

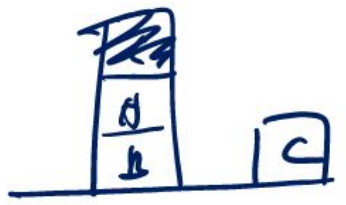
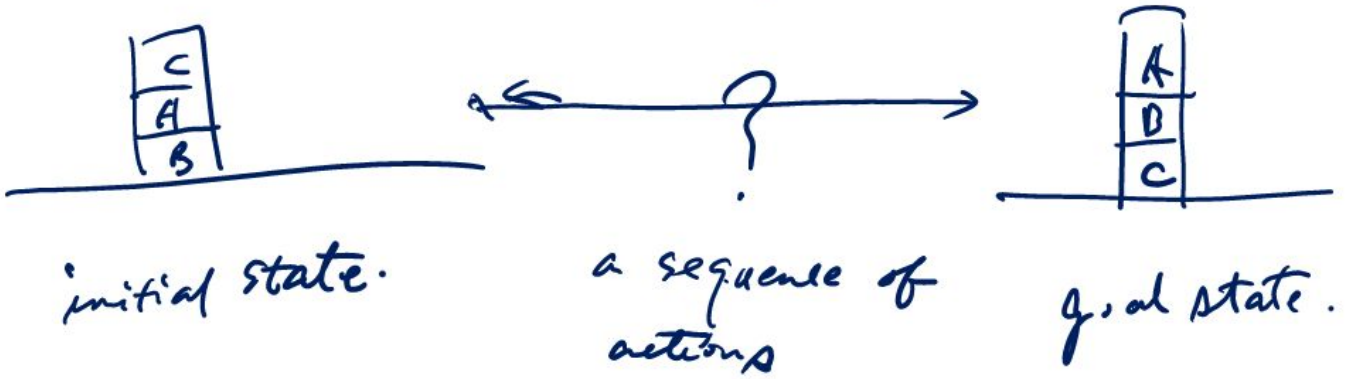
COMP-4475

