Department of Computer Science
Undergraduate Events

More details @ https://www.cs.ubc.ca/students/undergrad/life/upcoming-events

CS Co-op Q&A Session

Date: Thurs., Oct 24
Time: 1 – 2 pm
Location: Reboot Cafe

CSSS Movie Night: Gravity

Date: Fri., Oct 25
Time: ~ 7:45 pm
Location: Scotiabank Theatre

Mastering LinkedIn Workshop

Date: Mon., Oct 28
Time: 5:00 pm
Location: Wesbrook 100

Graduate Recruitment Panel

Date: Wed., Oct 30
Time: 12:30 – 1:30 pm
Location: X836, ICICS/CS

CSSS Meet the Profs Luncheon

Date: Thurs., Oct 31
Time: 12:30 – 2 pm
Location: X836, ICICS/CS

(finish Planning)

Propositional Logic Intro, Syntax

Computer Science cpsc322, Lecture 19

(Textbook Chpt 5.1- 5.1.1 – 5.2)

Oct, 21, 2013

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
```

Planning as CSP

If the algorithm for planning as CSP stops and returns a solution plan of length k, does it mean that there are no shorter solutions ?

[✓ A. Yes]

there are no shorter solutions

B. No

C. It depends …

STRIPS to CSP applet

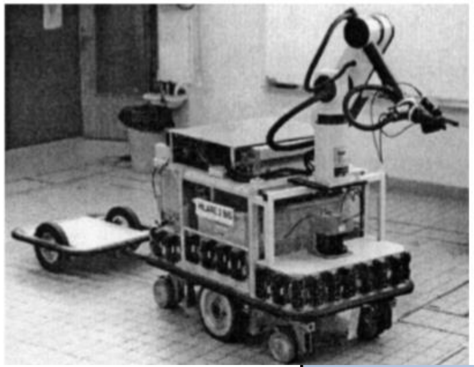
Allows you:

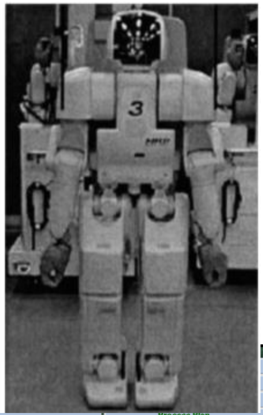
- to specify a planning problem in STRIPS
- to map it into a CSP for a given horizon
- the CSP translation is automatically loaded into the CSP applet where it can be solved

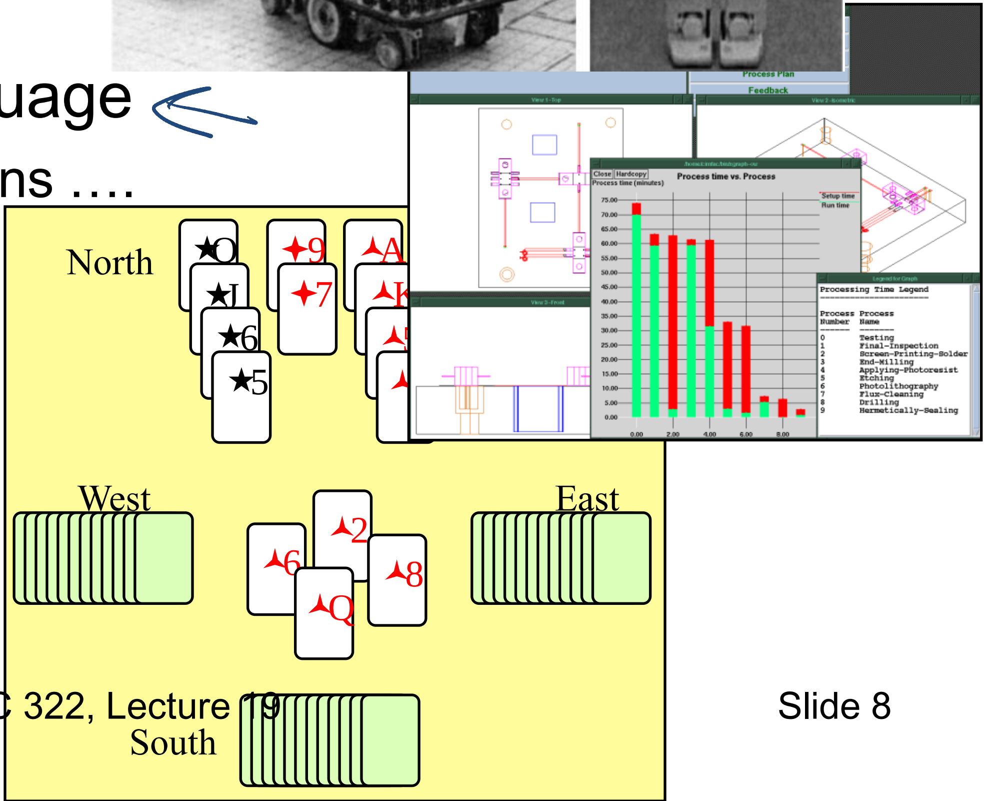
Practice exercise using STRIPS to CSP is available on AIspace

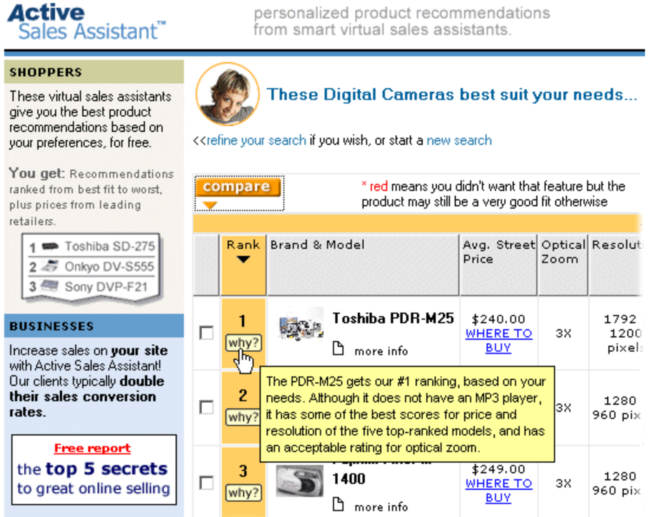
Now, do you know how to implement a planner for….

- 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.

1 Toshiba SD-275
2 Onkyo DV-S555
3 Sony DVP-F21

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

These Digital Cameras best suit your needs...

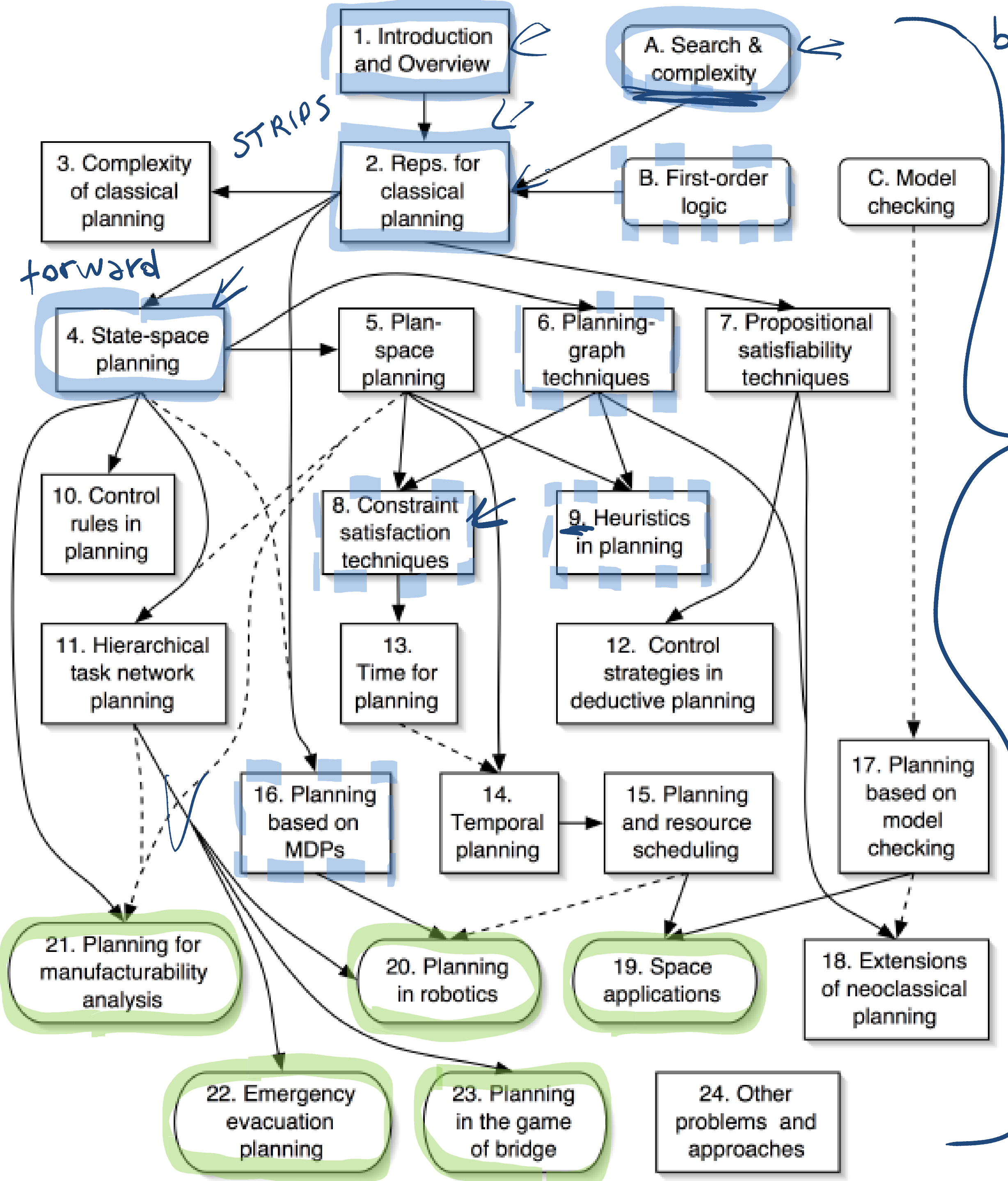
