Lecture 19 of 42

Knowledge Representation and Midterm Review
Discussion: Search, Inference, Planning

William H. Hsu
Department of Computing and Information Sciences, KSU

KSOL course page: http://snipurl.com/v9v3
Course web site: http://www.kddresearch.org/Courses/CIS730
Instructor home page: http://www.cis.ksu.edu/~bhsu

Reading for Next Class:

Review Chapters 1 – 10, Russell & Norvig 2nd edition
Protégé-OWL tutorial: http://bit.ly/3rM1pB

Lecture Outline

- **Reading for Next Class: Review Chapters 1 - 10, R&N 2e**
- **Last Class: Event and Fluent Calculi, CIKM**
  - **Representing time, events: from situation calculus to event, fluent calculi**
  - **Knowledge acquisition (KA) and capture**
  - **Computational information and knowledge management (CIKM)**
- **Today: Midterm Review**
  - **Section I: Intelligent Agents**
  - **Section II: Search**
  - **Section III: Knowledge and Reasoning**
- **Coming Week: Intro to Classical Planning**

Problem-Solving Agents: Review

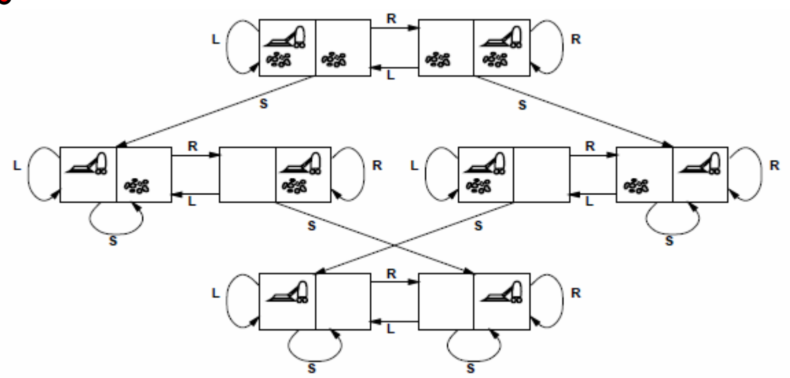
Restricted form of general agent:

```
function SIMPLE-PROBLEM-SOLVING-AGENT( percept) returns an action
   static: seq, an action sequence, initially empty
           state, some description of the current world state
           goal, a goal, initially null
           problem, a problem formulation

   state ← UPDATE-STATE(state, percept)
   if seq is empty then
       goal ← FORMULATE-GOAL(state)
       problem ← FORMULATE-PROBLEM(state, goal)
       seq ← SEARCH( problem)
   action ← RECOMMENDATION(seq, state)
   seq ← REMAINDER(seq, state)
   return action
```

Note: this is offline problem solving; solution executed "eyes closed."
Online problem solving involves acting without complete knowledge.

State Space Graph: Review



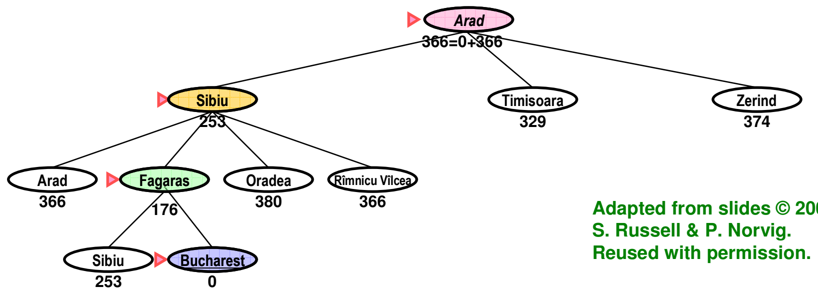
states??: integer dirt and robot locations (ignore dirt amounts etc.)

actions??: *Left*, *Right*, *Suck*, *NoOp*

goal test??: no dirt

path cost??: 1 per action (0 for *NoOp*)