All slides

W1ch2 lecture notes. Jan 11th

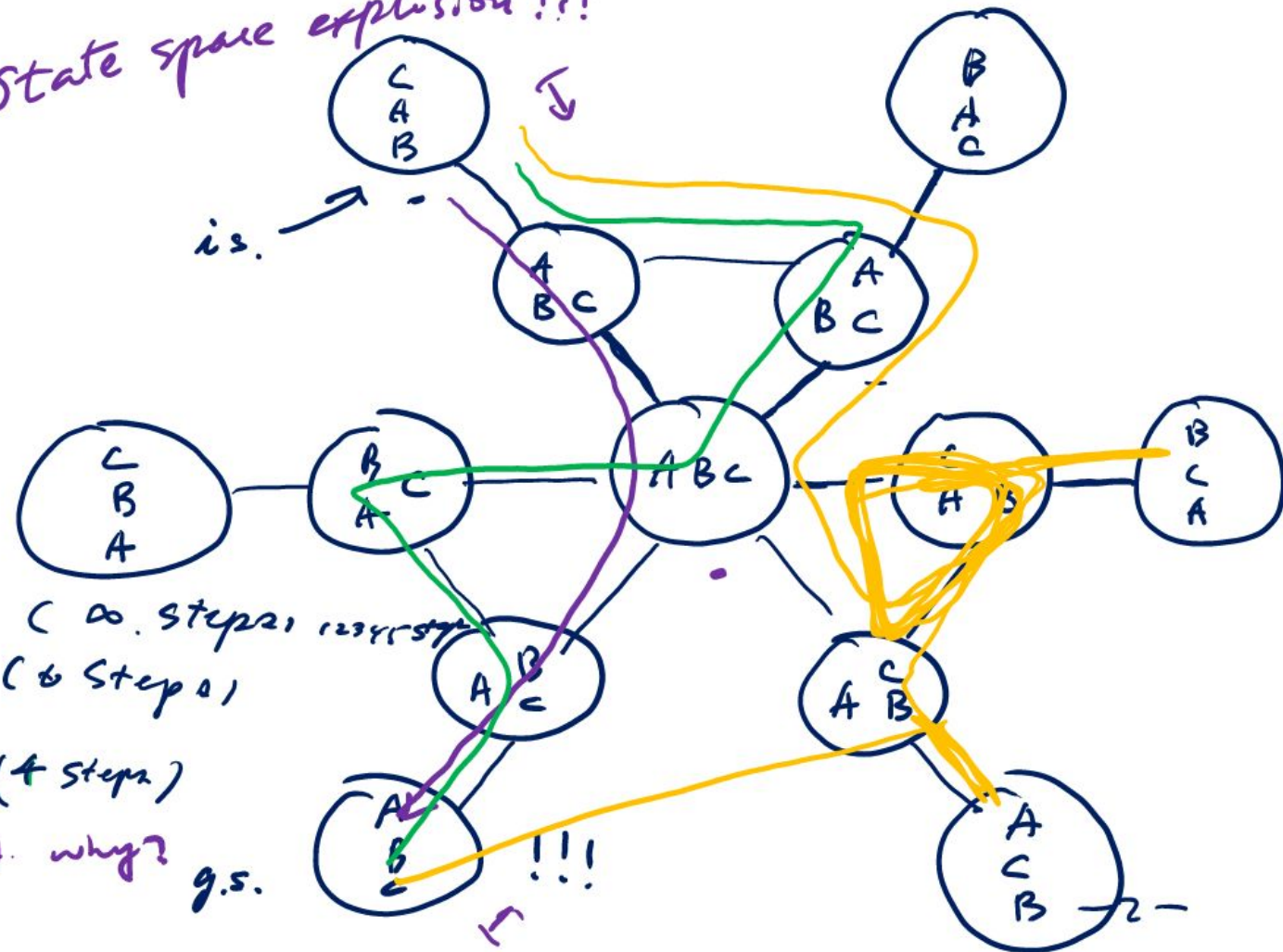
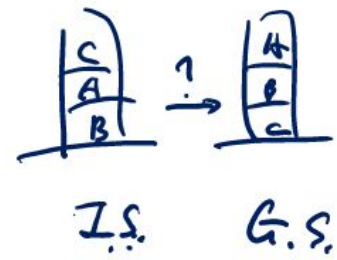
problem solving through searching on state space, block worlds, 8-puzzles, and Hanoi.

a. classical. AI. problem. : BLOCK WORLD.



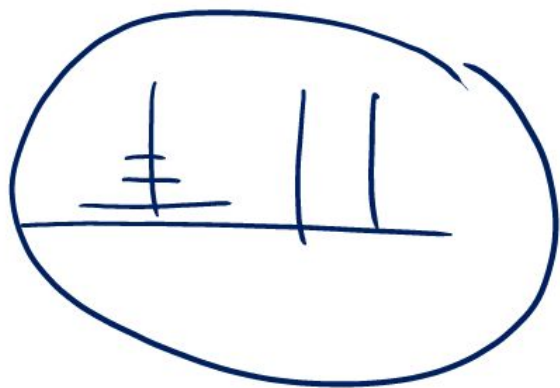
Searching in
State-space

state space explosion!!!