<<refine your search if you wish, or start a new search

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 why?	Toshiba PDR-M25 more info	$240.00 WHERE TO BUY	3X	1792 1200 pixel
□	2 why?			3X	1280 960 pix
□	3 why?	1400 more info	$249.00 WHERE TO BUY	3X	1280 960 pix

The PDR-M25 gets our #1 ranking, based on your needs. Although it does not have an MP3 player, it has some of the best scores for price and resolution of the five top-ranked models, and has an acceptable rating for optical zoom.



1. Introduction and Overview

A. Search & complexity

STRIPS

3. Complexity of classical planning

2. Reps. for classical planning

B. First-order logic

C. Model checking

forward

4. State-space planning

5. Plan-space planning

6. Planning-graph techniques

7. Propositional satisfiability techniques

10. Control rules in planning

8. Constraint satisfaction techniques

9. Heuristics in planning

11. Hierarchical task network planning

13. Time for planning

12. Control strategies in deductive planning

16. Planning based on MDPs

14. Temporal planning

15. Planning and resource scheduling

17. Planning based on model checking

21. Planning for manufacturability analysis

20. Planning in robotics

19. Space applications

18. Extensions of neoclassical planning

22. Emergency evacuation planning

23. Planning in the game of bridge

24. Other problems and approaches

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

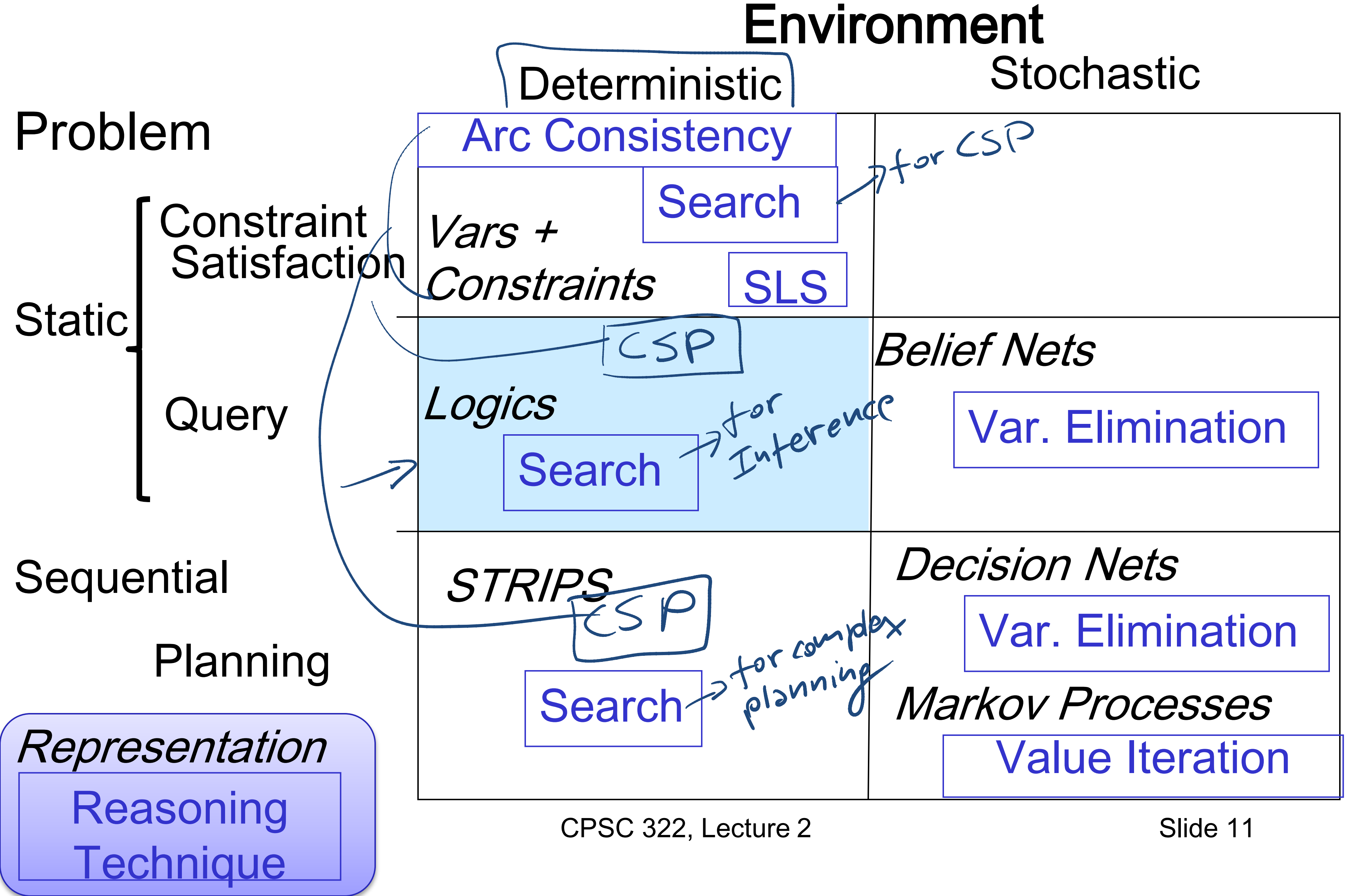
▪▪▪ you know a little

Applications

Lecture Overview

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

What is coming next ?



Environment

Problem		Deterministic	Stochastic
Static	Constraint Satisfaction	*Vars + Constraints* Arc Consistency Search → for CSP SLS	
