

(finish Planning)

Propositional Logic Intro, Syntax

Computer Science cpsc322, Lecture 19

(Textbook Chpt 5.1- 5.1.1 – 5.2)



Oct, 19, 2012

CPSC 322, Lecture 19

Slide 1

Lecture Overview

- **Recap Planning**
- Logic Intro
- Propositional Definite Clause Logic:
Syntax

Recap Planning

- Represent possible actions with STRIPS
- Plan can be found by..... search
- Or can be found by mapping planning problem into... CSP

Solve planning as CSP: pseudo code

horizon = 0 ; solved = false

while not solved

→ map STRIPS to CSP with horizon

solve CSP → solution

if solution found then

solved = true

else

horizon = horizon + 1

return solution



STRIPS to CSP applet

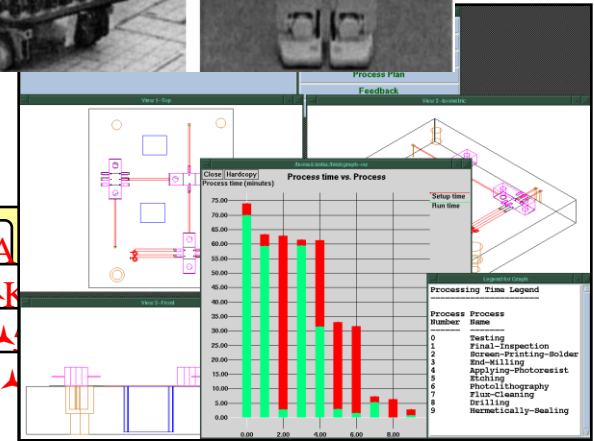
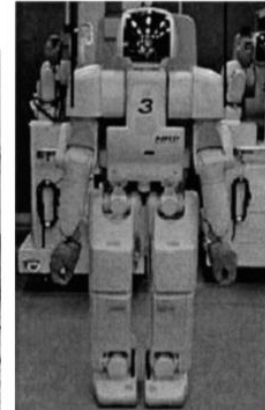
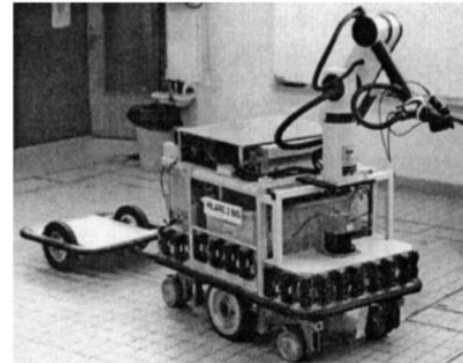
Allows you:

- to specify a planning problem in STRIPS ↩
- to map it into a CSP for a given horizon ↩
- the CSP translation is automatically loaded into the CSP applet where it can be solved

Practice exercise using STRIPS to CSP is available on AIspace

Now, do you know how to implement a planner for....

- Emergency Evacuation? ↩
- Robotics?
- Space Exploration?
- Manufacturing Analysis?
- Games (e.g., Bridge)?
- Generating Natural language ↩
- Product Recommendations



Active Sales Assistant™ personalized product recommendations from smart virtual sales assistants.

SHOPPERS

These virtual sales assistants give you the best product recommendations based on your preferences, for free.

You get: Recommendations ranked from best fit to worst, plus prices from leading retailers.

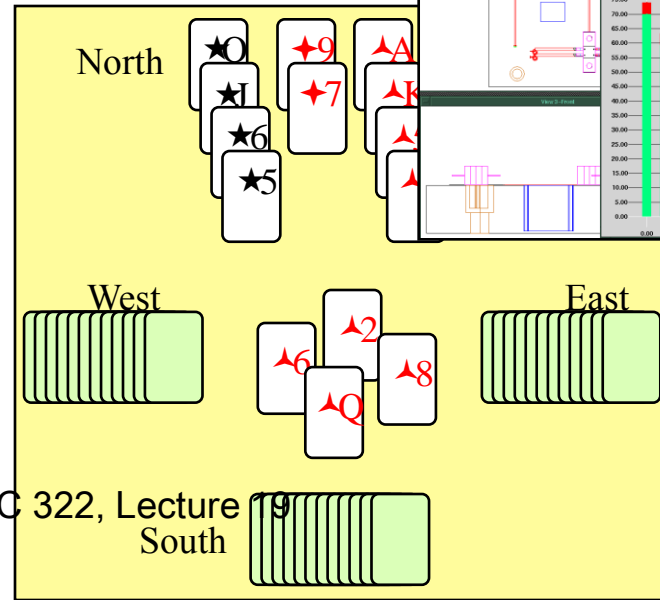
compare * red means you didn't want that feature but the product may still be a very good fit otherwise