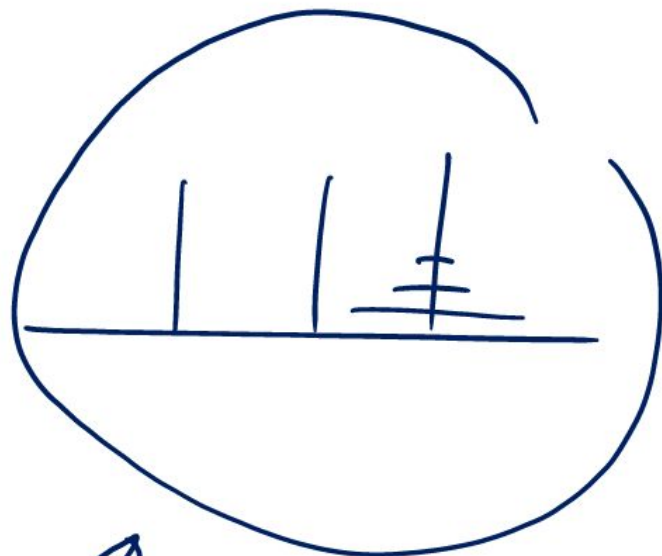


ouch! (10 steps, 12345678910 steps)
fine! (6 steps)
yeah!!! (4 steps)
non-easy. why?
g.s.

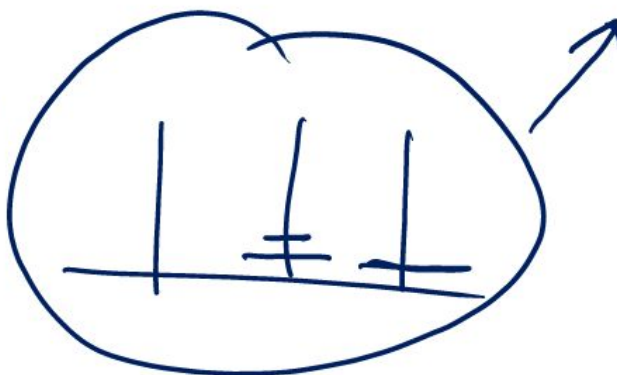
Tower of Hanoi



i. s.



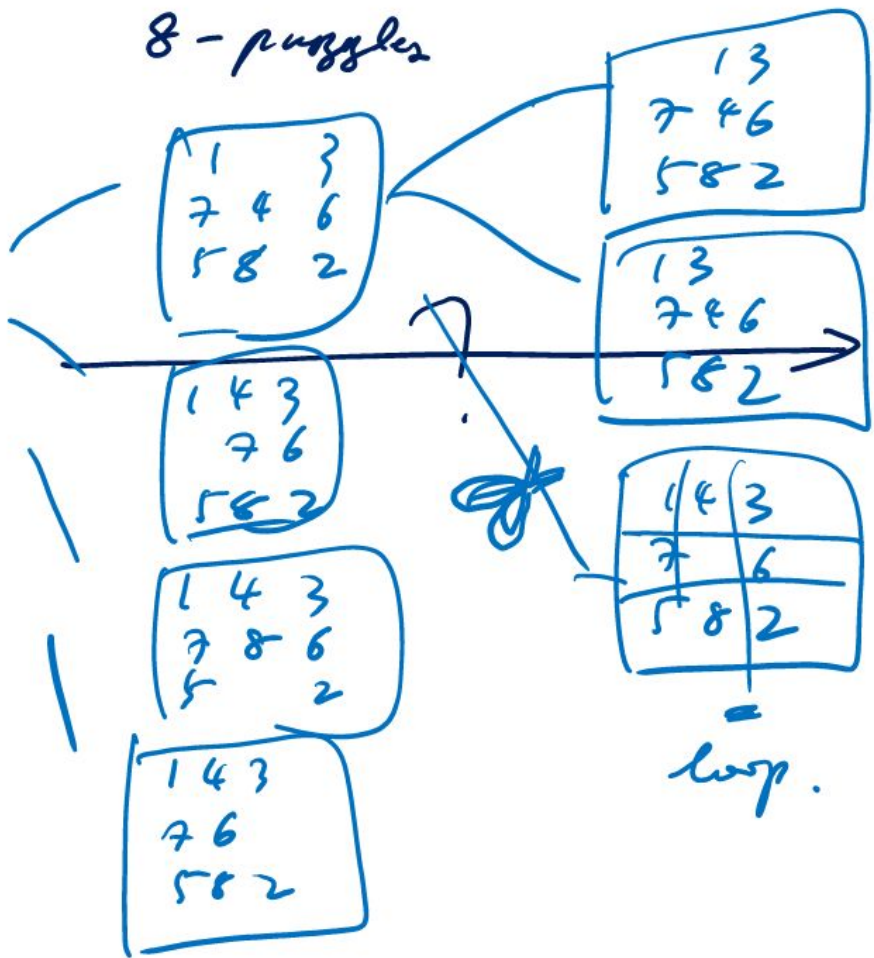
G. S.