	Query	*Logics* CSP Search → for Inference	*Belief Nets* Var. Elimination
Sequential	Planning	*STRIPS* CSP Search → for complex planning	*Decision Nets* Var. Elimination *Markov Processes* Value Iteration

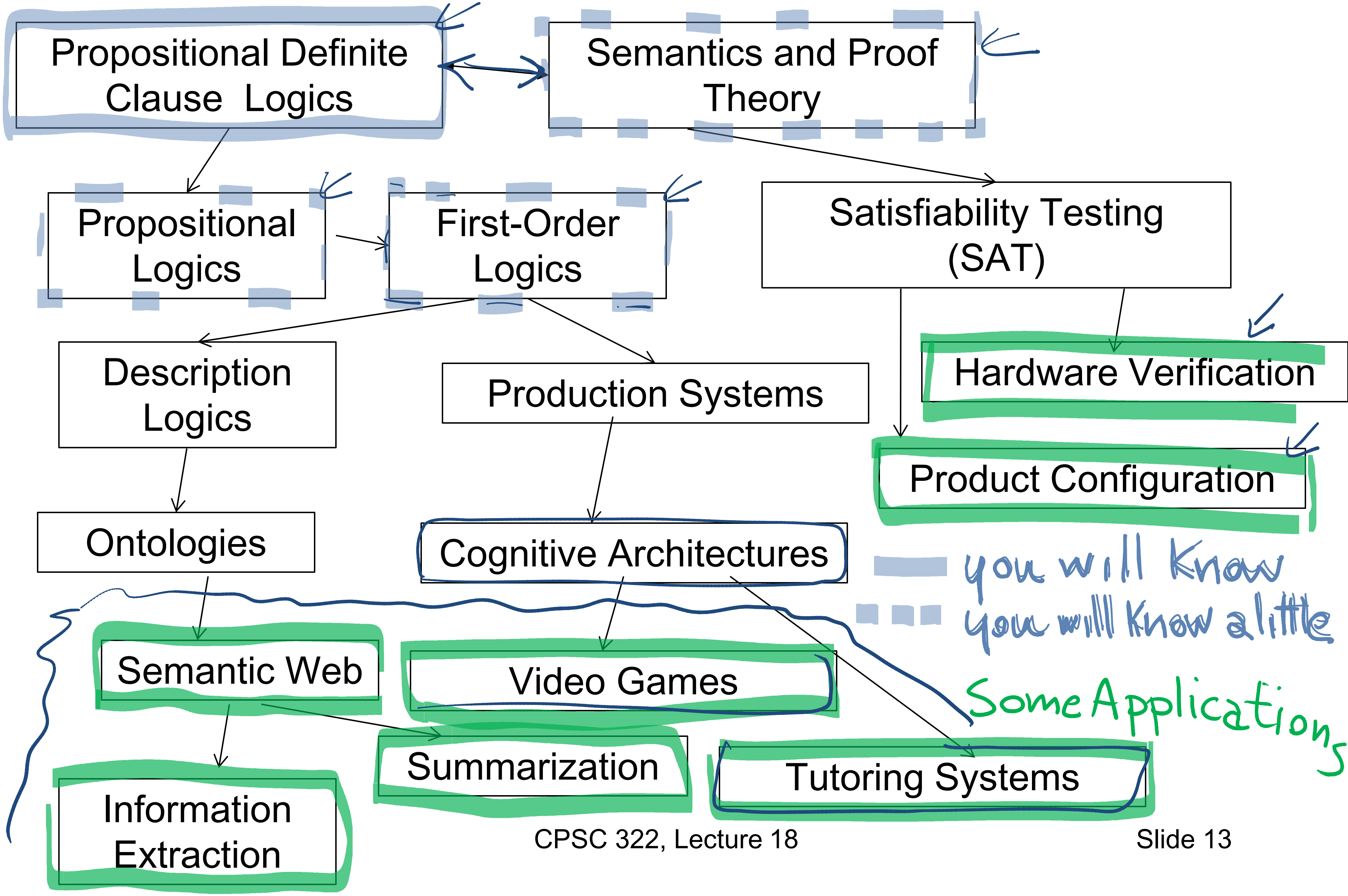
Representation

Reasoning Technique

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



Propositional Definite Clause Logics

Semantics and Proof Theory

Propositional Logics

First-Order Logics

Satisfiability Testing (SAT)

Description Logics

Production Systems

Hardware Verification

Product Configuration

Ontologies

Cognitive Architectures

Semantic Web

Video Games

Summarization

Tutoring Systems

Information Extraction

▬ you will know

▬ ▬ ▬ you will know a little

Some Applications

What you already know about logic...

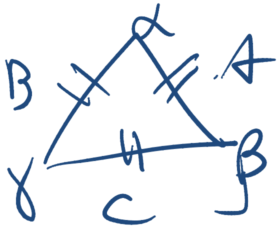
From programming: Some logical operators

