CS 440: Introduction to Artificial Intelligence

Lecture 7

Matthew Stone

September 23, 2015

Recap

Reasoning and architecture—How does the complexity of decision making affect behavior?

- What does an agent have to remember about the past?
- What does an agent have to consider about the future?
- How do you factor answers into code?

Reflex Agents

Simplest possible design: no memory or planning

- A list of condition–action statements
  - Each tests whether the condition is true
  - Then calls for the action appropriate to that condition
  - Such statements are often called RULES

Finite State Agents

Next simplest design: Robot has a variable *s* for state

- *s* takes on one of a small number of discrete values
- logic rules test state
- rules both: output intentions and update state

Rule template:

```
if (EXPRESSION(Percepts, S)) {
    S := New State
    Todo := New Intention
}
```

Limits of Reaction

- Reaction fails exactly when:
  Agent has same percepts
  but needs to make different decisions.
- Interesting connection between reasoning and capability:
  Need state to deal with ambiguity and uncertainty.

Combining Behaviors

Subsumption

- All behaviors get percepts, can propose actions
- Higher-level behaviors modulate actions proposed by lower-level ones

Weights

- All behaviors get percepts, can propose actions
- Actions are continuous and can be combined together

Flocking

Group behavior based on weighting actions

- separation
- alignment
- centering
- obstacle avoidance, etc.

Demo

- What does each heuristic do on its own?
- Why must you combine them?
- What kind of reasoning and representation?

Project Outline

Key part:

- Flocks

Work involves

- Basic agent
- Analysis in specific environments
- Optional, open-ended extensions

Skeleton code, examples, output, detailed description on sakai

Flocks—Interactions among agents

Basic agents

- Implement flocking forces: avoid enemies and obstacles, approach food, handle separation, alignment, centering
- See assignment text for details on calculating forces, weights, examples

Analysis

- Create scenarios that show specific features such as obstacles that keep flock from food flocks that split and remerge

Extensions—"Storytelling" with flocks

- boids with different speeds and abilities
- boids that scatter in response to predators
- separate flocks at war
- groups of predators that hunt as teams

Summary—Goals of assignment

Solidify material of last two weeks

- Understand agent architectures
  and the capabilities they give rise to
- Practice skills of analyzing agents
  acting in complex environments
- Get experience with behavioral simulation
  to visualize AI techniques and create virtual worlds

Summary: Agents

Architectures of intelligent behavior

- Programming systems to make their own decisions
- Perception, Deliberation and Action
- Probability and Utility
- Representation and Reasoning

Worked case studies—for future recitations

- Choosing actions by anticipating their effects
- Making uncertain decisions and learning from what happens

Agents and Knowledge

Strengths and weaknesses of scripted behavior

- Strength: Flexible routines that respond to current conditions
- Weakness: Remembering past experience
- Weakness: Anticipating future problems

Solution is to design agents with knowledge of the world

Fundamental Challenge

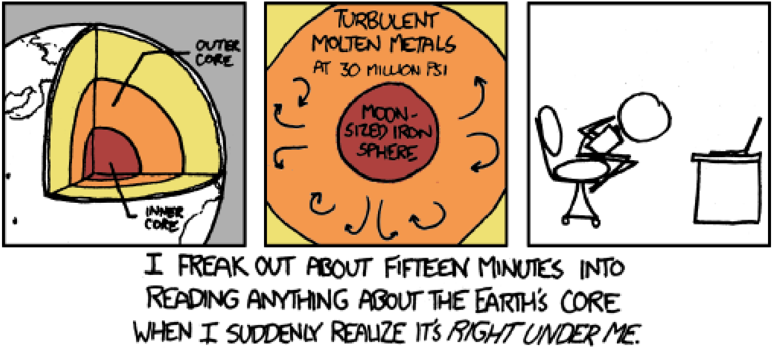
Applying knowledge in new situations

- Actual situations are *complex*.
  You usually haven't seen exactly the same thing before.
  Often: a novel mix of familiar features.
- Useful knowledge needs to be *generalizable*
  It describes large classes of situations
  in terms of underlying features.
- Consequence:
  New situations require a creative synthesis of existing knowledge

Example—Language understanding

"I freak out about fifteen minutes into reading anything about the Earth's core when I suddenly realize it's *right under me*."

Example—Language understanding



Example—Language understanding

Applying knowledge in new situations

- Situation: Creative language use.
  A sequence of words that nobody has ever used before.
- Knowledge: rules for understanding sentences.
  Forms and meanings of words, grammar of complex sentences
- Creative synthesis:
  Recognizing a new thought as the meaning of the sentence.

