



Lecture 17 of 42

Knowledge Representation and Ontologies Discussion: Situational Calculus

Wednesday, 03 October 2007

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/Fall-2007/CIS730>

Instructor home page: <http://www.cis.ksu.edu/~bhsu>

Reading for Next Class:

Sections 10.1 – 10.2, Russell & Norvig 2nd edition



Lecture Outline

- **Today's Reading: Sections 10.1 – 10.2, R&N 2e**
- **Friday's Reading: Section 10.3, Russell and Norvig**
- **Previously: Logical Representations and Theorem Proving**
 - * Propositional, predicate, and first-order logical languages
 - * Proof procedures: forward and backward chaining, resolution refutation
- **Today: Knowledge Rep, Ontologies, Situational Calculus**
- **Friday**
 - * Temporal logic
 - * Semantic networks
 - * Description Logics
- **Midterm Exam: 16 Oct 2006**
 - * Remote students: have exam agreement faxed to DCE
 - * Exam will be faxed to proctors Wednesday or Friday





Midterm Review – IAs, Search: Unclear Points?

- **Artificial Intelligence (AI)**
 - * Operational definition: study / development of systems capable of “thought processes” (reasoning, learning, problem solving)
 - * Constructive definition: expressed in artifacts (design and implementation)
- **Intelligent Agent Framework**
 - * Reactivity vs. state
 - * From goals to preferences (utilities)
- **Methodologies and Applications**
 - * Search: game-playing systems, problem solvers
 - * Planning, design, scheduling systems
 - * Control and optimization systems
 - * Machine learning: hypothesis space search (for pattern recognition, data mining)
- **Search**
 - * Problem formulation: state space (initial / operator / goal test / cost), graph
 - * State space search approaches
 - ⇒ Blind (uninformed) – DFS, BFS, B&B
 - ⇒ Heuristic (informed) – Greedy, Beam, A/A*; Hill-Climbing, SA



Midterm Review – KR, Logic, Proof Theory: Unclear Points?

- **Logical Frameworks**
 - * Knowledge Bases (KB)
 - * Logic in general: representation languages, syntax, semantics
 - * Propositional logic
 - * First-order logic (FOL, FOPC)
 - * Model theory, domain theory: possible worlds semantics, entailment
- **Normal Forms**
 - * Conjunctive Normal Form (CNF)
 - * Disjunctive Normal Form (DNF)
 - * Horn Form
- **Proof Theory and Inference Systems**
 - * Sequent calculi: rules of proof theory
 - * Derivability or provability
 - * Properties
 - ⇒ Knowledge bases, WFFs: consistency, satisfiability, validity, entailment





Midterm Review – Game Trees: Unclear Points?

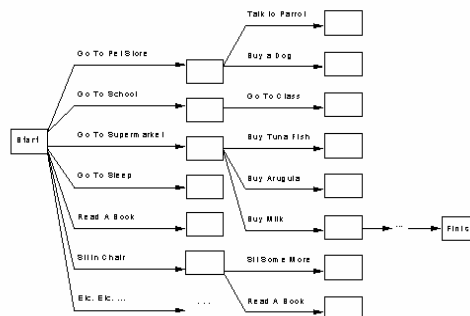
- Games as Search Problems
 - * Frameworks
 - * Concepts: utility, reinforcements, game trees
 - * Static evaluation under resource limitations
- Family of Algorithms for Game Trees: Minimax
 - * Static evaluation algorithm
 - ⇒ To arbitrary ply
 - ⇒ To fixed ply
 - ⇒ Sophistications: iterative deepening, alpha-beta pruning
 - * Credit propagation
 - ⇒ Intuitive concept
 - ⇒ Basis for simple (delta-rule) learning algorithms
- State of The Field
- Uncertainty in Games: Expectiminimax and Other Algorithms



Search versus Planning [1]

Consider the task *get milk, bananas, and a cordless drill*

Standard search algorithms seem to fail miserably:



After-the-fact heuristic/goal test inadequate

Adapted from slides by S. Russell, UC Berkeley





Planning in Situation Calculus

$PlanResult(p, s)$ is the situation resulting from executing p in s

$$PlanResult([], s) = s$$

$$PlanResult([a|p], s) = PlanResult(p, Result(a, s))$$

Initial state $At(Home, S_0) \wedge \neg Have(Milk, S_0) \wedge \dots$

Actions as Successor State axioms

$$Have(Milk, Result(a, s)) \Leftrightarrow$$

$$[(a = Buy(Milk) \wedge At(Supermarket, s)) \vee (Have(Milk, s) \wedge a \neq \dots)]$$

Query

$$s = PlanResult(p, S_0) \wedge At(Home, s) \wedge Have(Milk, s) \wedge \dots$$

Solution

$$p = [Go(Supermarket), Buy(Milk), Buy(Bananas), Go(HWS), \dots]$$

Principal difficulty: unconstrained branching, hard to apply heuristics

Adapted from slides by S. Russell, UC Berkeley



STRIPS Operators

Tidily arranged actions descriptions, restricted language

ACTION: $Buy(x)$

PRECONDITION: $At(p), Sells(p, x)$

EFFECT: $Have(x)$

[Note: this abstracts away many important details!]