```
If ((amount > 0) && (amount < 1000)) || !(age < 30)
...
```

AND $\wedge$ OR $\vee$ NOT $\sim, \neg$

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



$\underset{\exists}{\forall} x$ triangle(x) $A=B=C \Longleftrightarrow \alpha=\beta=\gamma$

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"

$\mathrm{Midterm}(m_1)$

$\mathrm{Course}(c_1)$

$\mathrm{Name\text{-}of}(c_1, 322)$

$\mathrm{Course\text{-}of}(m_1, c_1)$

$\forall z\ \mathrm{Student}(z) \wedge \mathrm{Registred}(z, c_1)$ $\wedge\ \mathrm{Follows\_advice}(z, \mathrm{Slide23})$

$\Rightarrow \mathrm{pass}(m_1, z)$

- 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

Complex Query

Student(Sue)

"will Sue pass all her midterms?"

$$\forall c, m \quad \text{registred}(\text{Sue}, c) \wedge \text{course-of}(m, c)$$

$$?\ \text{pass}(m, \text{Sue})$$

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 … etc are sentences

- If S is a sentence, $\neg\mathbf{S}$ is a sentence (negation)
- If S_1 and S_2 are sentences, $\mathbf{S_1} \underline{\wedge} \mathbf{S_2}$ is a sentence (conjunction)
- If S_1 and S_2 are sentences, $\mathbf{S_1} \underline{\vee} \mathbf{S_2}$ is a sentence (disjunction)
- If S_1 and S_2 are sentences, $\mathbf{S_1} \underline{\Rightarrow} \mathbf{S_2}$ is a sentence (implication)
- If S_1 and S_2 are sentences, $\mathbf{S_1} \underline{\Leftrightarrow} \mathbf{S_2}$ 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_1_on

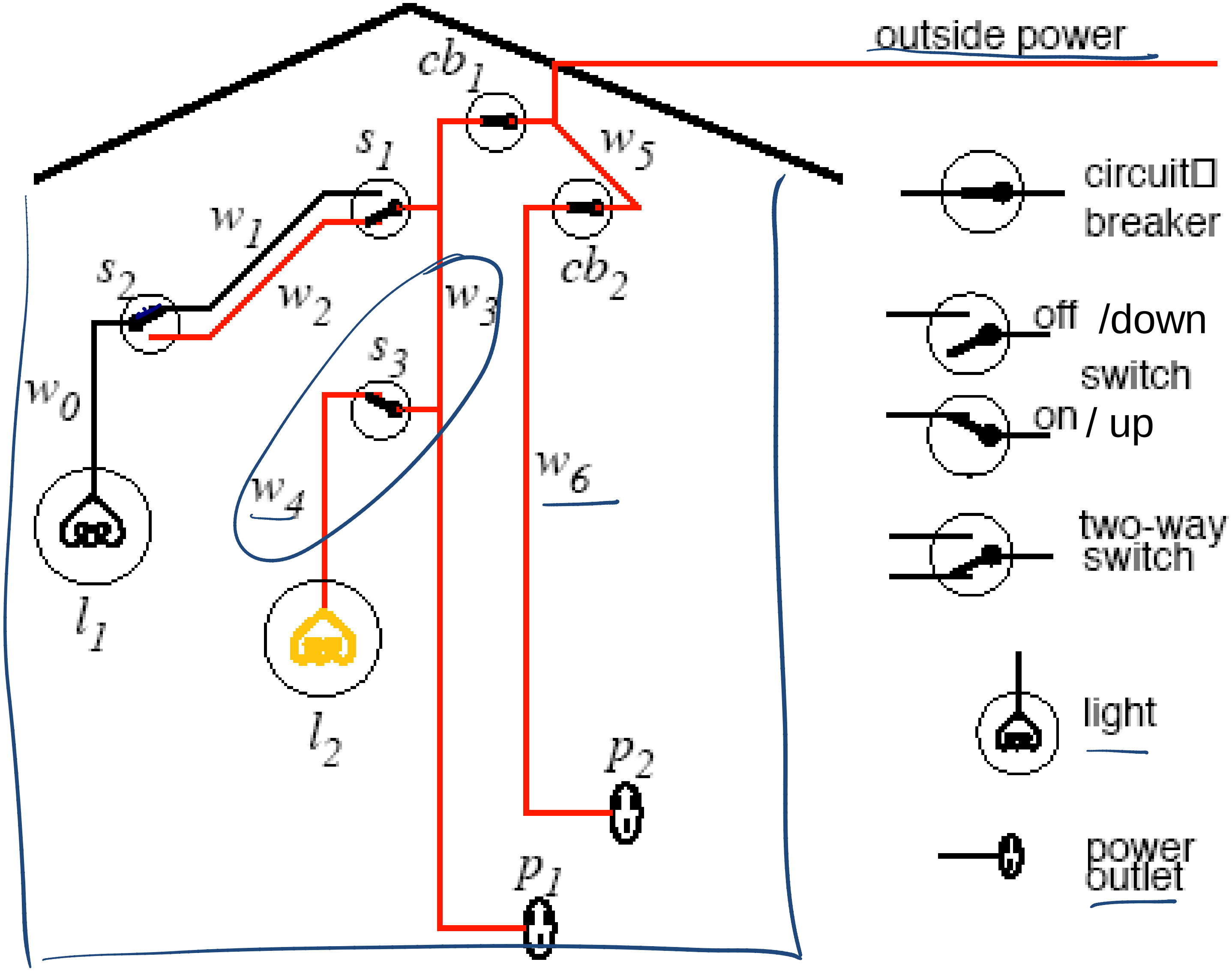