8-puzzles

1	4	3
7		6
5	8	2

i.s.



1	3	6
7	4	2
5	8	

g.s.

W3c4 lecture notes. Jan 18th

Graph Searching: DFS (Stack), LCFS.

Stack



0th.



1st



2nd



3rd



4th



5th



6th



7th



8th

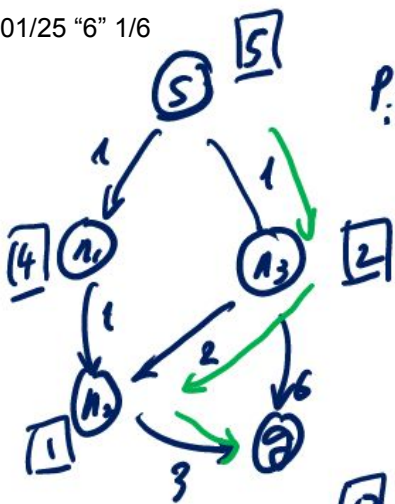


9th

1/1

w3 c6 lecture notes. Jan 25

Admissibility and optimality of A* search



P.Q.

$$\frac{5}{|S|}$$

\downarrow $\{S\}$

$$\frac{3 \quad 5}{|n_3 \quad n_1|}$$

\downarrow $\{S, n_3\}$

$$\frac{4 \quad 5 \quad 7 = 1 + 6 + 0}{|n_2 \quad n_1 \quad g|}$$

\downarrow $\{S, n_2, n_3\}$

$$\frac{5 \quad 6 \quad 7}{|n_1 \quad g' \quad g''|}$$

$\{S, n_1, n_2, n_3\}$

$$\frac{3 \quad 6 \quad 7}{|n_2 \quad g' \quad g''|}$$

\downarrow

$$\frac{5 \quad 6 \quad 7}{|g \quad g' \quad g''|}$$

\downarrow

$S \rightarrow n_1 \rightarrow n_2 - g$

Cost = $1 + 1 + 3 = 5$

optimal.

$$\frac{6 \quad 7}{|g' \quad g''|}$$

$S \rightarrow n_3 \rightarrow n_2 \rightarrow g$


Cost = $1 + 2 + 3 = 6$.

Non-optimal.

Notes: graph nodes with different subscripts are for different nodes in the graph (e.g. u_1, u_2 are two different nodes in the set V of G), whereas same graph nodes with different superscripts are for different appearances in the locations of the search tree (e.g. n_2^i and n_2^u).

Defn. (Admissibility of A^*). The A^* Search is admissible iff, as long as solutions exist, an optimal solution (i.e., a shortest path from a start node s to the goal node g), will be found by A^* , ~~even~~ even if the search is infinite (i.e. graphs with cycles).

Thm. The search A^* is admissible if

- the branching factor of the tree is finite. 
- edge costs are bounded above zero.
- the heuristic function $h(n)$ is a lower bound on the actual minimum cost of the shortest path from n to the goal node, g .

pf: 1. we will first prove that A^* always find a soln. upon the settings above. If the search tree is with finite depth (no cycle), then the frontier F will not be trapped into infinite cycles, and all nodes/paths will be explored sooner or later, including the soln paths. Since there exists at least one.

Meanwhile. if the tree is infinite, it means some nodes will be inserted into F and selected from F repeatedly. But each time the node (say node n) is selected. Its f -value. will be increased.

$$f(n) = \underline{g(n)} + \underline{h(n)}.$$

Since the actual cost for reaching n from S , $g(n)$ always increases. But this means a frontier node for the soln path will eventually be selected.