Greedy Search: Review



CLOSED List

∅ ≡ {}

Arad

Arad Sibiu

Arad Sibiu Fagaras

Arad Sibiu Fagaras Bucharest

OPEN List

Arad$_{366}$

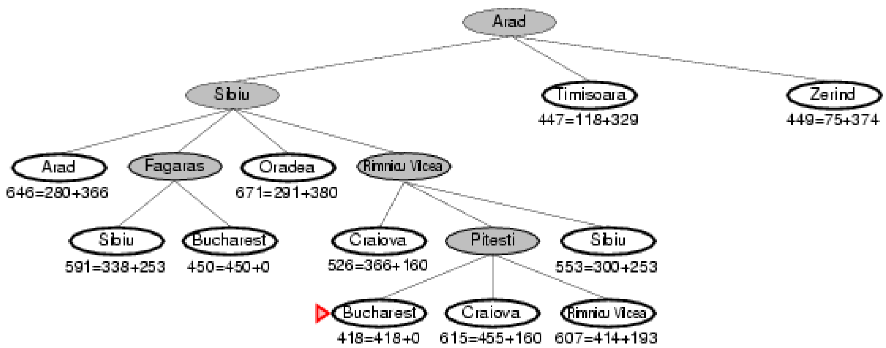
Sibiu$_{253}$ Timisoara$_{329}$ Zerind$_{374}$

Fagaras$_{176}$ T$_{329}$ RV$_{366}$ A$_{366}$ Z$_{374}$ O$_{380}$

Bucharest$_{0}$ S$_{253}$ T$_{329}$ RV$_{366}$ A$_{366}$ Z$_{374}$ O$_{380}$

Path found: (*Arad* $\xrightarrow{140}$ Sibiu $\xrightarrow{99}$ Fagaras $\xrightarrow{211}$ Bucharest)$_{450}$

Algorithm A/A*: Review



Nodes found/scheduled (opened): {A, S, T, Z, F, O, RV, S, B, C, P}

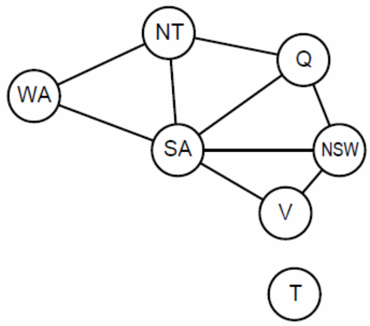
Nodes visited (closed): {A, S, F, RV, P, B}

Path found: (*Arad* $\xrightarrow{140}$ Sibiu $\xrightarrow{80}$ Rîmnicu Vîlcea $\xrightarrow{97}$ Pitesti $\xrightarrow{99}$ Bucharest)$_{416}$

Constraint Satisfaction Problems: Review

Binary CSP: each constraint relates at most two variables

Constraint graph: nodes are variables, arcs show constraints



General-purpose CSP algorithms use the graph structure
to speed up search. E.g., Tasmania is an independent subproblem!

Minimax & Alpha-Beta (α - β) Pruning: Review

What are α, β values here?

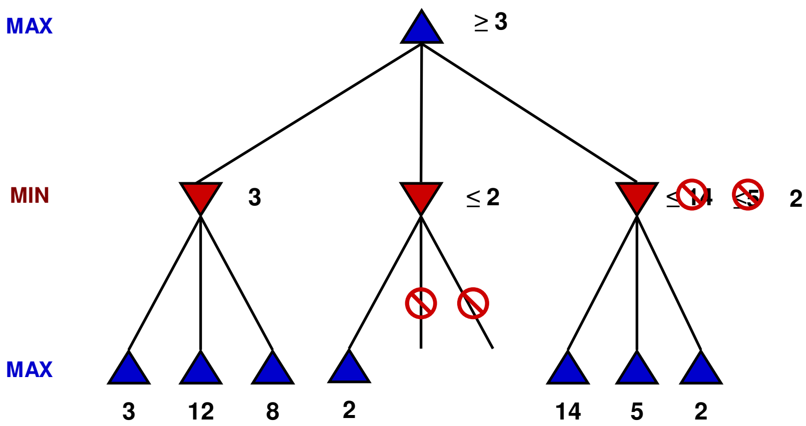


Figure 6.5 p. 168 R&N 2e

Inference: Review

$KB \vdash_i \alpha$ = sentence α can be derived from KB by procedure i

Consequences of KB are a haystack; α is a needle.
Entailment = needle in haystack; inference = finding it

Soundness: i is sound if
whenever $KB \vdash_i \alpha$, it is also true that $KB \models \alpha$

Completeness: i is complete if
whenever $KB \models \alpha$, it is also true that $KB \vdash_i \alpha$

Preview: we will define a logic (first-order logic) which is expressive enough to say almost anything of interest, and for which there exists a sound and complete inference procedure.