4) Tell the system **knowledge** about the domain.

$live\_w_3 \wedge sw_3\_on \rightarrow live\_w_4$

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
  - that a proposition is true if one or more other propositions are true

$p_1 \quad p_2$

$p_1 \leftarrow p_3 \wedge p_4$

- Why still useful?
  - Adequate in many domains (with some adjustments)
  - Reasoning steps easy to follow by humans
  - Inference linear in size of your set of statements
  - Similar formalisms used in cognitive architectures

Propositional Definite Clauses: Syntax

Definition (atom)

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

Definition (body)

$p_1 \wedge \cdots \wedge p_n$

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

Definition (definite clause)

$p_1 \leftarrow \overbrace{p_2 \wedge . p_5}$

A definite clause is an atom or is a rule of the form $\boxed{h \leftarrow b}$ where h is an atom and b is a body. (Read this as ``h if b.")

Definition (KB)

A knowledge base is a set of definite clauses

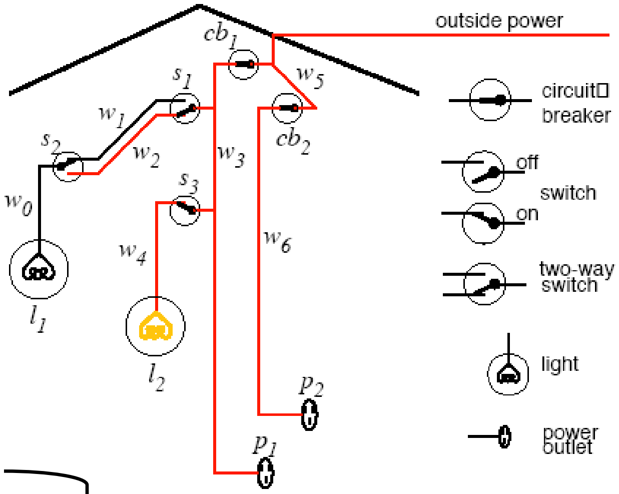
$clause_1$

$\vdots$

$clause_n$

$light\_l_1.$
$light\_l_2.$
$ok\_l_1.$
$ok\_l_2.$
$ok\_cb_1.$
$ok\_cb_2.$
$live\_outside.$

atoms



definite clauses, KB

$live\_l_1 \leftarrow live\_w_0.$
$live\_w_0 \leftarrow live\_w_1 \wedge up\_s_2.$
$live\_w_0 \leftarrow live\_w_2 \wedge down\_s_2.$
$live\_w_1 \leftarrow live\_w_3 \wedge up\_s_1.$
$live\_w_2 \leftarrow live\_w_3 \wedge down\_s_1.$
$live\_l_2 \leftarrow live\_w_4.$
