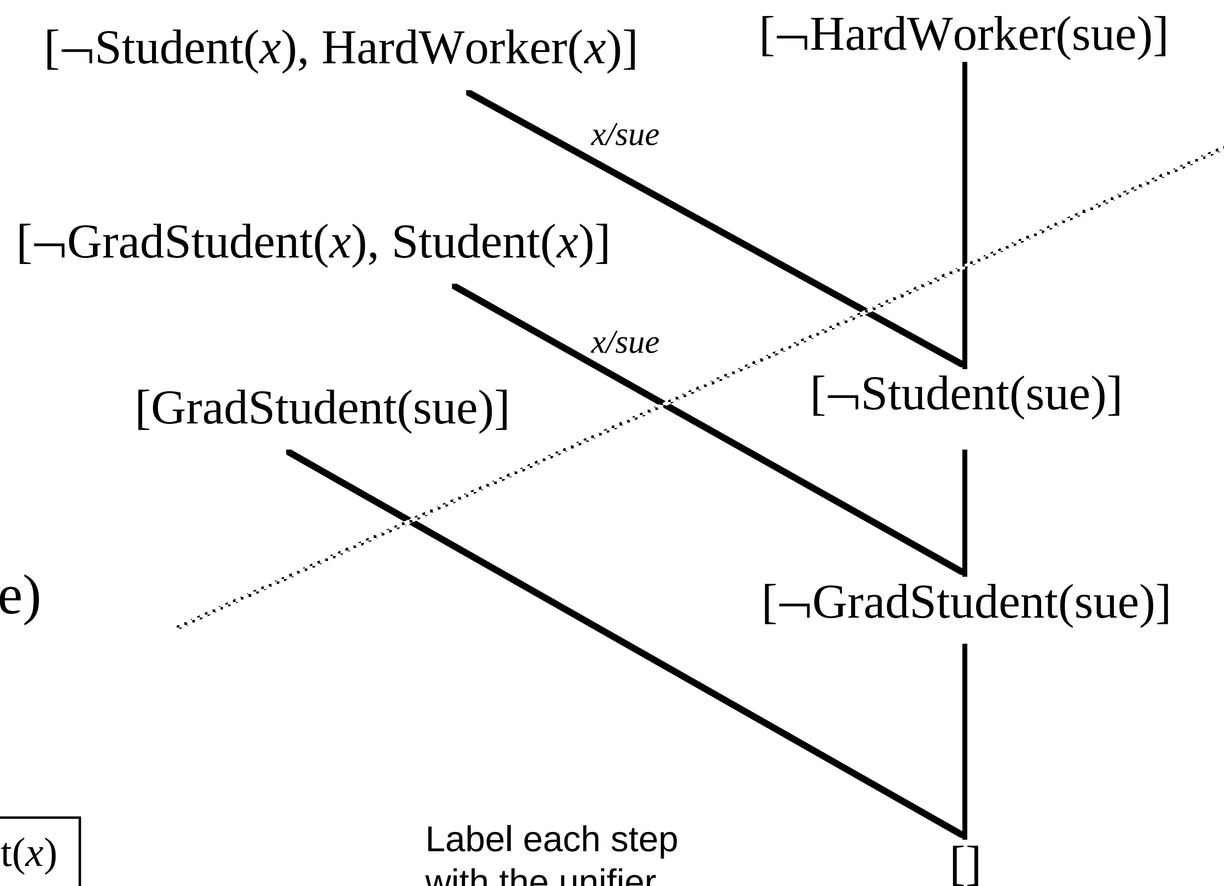
Resolution derivation: as before

Theorem: $S \rightarrow []$ iff $S \models []$ iff S is unsatisfiable

Note: There are pathological examples where a slightly more general definition of Resolution is required. We ignore them for now...