That is, the procedure will answer any question whose answer follows from what is known by the KB.

Logics in General: Review

Language	Ontological Commitment	Epistemological Commitment
Propositional logic	facts	true/false/unknown
First-order logic	facts, objects, relations	true/false/unknown
Temporal logic	facts, objects, relations, times	true/false/unknown
Probability theory	facts	degree of belief
Fuzzy logic	facts + degree of truth	known interval value

Ontological commitment – what entities, relationships, and facts exist in world and can be reasoned about

Epistemic commitment – what agents can know about the world

Clausal Form (CNF) Conversion: Review

- Implications Out (Replace with Disjunctive Clauses)
- Negations Inward (DeMorgan's Theorem)
- Standardize Variables Apart (Eliminate Duplicate Names)
- Existentials Out (Skolemize)
- Universals Made Implicit
- Distribute *And* Over *Or* (*i.e.*, Disjunctions Inward)
- Operators Made Implicit (Convert to List of Lists of Literals)
- Rename Variables (Independent Clauses)
- A Memonic for *Star Trek: The Next Generation* Fans

Captain Picard:

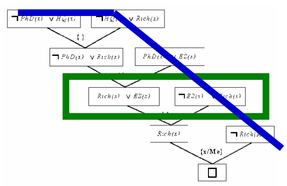
I'll Notify Spock's Eminent Underground Dissidents On Romulus

I'll Notify Sarek's Eminent Underground Descendant On Romulus

Adapted from: Nilsson and Genesereth (1987). *Logical Foundations of Artificial Intelligence.*
http://bit.ly/45Cmqq

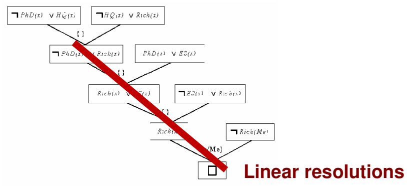
Resolution Strategies [1]: Review

- Unit Preference
  - Idea: Prefer inferences that produce shorter sentences
  - Compare: Occam's Razor
  - How? Prefer unit clause (*single-literal*) resolvents ($\alpha \vee \beta$ with $\neg \beta \vee \alpha$)
  - Reason: trying to produce a short sentence ($\perp \equiv$ True $\Rightarrow$ False)
- Input Resolution
  - Idea: "diagonal" proof (proof "list" instead of proof tree)
  - Every resolution combines some input sentence with some other sentence
  - Input sentence: *in original KB or query*



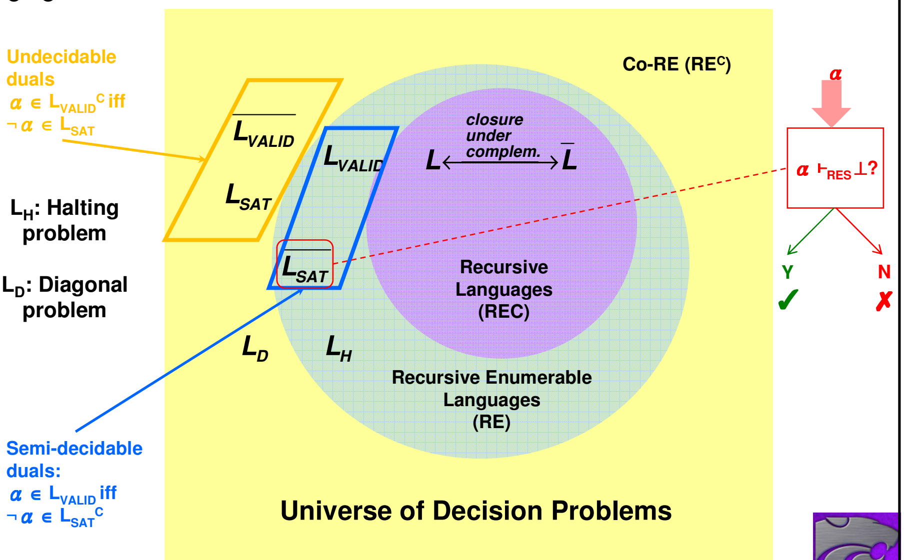
Resolution Strategies [2]: Review

- **Linear Resolution**
  - Generalization of input resolution
  - Include any *ancestor in proof tree* to be used



