CS 228 Tutorial 1

2.5 Puzzle

$p_A \quad p_B \quad p_C$: are A, B, C resp good

Key: A statement someone makes is true iff they are good

$$\left(p_A \iff \neg p_A \wedge \neg p_B \wedge \neg p_C \right) \wedge$$

$$\left(p_B \iff (p_A \wedge \neg p_B \wedge \neg p_C) \vee (\neg p_A \wedge p_B \wedge \neg p_C) \vee (\neg p_A \wedge \neg p_B \wedge p_C) \right)$$

$\cancel{p_A \leftarrow T}$ ✗

$\cancel{p_B \leftarrow \bot}$ — $p_C \leftarrow T$ ✗

$p_A \leftarrow \bot$ — $p_B \leftarrow T$ — $p_C \leftarrow \bot$

must   must   must

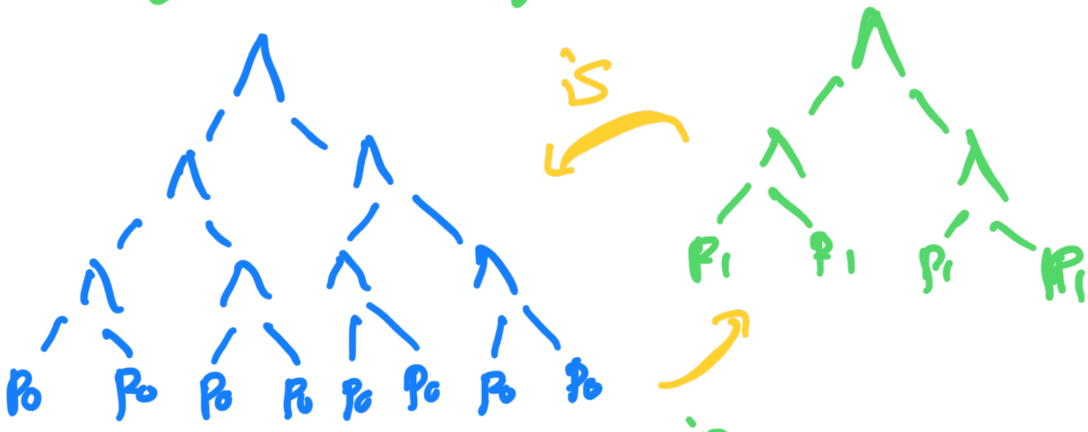
You will see this kind of elimination of possibilities and "backtracking" soon!

Foreshadowing DPLL/CDCL

2.7 The "let" expression.

Key: If you could compress

something, it has lots of redundant information.



let $p_1 = p_0 \wedge p_0$

let $p_2 = p_1 \wedge p_1$ in

is Δ

let $p_3 = p_2 \wedge p_2$ in p_3 is Δ

$\underbrace{\text{let } p_1 = p_0 \wedge p_0 \text{ in}}_{\text{only this much added for every level}}$ (let $p_2 = p_1 \wedge p_1$ in (let $p_3 = p_2 \wedge p_2$ in p_3))

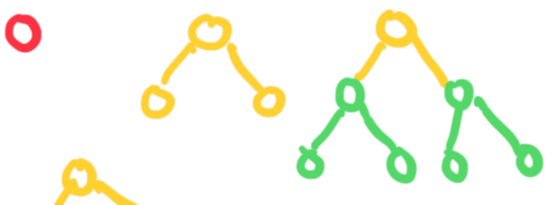
$\Delta \equiv$ let $p_1 = p_0 \wedge p_0$ in Δ

$\Delta \equiv$ let $p_2 = p_1 \wedge p_1$ in Δ

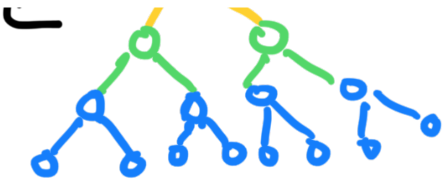
$\Delta \equiv$ let $p_3 = p_2 \wedge p_2$ in p_3

plug everything in

nesting)



in to out



3.13 Semantic entailment. $\models$

The definition of a model.