2.

2. we now prove that the 1st soln is optimal.

Suppose we have an optimal path in the form

$$(S, \underline{n_1}, n_2, \dots, n_k, \underline{g}). \quad \underline{f(g)} = \underline{g(g)} \leq \underline{g(g')} = \underline{f(g')}$$

and $(S \rightarrow g')$ refers to some non-optimal soln.

we know that.

in other words: $f(g') = g(g') + h(g') = g(g')$

$$\underline{f(n_1)} = \underline{f(g)} \leq \underline{f(g')} \quad \text{equals to the actual cost from } S \text{ to } g'.$$

Since n_1 is a neighbor of S , it must be inserted into F .

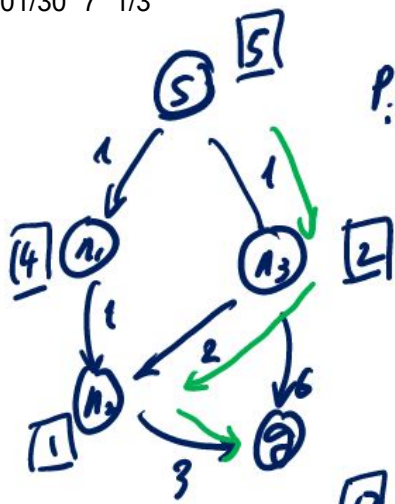
But. $f(n_1) = \underline{g(n_1)} + \underline{h(n_1)}$, is less than or equals to $f(g)$, which is the minimal cost of the path to g from S . — 5 —



$f(n_1) < f(g')$, which means n_1 will be expanded ~~before~~
 before g' . In addition, n_2 as a neighbour of n_1 will be inserted
 into F . Eventually, n_k will be inserted into F before g' is
 expanded. Since $f(n_k) < f(g')$, n_k will be expanded,
 leading to the insertion of g . Since $f(g) < f(g')$, finally
 the optimal path/node g , will be selected before g' .

W4c7 lecture notes. Jan 30

monotone restriction for MPC (multiple-path checking) in A^* , an example.



P.Q.

$$\frac{5}{151}$$

$\{S\}$

$$\frac{3 \quad 5}{|n_3 \quad n_1|}$$

$\{S, n_3\}$

$$\frac{4 \quad 5 \quad 7 = 1 + 6 + 0}{|n_2' \quad n_1 \quad g''|}$$

$\{S, n_2, n_3\}$

$$\frac{5 \quad 6 \quad 7}{|n_1 \quad g' \quad g''|}$$

$\{S, n_1, n_2, n_3\}$

$$\frac{3 \quad 6 \quad 7}{|n_2 \quad g' \quad g''|}$$

$$\frac{5 \quad 6 \quad 7}{|g \quad g' \quad g''|}$$

$S \rightarrow n_3 \rightarrow n_2 \rightarrow g$

Cost = $1 + 2 + 3 = 6$.

Non-optimal.

$S \rightarrow n_1 \rightarrow n_2 \rightarrow g$

Cost = $1 + 1 + 3 = 5$

optimal.

$\neg p \leftarrow \text{true.}$

Musk Is Rich

little guy.

