**Department of Computer Science**  
**Undergraduate Events**

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

\[
\begin{align*}
\text{horizon} &= 0 ; \text{ solved } = \text{false} \\
\text{while} & \text{ not solved} \\
& \text{map STRIPS to CSP with horizon} \\
& \text{solve CSP } \rightarrow \text{ solution} \\
& \text{if solution found then} \\
& \quad \text{solved} = \text{true} \\
& \text{else} \\
& \quad \text{horizon} = \text{horizon} + 1 \\
\text{return} & \text{ solution}
\end{align*}
\]
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
B. No
C. It depends …

there are no shorter solutions
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 Alspace
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 ….
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](http://www.laas.fr/planning)
Lecture Overview

• Recap Planning

• Logic Intro

• Propositional Definite Clause Logic: Syntax
What is coming next?

Environment
- Deterministic
  - Arc Consistency
  - Search
  - Vars + Constraints
  - SLS
- Stochastic
  - Belief Nets
    - Var. Elimination
  - Decision Nets
    - Var. Elimination
    - Value Iteration

Problem
- Constraint Satisfaction
- Query

Static
- Search

Sequential
- Planning

Representation
- Reasoning Technique

CPSC 322, Lecture 2
Slide 11
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
- Ontologies
- Cognitive Architectures
- Product Configuration
- Semantic Web
- Video Games
- Summarization
- Tutoring Systems
- Information Extraction

You will know a little more.
What you already know about logic...

From programming: Some logical operators

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

... AND OR NOT

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

\[ \text{\textbf{A} \times \text{Triangle}(x)} \implies \text{A = B = C} \iff \gamma = \beta = \alpha \]

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
- 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"

- 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

"Will Sue pass all her midterms?"

\[ \forall c, m \text{ registered}(Sue, c) \land \text{course-of}(m, c) \land \text{pass}(m, Sue) \]
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 \( p_1, p_2 \ldots \) etc are sentences

- If \( S \) is a sentence, \( \neg S \) is a sentence (negation)
- If \( S_1 \) and \( S_2 \) are sentences, \( S_1 \land S_2 \) is a sentence (conjunction)
- If \( S_1 \) and \( S_2 \) are sentences, \( S_1 \lor S_2 \) 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, \( S_1 \iff S_2 \) is a sentence (biconditional)

Sample Formula

\[
((p_1 \lor p_2) \land p_3) \iff ((p_2 \Rightarrow \neg p_4) \lor p_5)
\]
Propositional Logics in practice

• Agent is told (perceives) some facts about the world

\[ \text{some propositions are true} \]

• Agent is told (already knows / learns) how the world works

\[ \text{logical formulas} \]

• 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).

3) Choose symbols in the computer to denote propositions

\[
\text{live}_w_6 \quad \text{sw} \_ \text{on}
\]

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

\[
\text{live}_w_3 \land \text{sw} \_ \text{on} \rightarrow \text{live}_w_4
\]

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

\[
\text{l}_2 \_ \text{on} \quad ?
\]
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

- 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

\[ p_1 \leq p_2 \]
\[ p_1 \leq p_3 \land p_4 \]
Propositional Definite Clauses: Syntax

Definition (atom)
An **atom** is a symbol starting with a lower case letter.

Definition (body)
A **body** is an atom or is of the form \( b_1 \land b_2 \) where \( b_1 \) and \( b_2 \) are bodies.

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 \( b \)."")

Definition (KB)
A **knowledge base** is a set of definite clauses.
Atoms:
- light_l1.
- light_l2.
- ok_l1.
- ok_l2.
- ok_cb1.
- ok_cb2.
- live_outside.

Definite clauses, KB:
- live_l1 ← live_w0.
- live_w0 ← live_w1 ∧ up_s2.
- live_w0 ← live_w2 ∧ down_s2.
- live_w1 ← live_w3 ∧ up_s1.
- live_w2 ← live_w3 ∧ down_s1.
- live_l2 ← live_w4.
- live_w4 ← live_w3 ∧ up_s3.
- live_p1 ← live_w3.
- live_w3 ← live_w5 ∧ ok_cb1.
- live_p2 ← live_w6.
- live_w6 ← live_w5 ∧ ok_cb2.
- live_w5 ← live_outside.
- lit_l1 ← light_l1 ∧ live_l1 ∧ ok_l1.
- lit_l2 ← light_l2 ∧ live_l2 ∧ ok_l2.

Rules:
- circuit breaker
- on
- off switch
- two-way switch
- power outlet
- outside power
- light
- l1
- l2
- s1
- s2
- s3
- s4
- s5
- s6
- p1
- p2
- w0
- w1
- w2
- w3
- w4
- w5
- w6
- cb1
- cb2
- w_0
- w_1
- w_2
- w_3
- w_4
- w_5
- w_6
- cb_1
- cb_2
- s_1
- s_2
- s_3
- s_4
- s_5
- s_6
- p_1
- p_2
- l_1
- l_2
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 $b$.’)

a) $ai\_is\_fun$

b) $ai\_is\_fun \lor ai\_is\_boring$

c) $ai\_is\_fun \leftarrow \text{learn\_useful\_techniques}$

d) $ai\_is\_fun \leftarrow \text{learn\_useful\_techniques} \land \neg \text{TooMuch\_work}$

e) $ai\_is\_fun \leftarrow \text{learn\_useful\_techniques} \land \neg \text{TooMuch\_work}$

f) $ai\_is\_fun \leftarrow f(\text{time\_spent}, \text{material\_learned})$

g) $srtsyj \leftarrow errt \land gffdgddgd$

A. Legal    B. Not Legal
PDC Syntax: more examples

Legal PDC clause

- ai_is_fun

Not a legal PDC clause

- ai_is_fun \lor ai_is_boring
- ai_is_fun \leftarrow learn_useful_techniques
- ai_is_fun \leftarrow learn_useful_techniques \land notTooMuch_work
- ai_is_fun \leftarrow learn_useful_techniques \land \neg TooMuch_work
- ai_is_fun \leftarrow f(time_spent, material_learned)
- srtsyj \leftarrow errt \land gffdgdg}

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 \( (10pts \text{ each}) \) + 2 problems \( (20pts \text{ each}) \)

1. 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 😊
Next class

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