Example 3

$$KB \overset{?}{\models} \text{HardWorker(sue)}$$



KB

$\forall x$ GradStudent(x) $\supset$ Student(x)

$\forall x$ Student(x) $\supset$ HardWorker(x)

GradStudent(sue)

Label each step with the unifier

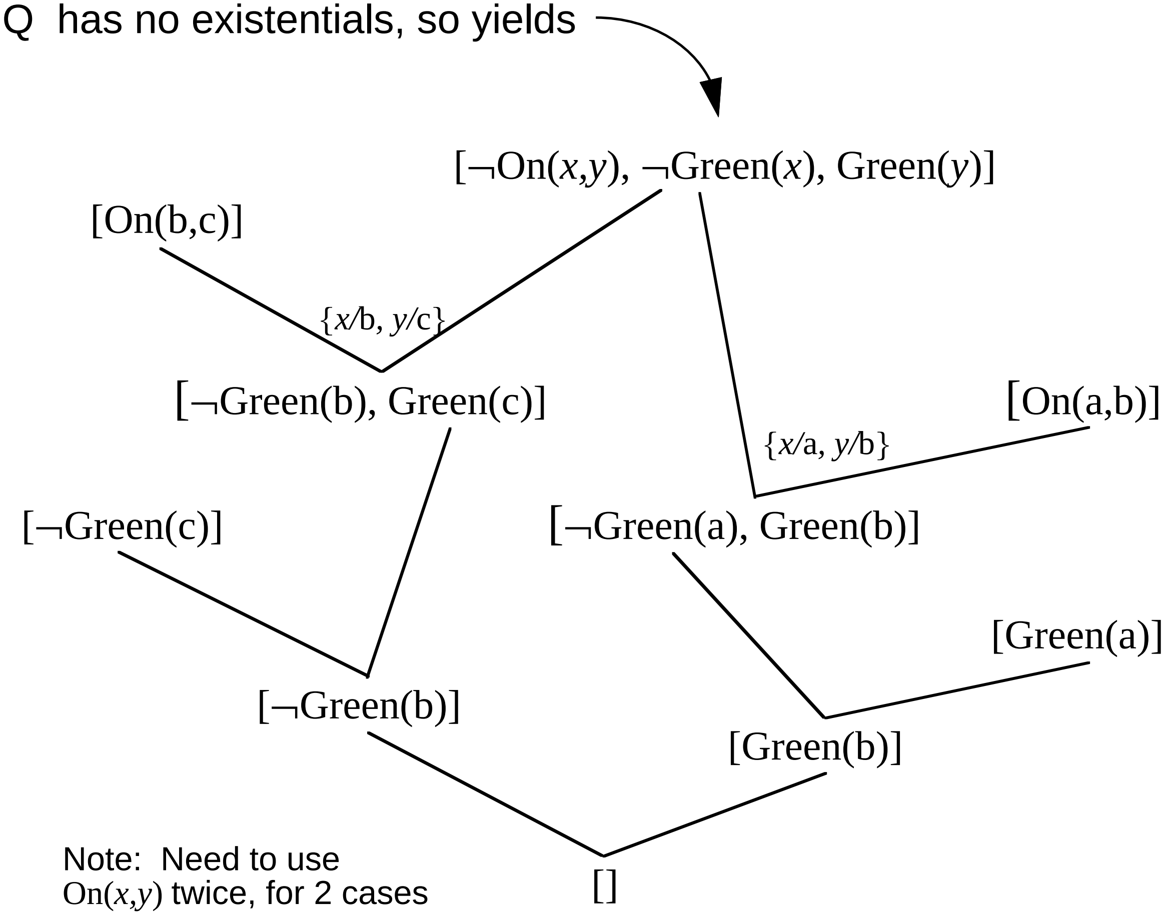
Point to relevant literals in clauses

The 3 block example

KB = {On(a,b), On(b,c), Green(a), ¬Green(c)}   already in CNF

Query = $\exists x \exists y [\mathrm{On}(x,y) \wedge \mathrm{Green}(x) \wedge \neg\mathrm{Green}(y)]$