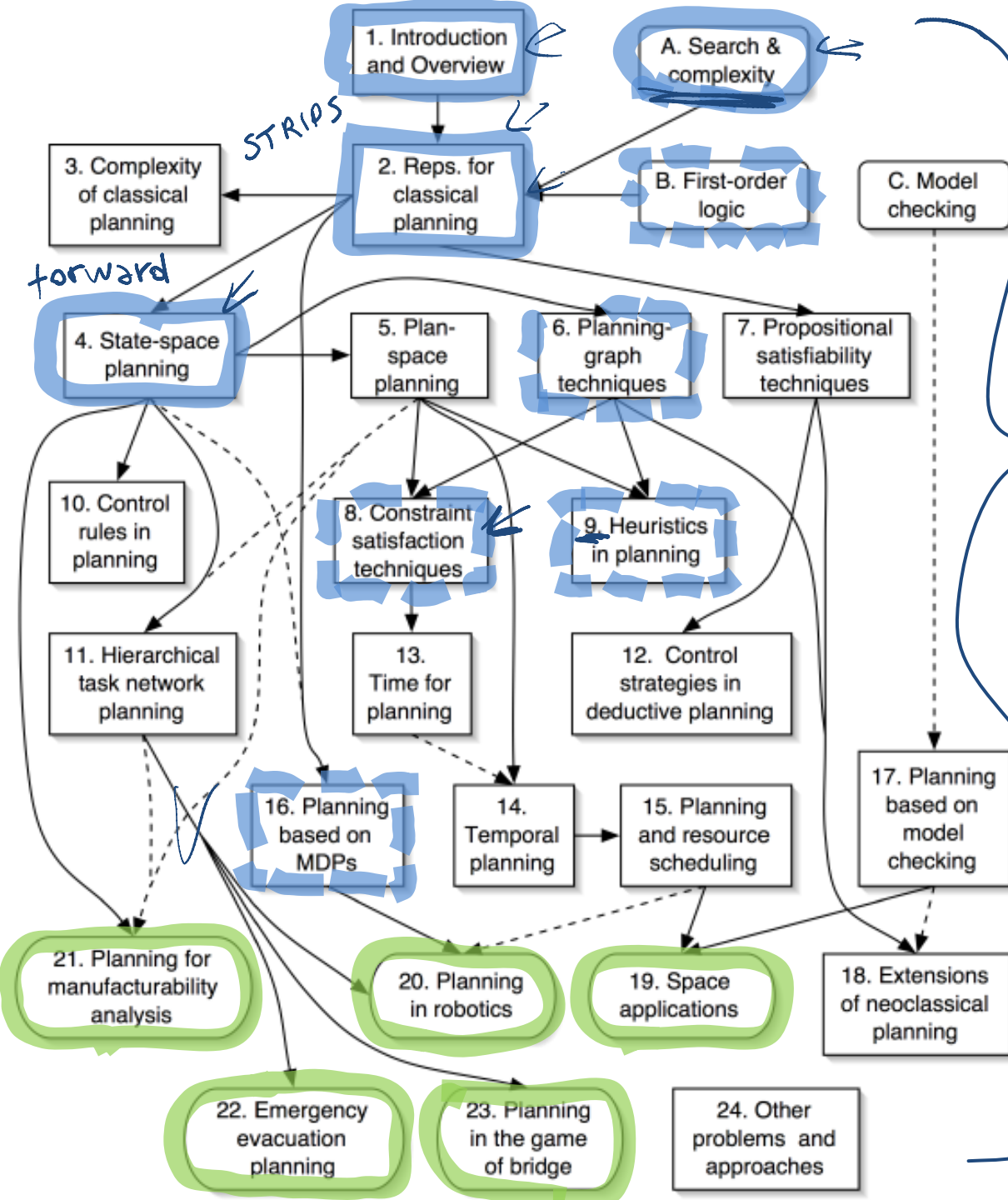
Rank	Brand & Model	Avg. Street Price	Optical Zoom	Resolut
1	Toshiba SD-275	\$240.00	3X	1792
2	Onkyo DV-S555	WHERE TO BUY	3X	1200
3	Sony DVP-F21	WHERE TO BUY	3X	1200

BUSINESSES

Increase sales on your site with Active Sales Assistant! Our clients typically double their sales conversion rates.