\downarrow
 $K B \vdash \alpha$

$(p \vee q)$

\Updownarrow

$(p \wedge q)$

$(p \wedge \neg p)$

$M \models \alpha$

fact

T

\swarrow
meter

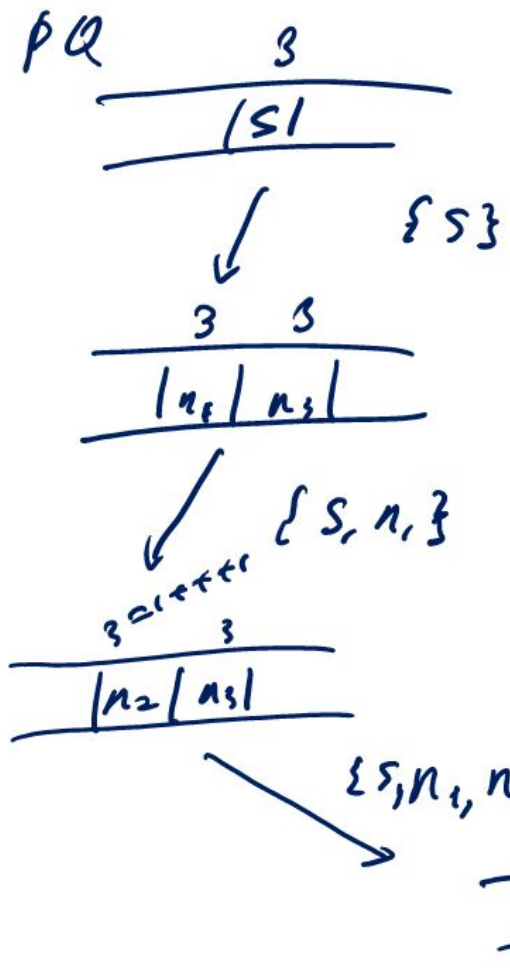
$K B$ \rightarrow FOL

~~\neg~~

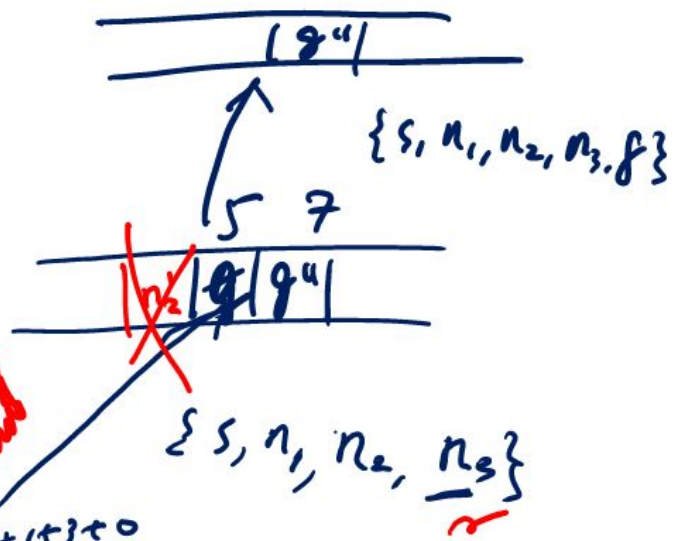
$(p \vee \neg p)$

ISO 9000

MPC



$S \rightarrow n_1 \rightarrow n_2 \rightarrow g$



MPC

$3 \quad 5 = 1 + 1 + 3 + 0$

$\underline{n_3} \quad g$

W4c8 lecture notes. Feb 01

FOL via an example of Alice, Bob, Carol, and David.

alphabet/grammar: FOL.

Constant: alice, bob, carol, david.

1-arity function: bestFriendOf.

0-arity predicate: ThisIsAliceWorld.
propositional.

1-arity predicate: Happy, Male, Female.

2-arity predicate: Couple. *denotation.*

$\forall, \exists, \wedge, \vee$ ~~($\rightarrow, \leftrightarrow$)~~

Interpretation I_1 , denoted by terms.

$D = \{\Delta, \square, \triangle, \nabla\}$ objects

happy $P_1 = \{\Delta\}$
male $P_2 = \{\square, \nabla\}$
female $P_3 = \{\Delta, \triangle\}$
Couple $P_4 = \{(\Delta, \nabla), (\nabla, \Delta), (\triangle, \square), (\square, \triangle)\}$

bestFriendOf $\left\{ \begin{array}{l} \Delta \rightarrow \Delta \\ \Delta \rightarrow \square \\ \square \rightarrow \square \\ \nabla \rightarrow \square \end{array} \right\}$

Sentences:
 \rightarrow Happy(alice)
 ① Happy(bob)
 ② \neg Happy(bob)