Restricted language \Rightarrow efficient algorithm

Precondition: conjunction of positive literals

Effect: conjunction of literals

$At(p) \ Sells(p, x)$

Buy(x)

$Have(x)$

Adapted from slides by S. Russell, UC Berkeley



State Space versus Plan Space

Standard search: node = concrete world state

Planning search: node = partial plan

Defn: open condition is a precondition of a step not yet fulfilled

Operators on partial plans:

add a link from an existing action to an open condition

add a step to fulfill an open condition

order one step wrt another

Gradually move from incomplete/vague plans to complete, correct plans

Adapted from slides by S. Russell, UC Berkeley

CIS 530 / 730: Artificial Intelligence

Wednesday, 03 Oct 2007

Computing & Information Sciences
Kansas State University



Describing Actions [1]: Frame, Qualification, and Ramification Problems

"Effect" axiom—describe changes due to action

$\forall s \text{ AtGold}(s) \Rightarrow \text{Holding}(\text{Gold}, \text{Result}(\text{Grab}, s))$

"Frame" axiom—describe non-changes due to action

$\forall s \text{ HaveArrow}(s) \Rightarrow \text{HaveArrow}(\text{Result}(\text{Grab}, s))$

Frame problem: find an elegant way to handle non-change

(a) representation—avoid frame axioms

(b) inference—avoid repeated "copy-overs" to keep track of state

Qualification problem: true descriptions of real actions require endless caveats—what if gold is slippery or nailed down or ...

Ramification problem: real actions have many secondary consequences—what about the dust on the gold, wear and tear on gloves, ...

Adapted from slides by S. Russell, UC Berkeley

CIS 530 / 730: Artificial Intelligence

Wednesday, 03 Oct 2007

Computing & Information Sciences
Kansas State University





Describing Actions [2]: Successor State Axioms

Successor-state axioms solve the representational frame problem

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

$$P \text{ true afterwards} \Leftrightarrow [\text{an action made } P \text{ true} \\ \vee P \text{ true already and no action made } P \text{ false}]$$

For holding the gold:

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

Adapted from slides by S. Russell, UC Berkeley



Making Plans

Initial condition in KB:

$$\text{At}(\text{Agent}, [1, 1], S_0) \\ \text{At}(\text{Gold}, [1, 2], S_0)$$

Query: $\text{ASK}(KB, \exists s \text{ Holding}(\text{Gold}, s))$

i.e., in what situation will I be holding the gold?

Answer: $\{s / \text{Result}(\text{Grab}, \text{Result}(\text{Forward}, S_0))\}$

i.e., go forward and then grab the gold

This assumes that the agent is interested in plans starting at S_0 and that S_0 is the only situation described in the KB

Adapted from slides by S. Russell, UC Berkeley





Making Plans: A Better Way

Represent plans as action sequences $[a_1, a_2, \dots, a_n]$

$PlanResult(p, s)$ is the result of executing p in s

Then the query $ASK(KB, \exists p \text{ Holding}(Gold, PlanResult(p, S_0)))$
has the solution $\{p/[Forward, Grab]\}$

Definition of $PlanResult$ in terms of $Result$:

$$\forall s \text{ PlanResult}([], s) = s$$

$$\forall a, p, s \text{ PlanResult}([a|p], s) = \text{PlanResult}(p, \text{Result}(a, s))$$

Planning systems are special-purpose reasoners designed to do this type
of inference more efficiently than a general-purpose reasoner

Adapted from slides by S. Russell, UC Berkeley



First-Order Logic: Summary

First-order logic:

- objects and relations are semantic primitives
- syntax: constants, functions, predicates, equality, quantifiers

Increased expressive power: sufficient to define wumpus world

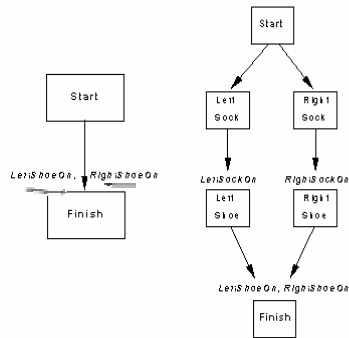
Situation calculus:

