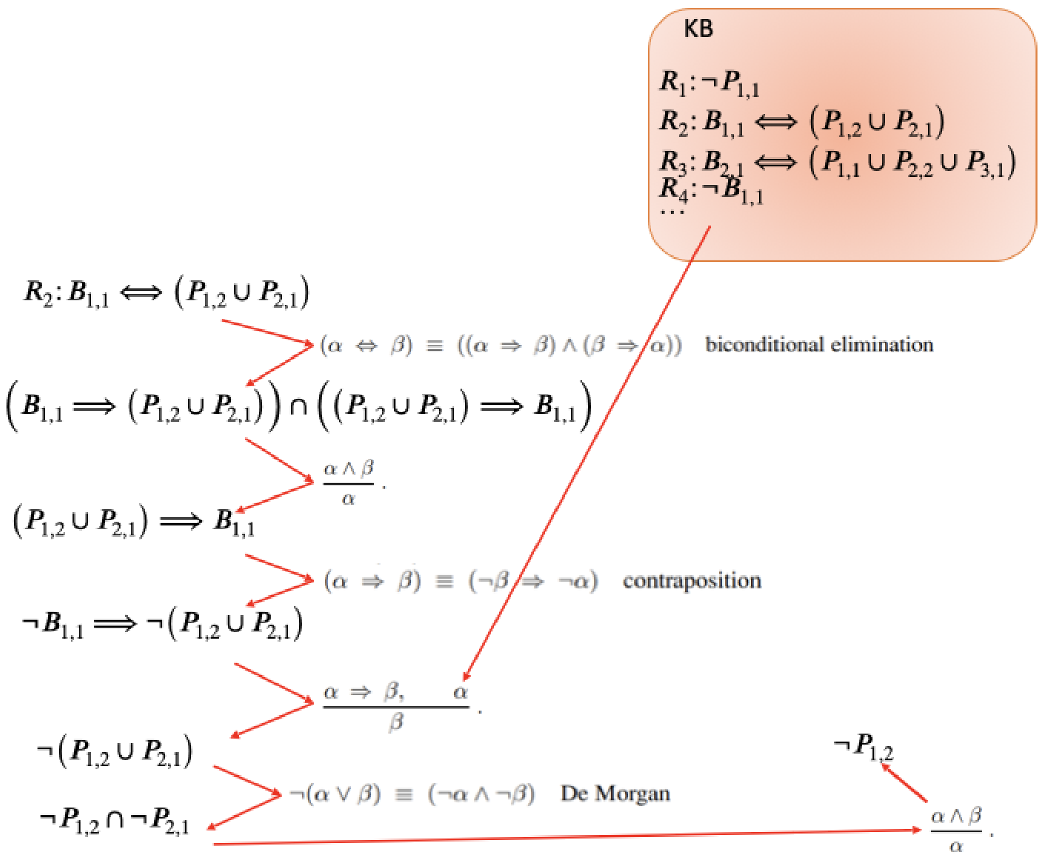
CAS CS 440
Lec 14

Logic III

1. Review
    a. Prepositional logic has flaws
        i. Two sentences can have the same meaning (even with the CNF)
        ii. It cannot express some things (cannot deal with time, functions – make variable with every square)
    b. Using Inference Rules
        i. Is there a pit in (1,2)?



        ii.
        iii. Chose R2 because it includes information regarding P(1,2) and we know from R4 that there is no breeze in (1,1)
        iv. Computers find all information regarding the new sentence and adds them into the KB
        v. It also has a query so we can delete information not needed and verify that sentences created are true or false
    c. Inference Algorithms
        i. Use any search algorithm

ii. Proof problem:
    1. Initial state = initial KB
    2. Actions = all inference rules applied to all sentences that match the top half of inference rule
    3. Result of applying an action = add the sentence in the bottom half of the inference rule to the KB
    4. Goal: state that contains the sentence we are trying to prove! (stop search)

iii. Searching for proof = enumerating all possible models!
    1. More efficient
        a. Ignoring irrelevant variables
        b. Skipping over the things that are false
        c. It is still inefficient

2. Completeness?
    a. So far we have a sound inference procedure
    b. Is it complete? (can I derive every possible good sentences?)
        i. No
        ii. Cannot "resolve" disjunctions (ors)
            1. Cannot put two sentences that are connected with "or" into new information
    c.