Free report the top 5 secrets to great online selling





book chapters

No ☹, but you
(will) know the
key ideas 😊!

- Ghallab, Nau, and Traverso
*Automated Planning:
Theory and Practice*
Morgan Kaufmann, May
2004
ISBN 1-55860-856-7
- Web site:
✓ <http://www.laas.fr/planning>

— you know

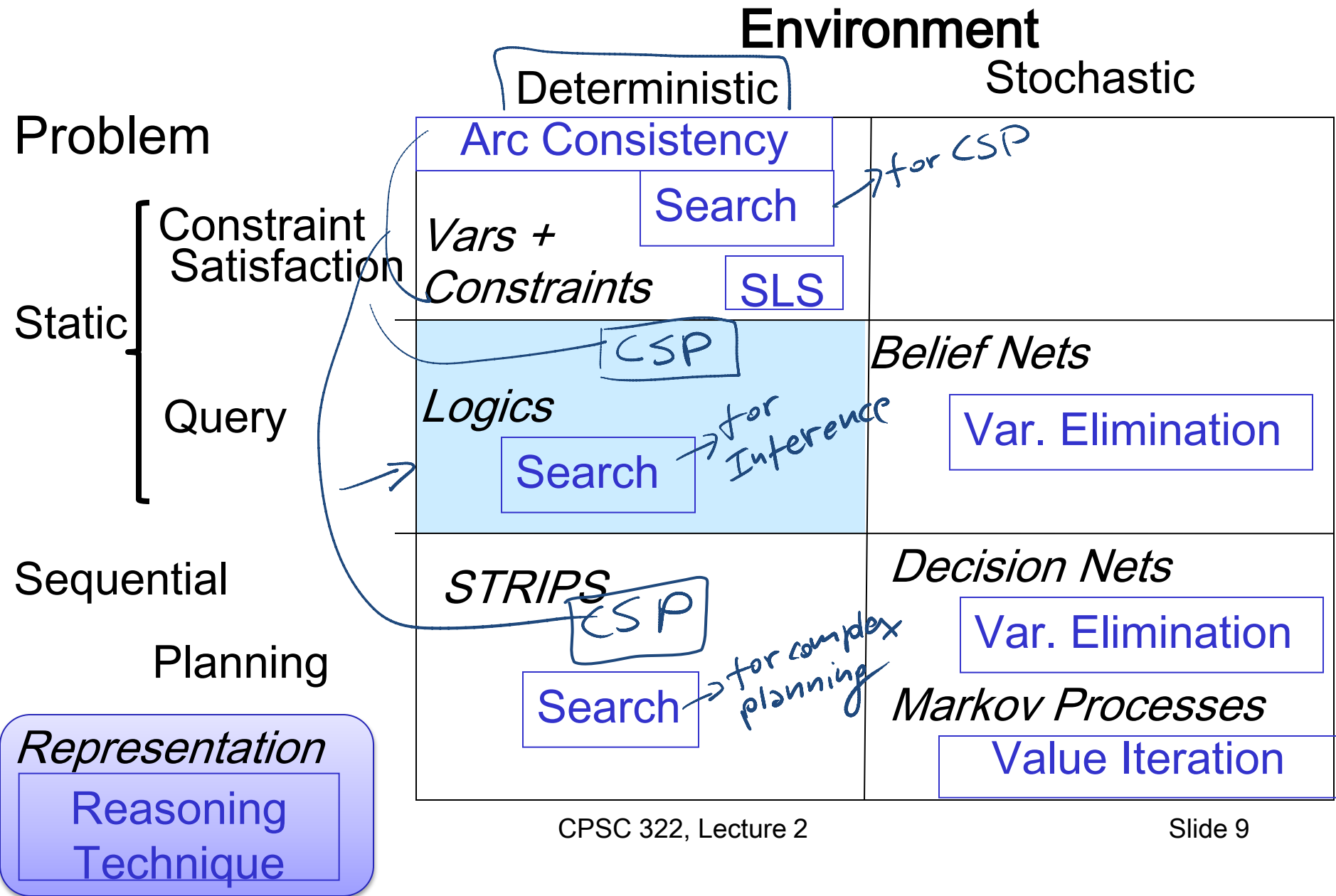
— you know a little

Applications

Lecture Overview

- Recap Planning
- **Logic Intro**
- Propositional Definite Clause Logic:
Syntax

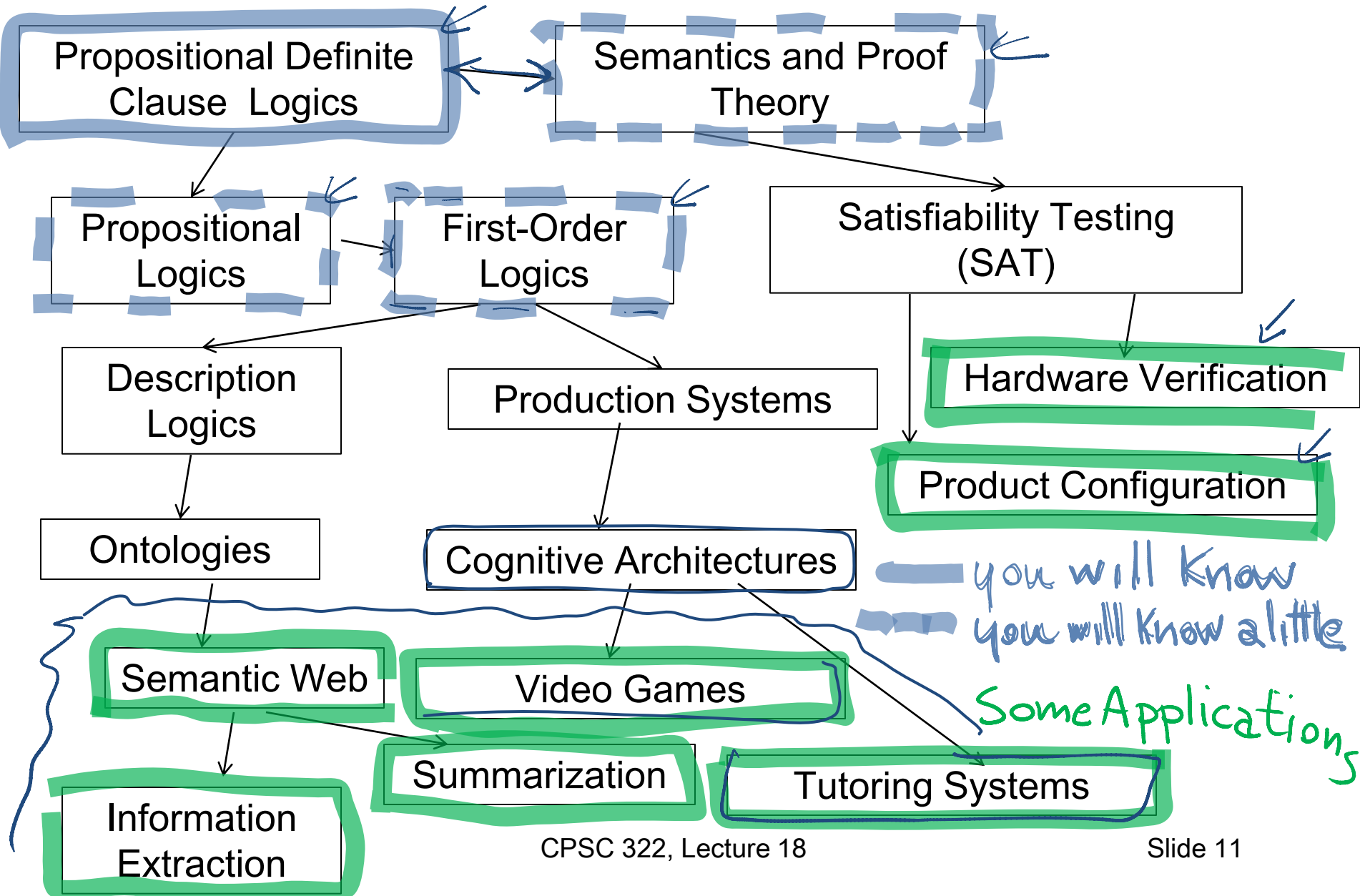
What is coming next ?