Search

Way to creatively synthesize pieces of knowledge

- Symbol structures represent current information
- Applying a piece of knowledge extends this information
- Can test whether current information solves your problem
- Systematically explore all the alternatives

Ingredients of search problems

- Initial state
- Possible actions in each state
- Transition model:
  Takes state and action and gives new state
- Goal test
  Describes whether state is what you want
- Path cost
  Says how easy or hard action sequence is

General Case

State space is a tree

- Each node has a set of children
  obtained by considering different actions
- Each action sequence represented as a new state

Sample Search Problem

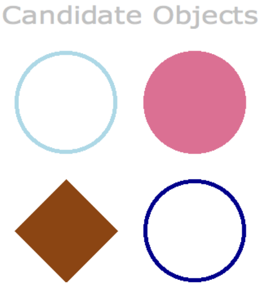
Generating English utterances

- intial state: empty string
- action: add a word
- goal: get your idea across clearly and correctly

Demo

Collaborative Reference

Picking out objects in the world for your interlocutor



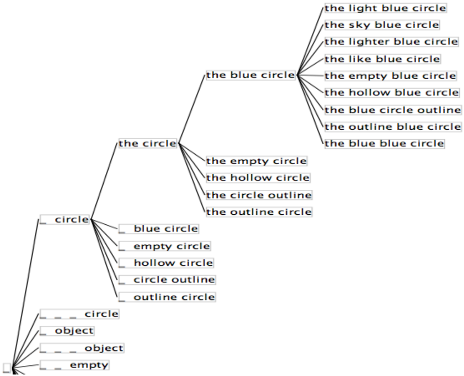
Agent: the light blue circle

Both interlocutors follow up to make sure they understand

- Inspired by the work of Stanford psychologist Herb Clark

Searching for an utterance

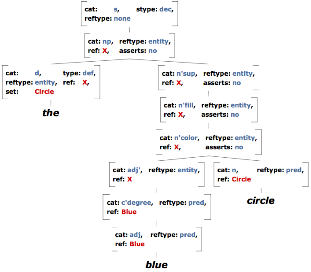
Search space arises by adding one word at a time to description



Representation

Description represented as syntactic tree

- shows dependency relationships (complements, modifiers)
- shows gaps and places to modify
- features and values handle agreement



Representation

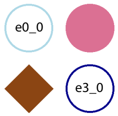
State keeps track of progress towards overall goal

- What syntactic information needs to be filled in?
- What does the description mean in context?

Here: nothing needs to be filled in but two referents are possible.

2 interpretations:
{Y←t1, PossVarVal←inTargetDomain, A←inFocus, Set←setPrag, Equals←equal, X←e0_0, Circle←circleFigureObject, M←addcr, Blue←lightblueFigureObject},
{Y←t1, PossVarVal←inTargetDomain, A←inFocus, Set←setPrag, Equals←equal, X←e3_0, Circle←circleFigureObject, M←addcr, Blue←darkblueFigureObject}

Candidate Objects

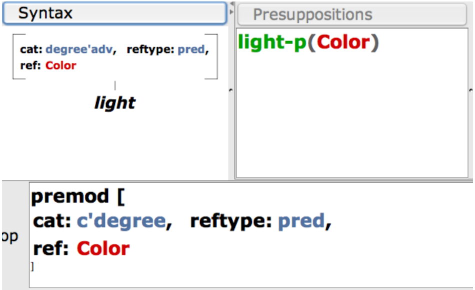


(key to symbols)

Representation

Words are associated with tree fragments that merge in

- Tree-adjoining Grammar: TAG by Aravind Joshi
- Items paired with meanings



This item precedes a color word and says the color is "light".

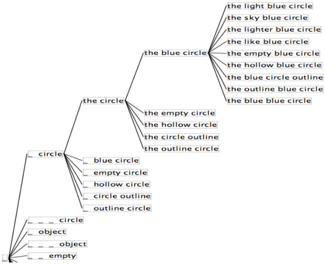
Summary— Why it works

Can make local decisions

- No “dead ends” – you can always say more
- Partial meaning is a good guide to progress

Explore only tiny part of search space

- Look at all actions in each state
- Pick the best and never look back
- $O(kd)$



Summary— Why it helps

- Program factored into knowledge and goals
- Knowledge
  - Grammatical structures
  - Words and their meanings
  - Shared context
- Goals
  - Complete sentence
  - Right meaning
  - Unambiguous
  - Natural

Summary— Why it helps

- Program factored into knowledge and goals
- Get flexible, general decision making without rules
- Can learn knowledge from other data sources
- Can adapt goals to new objectives
- Same code works