$F[\bot/p] \wedge F[\top/p]$ *

"$\bot$ for p"

i) Show that if $m \models$ *

then $m \models F$

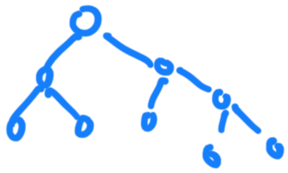
i) $m \models F[\bot/p]$

ii) $m \models F[\top/p]$

Key: a model makes an assignment to EVERYTHING in Vars.

In particular, what is $m(p)$

$m(p)$ is $\bot$ $\quad$ $m(p)$ is $\top$



$F[\bot/p] \longrightarrow F$

$\underbrace{m:\ p \leftarrow \bot}$

this is the only place parse trees differ

But under assignment, evaluation at every node is the same

Principle: In prop. logic, p must necessarily be true or false under a model:

$m' \models F$

$m'(p)$ is $\bot$ $\quad$ $m'(p)$ is $\top$

$m' \models F[\bot/p]$ $\quad$ $m' \models F[\top/p]$

$$m' \models F[\bot/p] \vee F[\top/p]$$

Foreshadowing: Law of Excluded Middle, Resolution

3.23 Structural Induction

Either atom p or its negation $\neg p$ is excluded from Σ. BASE CASE.

Show that Σ is satisfiable, given the bunch of rules

* Suppose there is H that proposed model doesn't satisfy. Let H be the smallest possible.

if $H = \neg\neg F$ × (F smaller)

$F \wedge G$ × (either F or G smaller counter)

$F \vee G$ × (both F and G bad & smaller).

$\neg(F \vee G)$

formula $F \vee G$ is satisfied

F is satisfied or
G is satisfied

But $\neg F$ and $\neg G$ occur in Σ

$\neg(F \wedge G)$

$F \wedge G$ satisfied

$\neg F$ or $\neg G$ occurs
these are smaller unsatisfied

Proposed model.

if neither p nor $\neg p$ occur in Σ,
then $m(p) \leftarrow T$ (arbit)

else exactly one of them occur.

if p $m(p) \leftarrow T$
else $m(p) \leftarrow \bot$

Base case: atomic formulae are satisfied by assignment.

3.15 Expressive Power

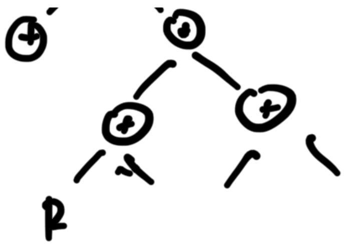
$\neg(F \oplus G) \equiv \neg F \oplus G$

(i)

truth table check.
works with atomic props, can lift to parse tree nodes.

$(F \oplus G) \oplus H \equiv F \oplus (G \oplus H)$ truth table check

↳  $p_1 \oplus p_2 \oplus p_3 \oplus \dots \oplus p_n$



if associative, then can flatten, and parenthesis arrangement doesn't matter.

(inductive, strong induction) Needs proof. Standard.

Use (i) to push the $\neg$ inside.
Use (ii) to flatten out

$p_1 \oplus \neg p_1 \equiv \top \qquad p_1 \oplus p_1 \equiv \bot$

$\top \oplus p_1 \equiv \neg p_1$

$\top, \bot, \quad l_1 \oplus l_2 \oplus l_3 \ldots$

l_n is $\neg p_n$ or p_n

Q. How many assignments to n variables satisfy in each case?

$2^n, 0,$

$p_1 \oplus \neg p_2 \oplus p_3 \ldots \oplus p_k$

$2^{n-k} \cdot 2^{k-1} \cdot 1 = 2^{n-1}$

You cannot express

$p_1 \wedge p_2 \wedge p_3 \wedge \ldots \wedge p_n.$