$live\_w_4 \leftarrow live\_w_3 \wedge up\_s_3.$
$live\_p_1 \leftarrow live\_w_3.$
$live\_w_3 \leftarrow live\_w_5 \wedge ok\_cb_1.$
$live\_p_2 \leftarrow live\_w_6.$
$live\_w_6 \leftarrow live\_w_5 \wedge ok\_cb_2.$
$live\_w_5 \leftarrow live\_outside.$
$lit\_l_1 \leftarrow light\_l_1 \wedge live\_l_1 \wedge ok\_l_1.$
$lit\_l_2 \leftarrow light\_l_2 \wedge live\_l_2 \wedge ok\_l_2.$

rules

PDC Syntax: more examples

Definition (definite clause)

A **definite clause** is

- an atom or
- a rule of the form *h* ← *b* where *h* is an atom (‘head’) and *b* is a body. (Read this as ‘*h* if *b*.’)

A. Legal B. Not Legal

a) ai_is_fun

b) ai_is_fun ∨ ai_is_boring

c) ai_is_fun ← learn_useful_techniques

d) ai_is_fun ← learn_useful_techniques ∧ notTooMuch_work

e) ai_is_fun ← learn_useful_techniques ∧ ¬ TooMuch_work

f) ai_is_fun ← f(time_spent, material_learned)

g) srtsyj ← errt ∧ gffdgdgd

PDC Syntax: more examples

🟩 Legal PDC clause

🟨 Not a legal PDC clause

a) ai_is_fun 🟩

b) ai_is_fun ∨ ai_is_boring 🟨

c) ai_is_fun ← learn_useful_techniques 🟩

d) ai_is_fun ← learn_useful_techniques ∧ notTooMuch_work 🟩

e) ai_is_fun ← learn_useful_techniques ∧ ¬ TooMuch_work 🟨

f) ai_is_fun ← f(time_spent, material_learned) 🟨

g) srtsyj ← errt ∧ 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 28)

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

1 ~~or 2~~ on Logics

search CSP

- 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 ☺

Next class

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