Happy(bestFriendOf(alice))

\rightarrow Happy(bestFriendOf(carol))

Couple(alice, bob)

\rightarrow Couple(alice, david)

\rightarrow Couple(carol, bestFriendOf(bob))

$\forall x. \text{Happy}(x)$

$\rightarrow \exists x. \text{Happy}(x)$

$S_{all} \quad \underline{S}$

$I_1 \models S$

$\underline{S \cup \textcircled{1}}$

S_1

$I_1 \not\models S_1$

$\textcircled{1} \vee \textcircled{2} \quad S_i$

$I_2 \models$



$S_i \not\models$

$I_2 \models$

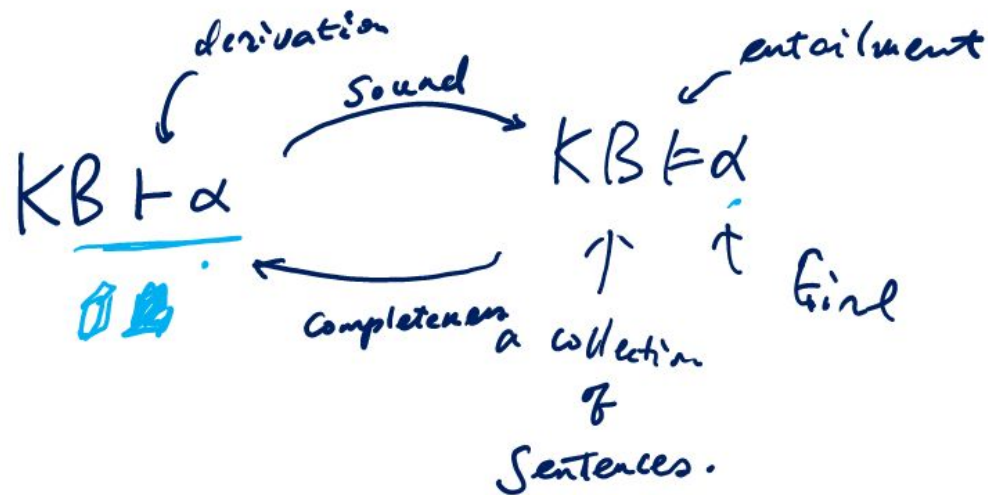
$\exists x. \text{Happy}(x)$

W5c10 lecture notes. Feb 8

propositional resolution. soundness and refutation completeness of resolution.

$$I \models \alpha. \quad \text{KB} \models \alpha$$

\uparrow \uparrow
 \neg \neg



th. we will first
consider propositional case

KB: clausal form.

a conjunction of clauses

disjunction of literals

$$\rightarrow (\underbrace{p \vee \neg q \vee r}_p) \wedge (\underbrace{\neg p \vee q \vee r}_{\neg p})$$

proposition / complement of a proposition.

p

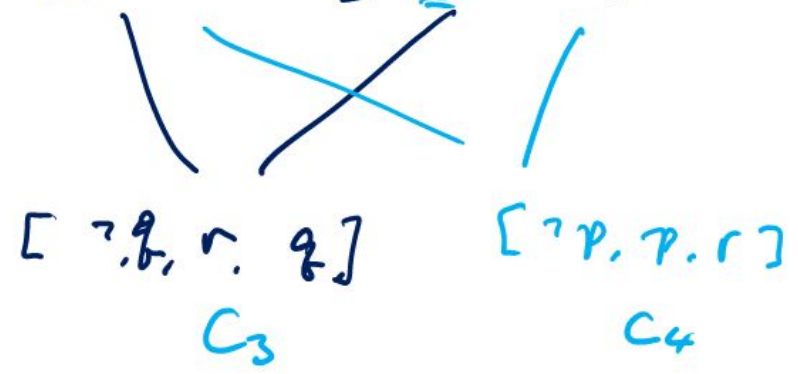
$\neg p$

$$KB_{\alpha} = \{ [\underbrace{p, \neg q, r}_p], [\neg p, q, r] \}$$

$$KB_{ex} = \{ [\underline{p}, \underline{q}, r], [\underline{p}, \underline{s}, r] \}.$$