Logics

- **Mostly only propositional....** This is the starting point for more complex ones
- **Natural** to express **knowledge** about the world
 - What is true (boolean variables)
 - How it works (logical formulas)
- Well understood formal properties
- Boolean nature can be exploited for efficiency
-

Logics in AI: Similar slide to the one for planning



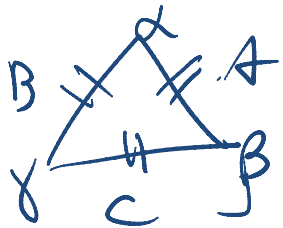
What you already know about logic...

From programming: Some logical operators

If ((amount > 0) $\&\&$ (amount < 1000)) $\vee\vee$ \neg (age < 30)
...
AND OR NOT
 $\wedge \vee \neg, \neg$

You know what they mean in a “procedural” way

Logic is the language of Mathematics. To define formal structures (e.g., sets, graphs) and to proof statements about those



$\forall x \text{ triangle}(x) \quad A=B=C \Leftrightarrow \alpha=\beta=\gamma$

We are going to look at Logic as a **Representation and Reasoning System** that can be used to formalize a domain (e.g., an electrical system, an organization) and to reason about it

Logic: A general framework for representation & reasoning

- Let's now think about **how to represent an environment** about which we have only partial (but certain) information
- What do we need to represent?

objects

events

actions

space

time

Why Logics?

- “**Natural**” to express **knowledge** about the world
(more natural than a “flat” set of variables & constraints)

“Every 322 student will pass the midterm”

Midterm(m_1)

Course(c_1)

Name-of($c_1, 322$)

Course-of(m_1, c_1)

$\bigwedge \text{Follows_advice}(z, \text{Slide } 23)$
 $\forall z \text{ Student}(z) \wedge \text{Registered}(z, c_1)$
 $\Rightarrow \text{pass}(m_1, z)$

- It is easy to **incrementally** add knowledge
- It is easy to **check** and **debug** knowledge
- Provide language for **asking complex queries**
- Well understood **formal properties**

Propositional Logic

We will study the simplest form of Logic: Propositional

- The primitive elements are propositions: Boolean variables that can be $\{true, false\}$
 p_1 p_2
- The goal is to illustrate the basic ideas
- This is a starting point for more complex logics (e.g., first-order logic)
- Boolean nature can be exploited for efficiency.

Propositional logic: Complete Language

The **proposition** symbols $p_1, p_2 \dots$ etc are sentences

- If S is a sentence, $\neg S$ is a sentence (**negation**)
- If S_1 and S_2 are sentences, $S_1 \wedge S_2$ is a sentence (**conjunction**)
- If S_1 and S_2 are sentences, $S_1 \vee S_2$ is a sentence (**disjunction**)
- If S_1 and S_2 are sentences, $S_1 \Rightarrow S_2$ is a sentence (**implication**)
- If S_1 and S_2 are sentences, $S_1 \Leftrightarrow S_2$ is a sentence (**biconditional**)

Sample Formula

$$((p_1 \vee p_2) \wedge p_3) \Leftrightarrow ((p_2 \Rightarrow \neg p_4) \vee p_5)$$

Propositional Logics in practice

- Agent is told (perceives) some facts about the world
some propositions are true

- Agent is told (already knows / learns) how the world works
logical formulas

- Agent can answer yes/no questions about whether other facts must be true

Using Logics to make inferences...

- 1) Begin with a **task domain**.
- 2) Distinguish those things you want to talk about (the ontology).