- conventions for describing actions and change in FOL
- can formulate planning as inference on a situation calculus KB

Adapted from slides by S. Russell, UC Berkeley



Partially-Ordered Plans



A plan is complete iff every precondition is achieved

A precondition is achieved iff it is the effect of an earlier step and no possibly intervening step undoes it

Adapted from slides by S. Russell, UC Berkeley



POP Algorithm [1]: Sketch

function POP(*initial, goal, operators*) **returns** *plan*

plan ← MAKE-MINIMAL-PLAN(*initial, goal*)

loop do

if SOLUTION?(*plan*) **then return** *plan*

S_{need}, c ← SELECT-SUBGOAL(*plan*)

 CHOOSE-OPERATOR(*plan, operators, S_{need}, c*)

 RESOLVE-THREATS(*plan*)

end

function SELECT-SUBGOAL(*plan*) **returns** S_{need}, c

 pick a plan step S_{need} from STEPS(*plan*)

 with a precondition *c* that has not been achieved

return S_{need}, c

Adapted from slides by S. Russell, UC Berkeley



POP Algorithm [2]: Subroutines and Properties

```

procedure CHOOSE-OPERATOR(plan, operators, Sneed, c)
  choose a step Sadd from operators or STEPS(plan) that has c as an effect
  if there is no such step then fail
  add the causal link  $S_{add} \xrightarrow{c} S_{need}$  to LINKS(plan)
  add the ordering constraint  $S_{add} \prec S_{need}$  to ORDERINGS(plan)
  if Sadd is a newly added step from operators then
    add Sadd to STEPS(plan)
    add  $Start \prec S_{add} \prec Finish$  to ORDERINGS(plan)
  
```

```

procedure RESOLVE-THREATS(plan)
  for each Sthreat that threatens a link  $S_i \xrightarrow{c} S_j$  in LINKS(plan) do
    choose either
      Demotion: Add  $S_{threat} \prec S_i$  to ORDERINGS(plan)
      Promotion: Add  $S_j \prec S_{threat}$  to ORDERINGS(plan)
    if not CONSISTENT(plan) then fail
  end
  
```

POP is sound, complete, and systematic (no repetition)

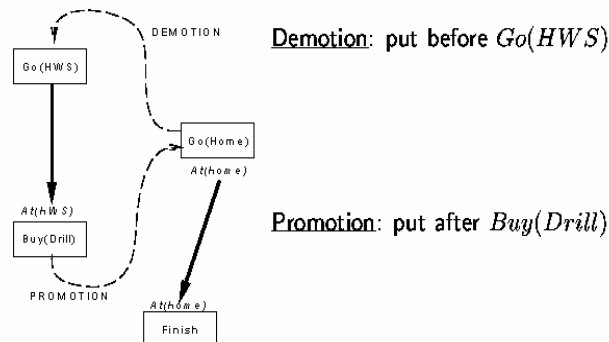
Extensions for disjunction, universals, negation, conditionals

Adapted from slides by S. Russell, UC Berkeley



Clobbering and Promotion / Demotion

A clobberer is a potentially intervening step that destroys the condition achieved by a causal link. E.g., *Go(Home)* clobbers *At(HWS)*:



Adapted from slides by S. Russell, UC Berkeley





Summary Points

- **Previously: Logical Representations and Theorem Proving**
 - * Propositional, predicate, and first-order logical languages
 - * Proof procedures: forward and backward chaining, resolution refutation
- **Today: Introduction to Classical Planning**
 - * Search vs. planning
 - * STRIPS axioms
 - ⇒ Operator representation
 - ⇒ Components: preconditions, postconditions (ADD, DELETE lists)
- **Next Monday: More Classical Planning**
 - * Partial-order planning (NOAH, etc.)
 - * Limitations



Terminology

- **Classical Planning**
 - * Planning versus search
 - * Problematic approaches to planning
 - ⇒ Forward chaining
 - ⇒ Situation calculus
 - * Representation
 - ⇒ Initial state
 - ⇒ Goal state / test
 - ⇒ Operators
- **Efficient Representations**
 - * STRIPS axioms
 - ⇒ Components: preconditions, postconditions (ADD, DELETE lists)
 - ⇒ Clobbering / threatening
 - * Reactive plans and policies
 - * Markov decision processes

Adapted from slides by S. Russell, UC Berkeley