Note: ¬Q has no existentials, so yields



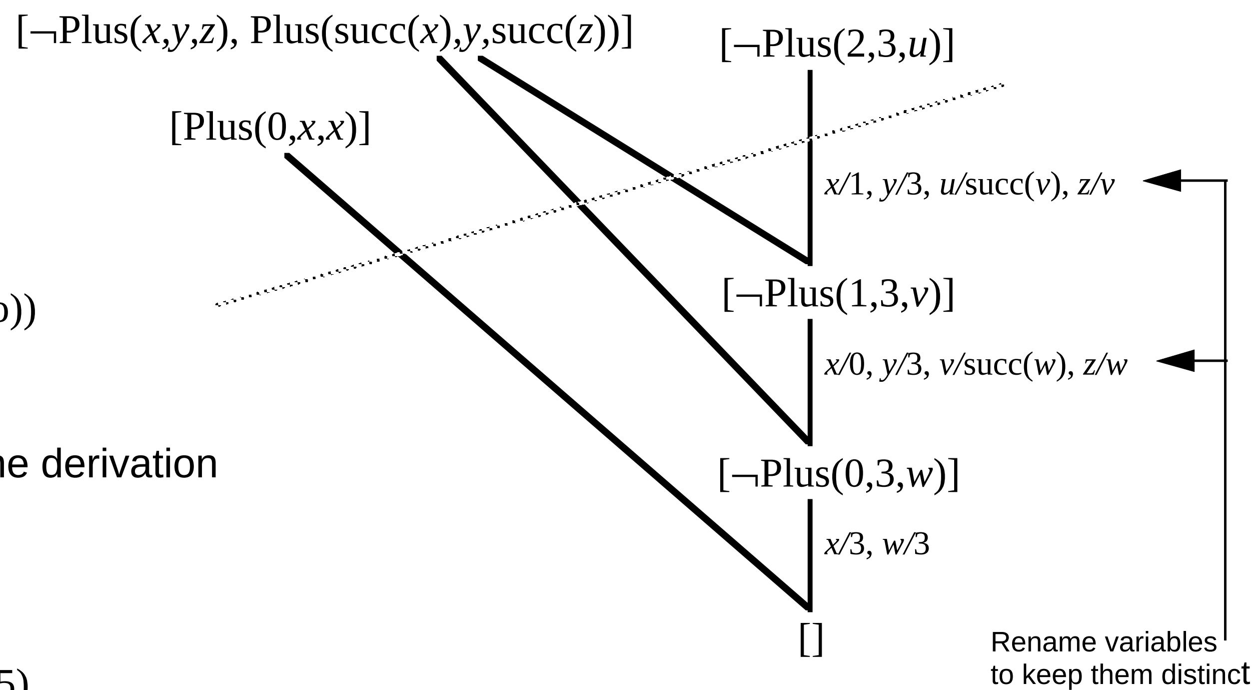
Arithmetic

KB: Plus(zero,x,x)
Plus(x,y,z) $\supset$ Plus(succ(x),y,succ(z))

Q: $\exists u$ Plus(2,3,u)

For readability,
we use
- 0 for zero,
- 1 for succ(zero),
- 2 for succ(succ(zero))

etc.



Can find the answer in the derivation

u/succ(succ(3))

that is: *u*/5

Can also derive Plus(2,3,5)

Answer predicates

In full FOL, we have the possibility of deriving $\exists x P(x)$ without being able to derive $P(t)$ for any t.

e.g. the three-blocks problem

$$\exists x \exists y [\mathrm{On}(x,y) \wedge \mathrm{Green}(x) \wedge \neg\mathrm{Green}(y)]$$

but cannot derive which block is which

Solution: answer-extraction process

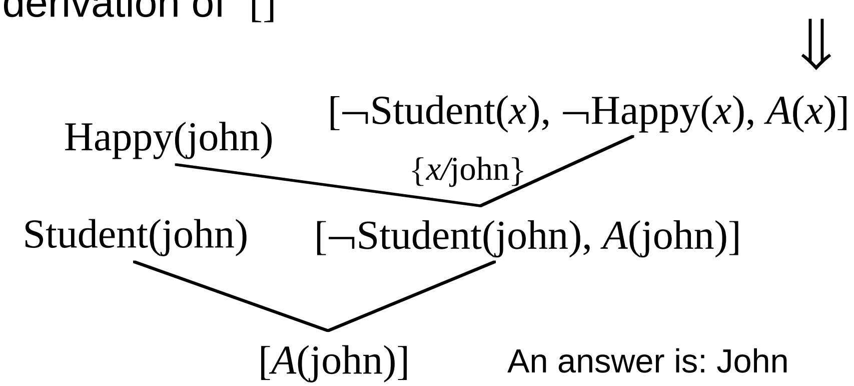
- replace query $\exists x P(x)$ by $\exists x [P(x) \wedge \neg A(x)]$

  where A is a new predicate symbol called the <u>answer predicate</u>

- instead of deriving [], derive any clause containing just the answer predicate
- can always convert to and from a derivation of []