SEE NEXT
SLIDE

- 3) Choose symbols in the computer to **denote propositions**

$live_w_6$ sw_2-on

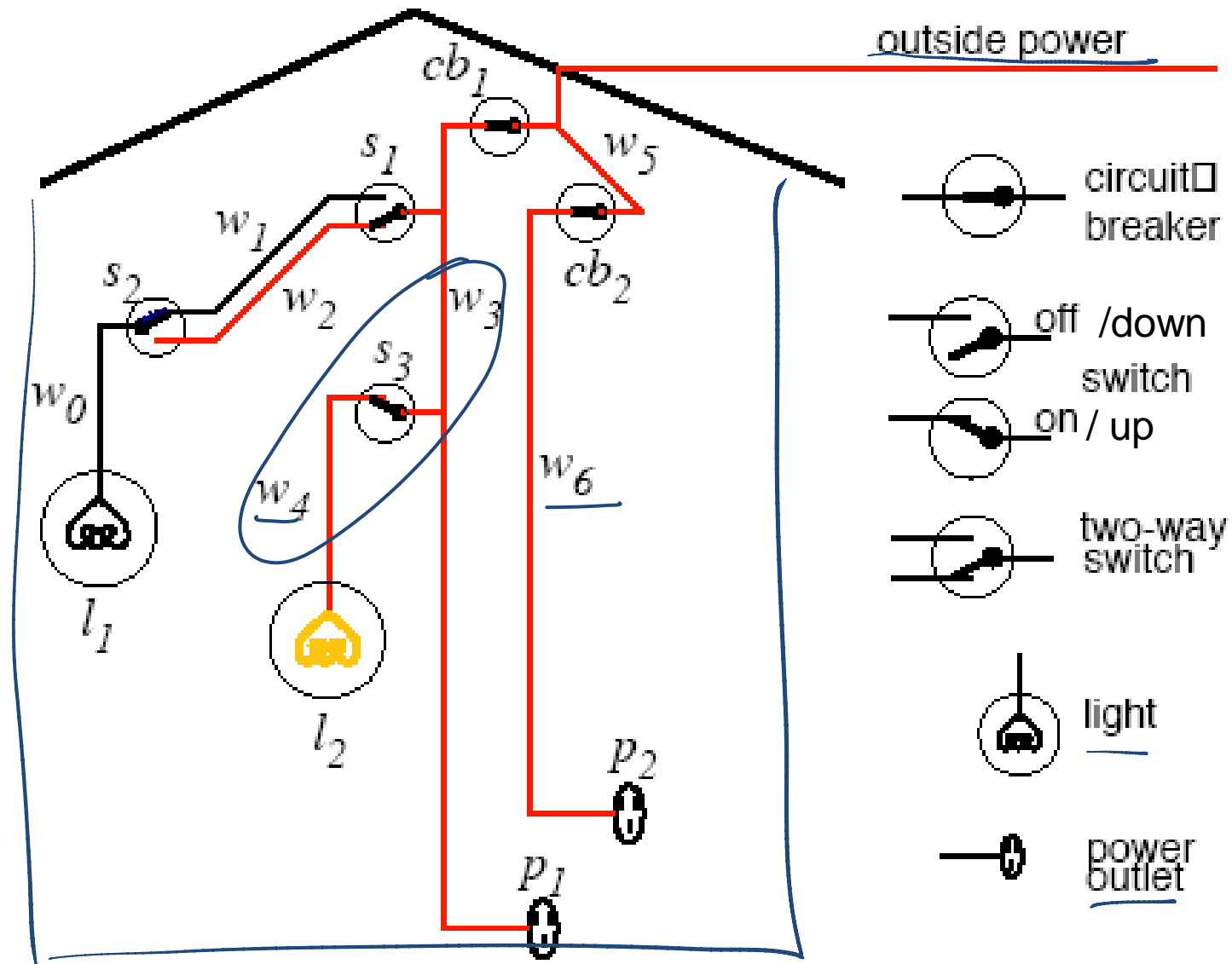
- 4) Tell the system **knowledge** about the domain. 

$live_w_3 \wedge sw_3-on \rightarrow live_w_4$

- 5) Ask the system whether new statements about the domain are true or false.

$l_2-on ?$

Electrical Environment



Lecture Overview

- Recap Planning
- Logic Intro
- **Propositional Definite Clause Logic:
Syntax**

Propositional Definite Clauses

- **Propositional Definite Clauses:** our first logical representation and reasoning system.
(very simple!)
- Only two kinds of statements:
 - that a proposition is true P_1
 - that a proposition is true if one or more other propositions are true P_2
 $P_1 \leftarrow P_3 \wedge P_4$
- Why still useful?
 - Adequate in many domains (with some adjustments)
 - Reasoning steps easy to follow by humans \leftarrow
 - Inference linear in size of your set of statements \leftarrow
 - Similar formalisms used in cognitive architectures \leftarrow

Propositional Definite Clauses: Syntax

Definition (atom)

An **atom** is a symbol starting with a lower case letter p_1

Definition (body)

A **body** is an atom or is of the form $b_1 \wedge b_2$ where b_1 and b_2 are bodies. $p_2 \wedge \dots \wedge p_n$

Definition (definite clause)

A **definite clause** is an atom or is a rule of the form $h \leftarrow b$ where h is an atom and b is a body. (Read this as " h if b .") $p_1 \leftarrow p_2 \wedge \dots \wedge p_5$

Definition (KB)

A **knowledge base** is a set of definite clauses $\text{clause}_1, \dots, \text{clause}_n$