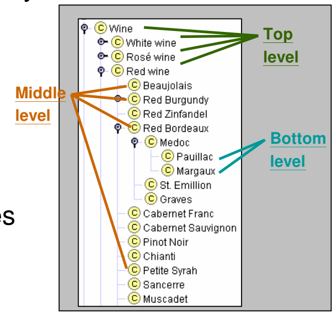
- **Set of Support (SoS)**
  - Idea: try to eliminate some potential resolutions
  - Prevention as opposed to cure
  - How?
    - ⇨ Maintain set SoS of resolution results
    - ⇨ Always take *one resolvent* from it
  - Caveat: need right choice for SoS to ensure completeness

Logic and Decision Problems: Review



Concepts/Classes: Review

- "Concept" and "Class" are used synonymously
- Class: concept in the domain
  - wines
  - wineries
  - red wines
- Collection of elements with similar properties
- Instances of classes
  - Particular glass of California wine



Adapted from slides © 2005 N. Noy & S. Tu
Stanford Center for Biomedical Informatics Research
http://bmir.stanford.edu

Slots/Attributes/Relations: Review

- Slots in class definition *C*
  - *Describe attributes of instances of C*
  - *Describe relationships to other instances*
  - *e.g., each wine will have color, sugar content, producer, etc.*
- Property constraints (facets): describe/limit possible values for slot

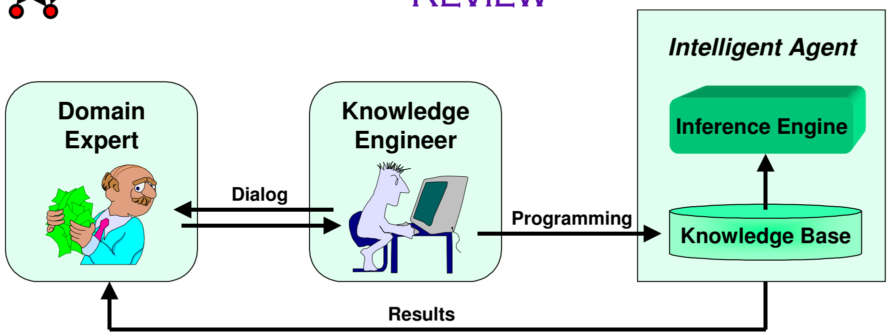
Template Slots

Name	Type	Cardinality	Other Facets
body	Symbol	single	allowed-values={FULL,MEDIUM,LIGHT}
color	Symbol	single	allowed-values={RED,ROSÉ,WHITE}
flavor	Symbol	single	allowed-values={DELICATE,MODERATE,STRONG}
grape	Instance	multiple	classes={Wine grape}
maker	Instance	single	classes={Winery}
name	String	single	
sugar	Symbol	single	allowed-values={DRY,SWEET,OFF-DRY}

Slots & facets for Concept/Class *Wine*

Adapted from slides © 2005 N. Noy & S. Tu
Stanford Center for Biomedical Informatics Research
http://bmir.stanford.edu

How Agents Are Built: Review



A knowledge engineer attempts to understand how a subject matter expert reasons and solves problems and then encodes the acquired expertise into the agent's knowledge base.

The expert analyzes the solutions generated by the agent (and often the knowledge base itself) to identify errors, and the knowledge engineer corrects the knowledge base.

CS 785 Knowledge Acquisition and Problem-Solving http://lalab.gmu.edu/cs785/

Elicitation Methodology: Review

(based primarily on Gammack, 1987)

1. **Concept elicitation: methods**
   (elicit concepts of domain, *i.e.* agreed-upon vocabulary)
2. **Structure elicitation: card-sort method**
   (elicit some structure for concepts)
3. **Structure representation**
   (formally represent structure in semantic network)
4. **Transformation of representation**
   (transform representation to be used for some desired purpose)

CS 785 Knowledge Acquisition and Problem-Solving http://lalab.gmu.edu/cs785/

Hierarchy and Taxonomy: Review

- Hierarchy or taxonomy is a natural way to view the world
  - It is used in frames (IS-A relation) and in DL
- importance of *abstraction* in remembering and reasoning
  - groups of things share properties in the world
  - we do not have to repeat representations

Example:

- Saying "elephants are mammals" is sufficient to know a lot about them

Inheritance is the result of reasoning over paths in a hierarchy

- "does *a* inherit from *b*?" is the same as "is *b* in the transitive closure of :**IS-A** (or subsumption) from *a*?"