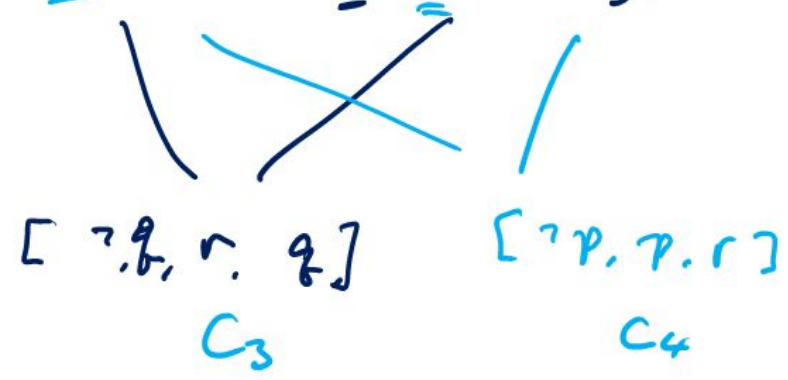
C_1 C_2

resolution.



$$KB_{ex} = \{ [\underline{p}, \underline{\neg q}, r], [\underline{\neg p}, \underline{q}, r] \}.$$

resolution.



special case.

$$KB_{unsat} \quad \frac{\{ \dots [p]^{c_1} . [\neg p]^{c_2} \}}{\underbrace{\quad}} \quad \bigvee \quad \underline{[]}$$

$$KB_{unsat} \vdash \underline{[]} \quad \Rightarrow \quad \text{empty clause}$$

$$KB_{unsat} \models \text{False} \quad \text{i.e.}$$

KB_{unsat} is indeed unsatisfiable.

on completeness.

bad news!

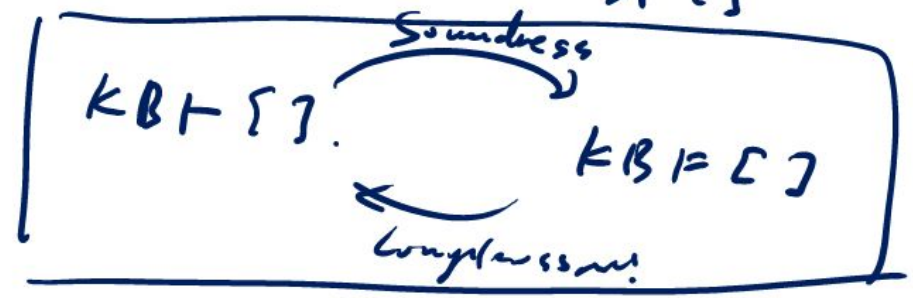
there exists $KB \models \alpha \not\rightarrow KB \vdash \alpha$

e.g. $\underbrace{\{\neg p\}}_{KB} \models \underbrace{[p, \neg p]}_{C_1}$ but $\neg p$?

good news, resolution is actually refutation complete!

$KB \models [] \rightarrow KB \vdash []$

if $\alpha \models []$.



getting back to the question.

How to prove that. $KB \models \alpha$.

Two steps:

KB'

① $KB \models \alpha \equiv \{KB \cup \neg \alpha\}$ is unsatisfiable.

② but $KB' \models []$ iff $KB' \vdash []$

" $KB \models \text{Girl}$ " on slide pg 53.

$\text{FirstGrade} \supset \text{Child} \equiv [\neg \text{FirstGrade}, \text{Child}]$.

$(\text{Child} \wedge \text{Male}) \supset \text{Boy} \equiv \neg (\text{Child} \wedge \text{male}) \vee \text{Boy}$

$\equiv \neg \text{child} \vee \neg \text{male} \vee \text{Boy} \equiv [\neg \text{child}, \neg \text{male}, \text{Boy}]$.