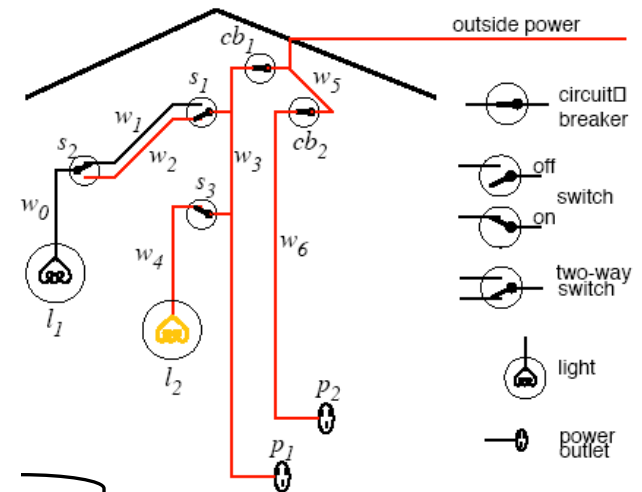
definite
clauses,
KB

light_l1.
light_l2.
ok_l1.
ok_l2.
ok_cb1.
ok_cb2.
live_outside.

atoms

live_l1 \leftarrow *live_w0*.
live_w0 \leftarrow *live_w1* \wedge *up_s2*.
live_w0 \leftarrow *live_w2* \wedge *down_s2*.
live_w1 \leftarrow *live_w3* \wedge *up_s1*.
live_w2 \leftarrow *live_w3* \wedge *down_s1*.
live_l2 \leftarrow *live_w4*.
live_w4 \leftarrow *live_w3* \wedge *up_s3*.
live_p1 \leftarrow *live_w3*.
live_w3 \leftarrow *live_w5* \wedge *ok_cb1*.
live_p2 \leftarrow *live_w6*.
live_w6 \leftarrow *live_w5* \wedge *ok_cb2*.
live_w5 \leftarrow *live_outside*.
lit_l1 \leftarrow *light_l1* \wedge *live_l1* \wedge *ok_l1*.
lit_l2 \leftarrow *light_l2* \wedge *live_l2* \wedge *ok_l2*.

rules



PDC Syntax: more examples















Definition (definite clause)

A **definite clause** is

- an atom or
- a rule of the form $h \leftarrow b$ where h is an atom ('head') and b is a body.
(Read this as ' h if b .')

Legal PDC clause








Not a legal PDC clause

- a) ai_is_fun  
- b) $ai_is_fun \vee ai_is_boring$  
- c) $ai_is_fun \leftarrow learn_useful_techniques$  
- d) $ai_is_fun \leftarrow learn_useful_techniques \wedge notTooMuch_work$  
- e) $ai_is_fun \leftarrow learn_useful_techniques \wedge \neg TooMuch_work$  
- f) $ai_is_fun \leftarrow f(time_spent, material_learned)$  
- g) $srtisyj \leftarrow errt \wedge gffdgdgd$  

PDC Syntax: more examples

Legal PDC clause

Not a legal PDC clause

- a) ai_is_fun 
- b) $ai_is_fun \vee ai_is_boring$ 
- c) $ai_is_fun \leftarrow learn_useful_techniques$ 
- d) $ai_is_fun \leftarrow learn_useful_techniques \wedge notTooMuch_work$ 
- e) $ai_is_fun \leftarrow learn_useful_techniques \wedge \neg TooMuch_work$ 
- f) $ai_is_fun \leftarrow f(time_spent, material_learned)$ 
- g) $srtisyj \leftarrow errt \wedge gffdgdgd$ 

Do any of these statements **mean** anything?
Syntax doesn't answer this question!

Learning Goals for today's class

You can:

- Verify whether a logical statement belongs to the language of full propositional logics.
- Verify whether a logical statement belongs to the language of propositional definite clauses.

Study for midterm (Mon Oct 29)

Midterm: ~6 short questions (10pts each) + 2 problems (20pts each)

1 or 2 on Logics

- Study: textbook and inked slides
- Work on all practice exercises and revise assignments!
- While you revise the learning goals, work on review questions (will post them tomorrow)- I may even reuse some verbatim 😊
- Will post a couple of problems from previous offering (maybe slightly more difficult) ... but I'll give you the solutions 😊

search CSP

Next class

- Definite clauses Semantics and Proofs
(textbook 5.1.2, 5.2.2)