Inheritance: Review

- IS relations:
- Clyde is an Elephant, Elephant is Gray

Grey

↓

Elephant

↓

Clyde

- Reasoning with paths and conclusions they represent:
  - Transitive relations
- Transitive closure:
- Clyde is Gray, Elephant is Gray, Clyde is Elephant

Actions, Situations, Time & Events: Review

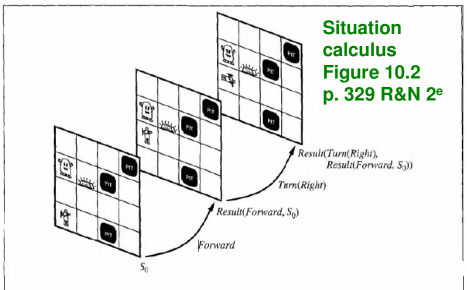


Figure 10.2 In situation calculus, each situation (except S_0) is the result of an action.

Situation calculus
Figure 10.2
p. 329 R&N 2e

- **Axioms: Truth of Predicate *P***
  - **Fully specify situations where *P* true**
  - **∴ biconditional (⇔, iff)**
- **Original Predicates**
  - **Describe state of world**
  - **Each augmented with situation argument *s***

$$\forall a, s \;\; Holding(Gold, Result(a, s)) \;\Leftrightarrow\; [(a = Grab \wedge AtGold(s)) \;\vee\; (Holding(Gold, s) \wedge a \neq Release)]$$

Successor-state axioms solve the representational frame problem

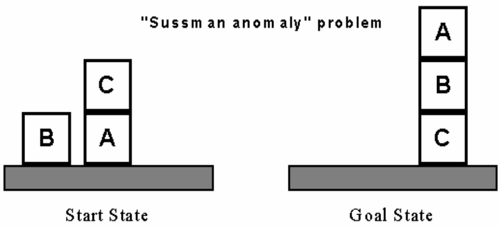
Each axiom is "about" a predicate (not an action per se):

P true afterwards ⇔ [an action made P true
∨ P true already and no action made P false]

Adapted from material © 2003 – 2004 S. Russell & P. Norvig.

Looking Ahead: Planning & Blocks World

"Sussman anomaly" problem



Clear(x) On(x,z) Clear(y)

PutOn(x,y)

~On(x,z) ~Clear(y) Clear(z) On(x,y)

Clear(x) On(x,z)

PutOnTable(x)

~On(x,z) Clear(z) On(x,Table)

+ several inequality constraints

Terminology

- Intelligent Agents
  - Chapter 1: Overview
  - Chapter 2: Definition of IAs
  - Types: Reflex, Reflex with State, Goal-Based, Preference-Based
- Search
  - Chapter 3: blind search
  - Chapter 4: informed search, heuristics, Best-First & variants
  - Chapter 5: constraints
  - Chapter 6: game tree search
- Section III: Knowledge Representation and Reasoning
  - Chapter 7: propositional logic
  - Chapter 8: first-order logic
  - Chapter 9: inference in FOL (resolution)
  - Chapter 10: knowledge representation

Summary Points

- Section I: Intelligent Agents, Chapters 1 – 2
  - Chapter 1: Overview
  - Chapter 2: Definition of IAs
  - Types: Reflex, Reflex with State, Goal-Based, Preference-Based
- Section II: Search, Chapters 3 – 6
  - Chapter 3: blind search
  - Chapter 4: informed search, heuristics, Best-First & variants
  - Chapter 5: constraints
  - Chapter 6: game tree search
- Section III: Knowledge Representation and Reasoning
  - Chapter 7: propositional logic
  - Chapter 8: first-order logic
  - Chapter 9: inference in FOL (resolution)
  - Chapter 10: knowledge representation