Lets say we know this   Also know this

$$\frac{P_{1,1} \vee P_{2,2} \vee P_{3,1} \qquad \neg P_{2,2}}{P_{1,1} \vee P_{3,1}}$$

Can infer this!

→ this is not a logical rule in the fourteen rules defined

    d.

$$\frac{P_{1,1} \vee P_{3,1}, \qquad \neg P_{1,1} \vee \neg P_{2,2}}{P_{3,1} \vee \neg P_{2,2}}$$

    e. Resolution inference rule

$$\frac{\ell_1 \vee \cdots \vee \ell_k, \qquad m_1 \vee \cdots \vee m_n}{\ell_1 \vee \cdots \vee \ell_{i-1} \vee \ell_{i+1} \vee \cdots \vee \ell_k \vee m_1 \vee \cdots \vee m_{j-1} \vee m_{j+1} \vee \cdots \vee m_n}$$

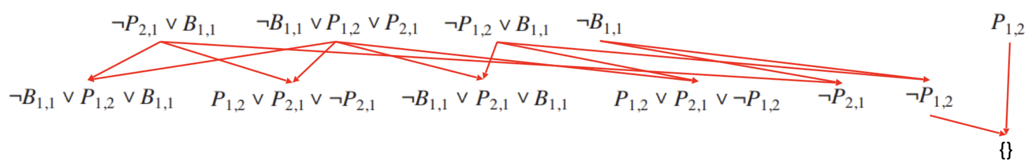
        i. Resulting clause should only have one copy per literal (variable)
    f. Now our inferences are complete

3. Conjuctive Normal Form
    a. Resolution only applies to disjunctions
    b. Good news!
        i. Every sentence can be converted into a conjunction of clauses!
        ii. It is very powerful and important that we can ignore all the fourteen previous rules (it is that powerful)
        iii. Might look ugly to us, but search algorithm doesn't care!
    c.
$$B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$$
$$(B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$$
$$(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg(P_{1,2} \vee P_{2,1}) \vee B_{1,1})$$
$$(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge ((\neg P_{1,2} \wedge \neg P_{2,1}) \vee B_{1,1})$$
$$(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg P_{1,2} \vee B_{1,1}) \wedge (\neg P_{2,1} \vee B_{1,1})$$
    d. Grouping of disjunctions is called clauses
    e. Therefore, we use the 14 rules to make the sentences into CNFs and then use CNFs → we can only check resolution now
    f. Makes everything way faster
4. Proving via Resolution
    a. Turns out all we need is resolution (to prove if KB |= a)
    b. Convert $\boldsymbol{KB} \cap \neg \boldsymbol{\alpha}$ into CNF → proof by contradiction
    c. Apply resolution to resulting clauses
        i. Each pair of clauses that contains complimentary literals produces new clauses
    d. Repeat until one of two outcomes occurs:
        i. No new clauses → KB does not entail a
        ii. Two clauses resolve to the empty clause → KB entails a
    e. The runtime is O(n^2)
        i. Only pick pairs that are contradictory (only do resolution that have complementary literals) → not necessarily O(n^2)
5. Example
    a. KB =
$$(B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})) \wedge \neg B_{1,1}$$
    b. a =
$$\neg P_{1,2},$$

c. $\boldsymbol{KB \cap \neg\alpha}$:

$$(\neg B_{1,1} \lor P_{1,2} \lor P_{2,1}) \land (\neg P_{1,2} \lor B_{1,1}) \land (\neg P_{2,1} \lor B_{1,1}) \land \neg B_{1,1} \land P_{1,2}$$

d. CNF of $\boldsymbol{KB \cap \neg\alpha}$:



i. Whenever we encounter a contradiction, that statement is true (proof by contradiction) → KB entails 'a'

6. Prepositional Logic Resolution Algorithm

a.

function PL-RESOLUTION(KB, α) **returns** *true* or *false*

inputs: KB, the knowledge base, a sentence in propositional logic
α, the query, a sentence in propositional logic

clauses ← the set of clauses in the CNF representation of $KB \land \neg\alpha$
new ← { }
loop do
  for each pair of clauses C_i, C_j **in** *clauses* **do**
    resolvents ← PL-RESOLVE(C_i, C_j)
    if *resolvents* contains the empty clause **then return** *true*
    new ← *new* ∪ *resolvents*
  if *new* ⊆ *clauses* **then return** *false*
  clauses ← *clauses* ∪ *new*