KB: Student(john)
Student(jane)
Happy(john)

Q: $\exists x [\mathrm{Student}(x) \wedge \mathrm{Happy}(x)]$



An answer is: John

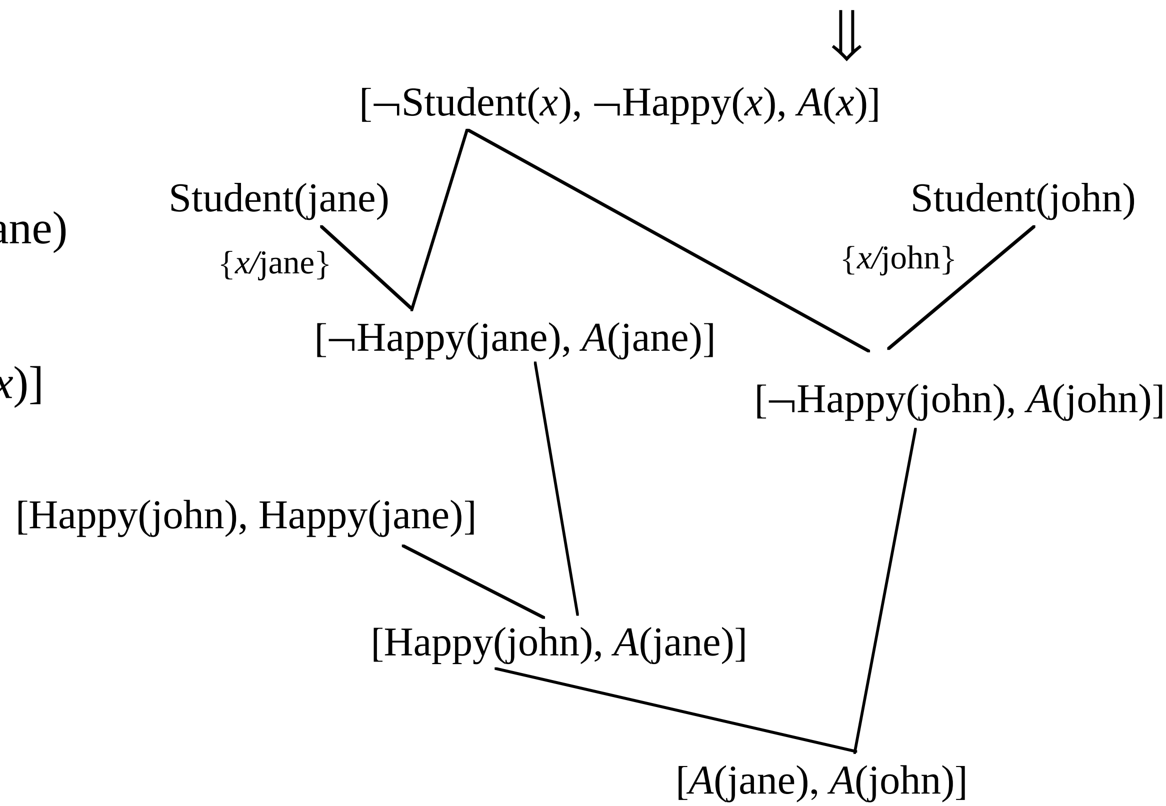
Disjunctive answers

KB:

Student(john)
Student(jane)
Happy(john) $\vee$ Happy(jane)

Query:

$\exists x$[Student(x) $\wedge$ Happy(x)]



An answer is: either Jane or John

Note:

- can have variables in answer
- need to watch for Skolem symbols... (next)