Planning: Representation

CPSC 322 Lecture 14

February 7, 2007
Textbook §11.1

Lecture Overview

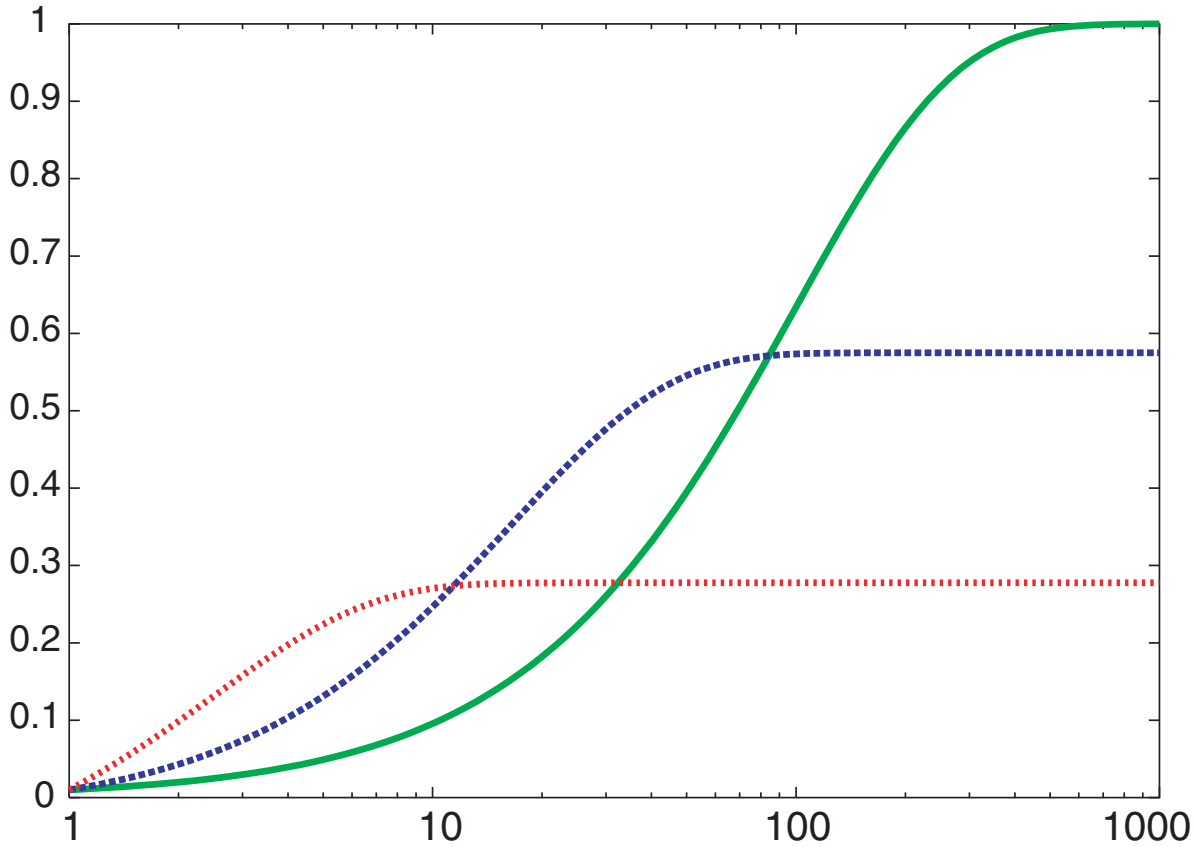
1. Recap
2. More SLS Variants
3. Planning
4. State-Based Representation
5. Feature-Based Representation

Stochastic Local Search

- Randomness can help in local search to improve hill climbing.
- As well as uphill steps we can allow for:
  - Random steps: move to a random neighbor.
  - Random restart: reassign random values to all variables.
- We can also mix together these approaches.

Runtime Distribution

- To compare SLS algorithms, use runtime distributions



SLS Variants

- Greedy Descent with Min-Conflict Heuristic
  - randomly select a variable that participates in a violated constraint; set it to the value that minimizes num violated constraints
- Simulated Annealing
  - randomly choose a variable/value combination
  - always keep it if it's an improvement; if it's not an improvement, keep it anyway with a probability that depends on the "temperature"
  - gradually "cool down"
- Tabu lists
  - keep a list of the last k nodes visited
  - don't revisit nodes on this list

Lecture Overview

1. Recap
2. More SLS Variants
3. Planning
4. State-Based Representation
5. Feature-Based Representation

Parallel Search

- Idea: maintain k nodes instead of one.
- At every stage, update each node.
- Whenever one node is a solution, report it.
- Like k restarts, but uses k times the minimum number of steps.
- There's not really any reason to use this method, but it provides a framework for talking about what follows...

Beam Search

- Like parallel search, with k nodes, but you choose the k best out of all of the neighbors.
- When $k = 1$, it is hill climbing.
- When $k = \infty$, it is breadth-first search.
- The value of k lets us limit space and parallelism.

Stochastic Beam Search

- Like beam search, but you probabilistically choose the k nodes at the next generation.
- The probability that a neighbor is chosen is proportional to the value of the scoring function.
  - This maintains diversity amongst the nodes.
  - The heuristic value reflects the fitness of the node.
  - Biological metaphor: like asexual reproduction, as each node gives its mutations and the fittest ones survive.

Genetic Algorithms

- Like stochastic beam search, but pairs of nodes are combined to create the offspring:
- For each generation:
  - Randomly choose pairs of nodes, with the best-scoring nodes being more likely to be chosen.
  - For each pair, perform a cross-over: form two offspring each taking different parts of their parents
  - Mutate some values
- Report best node found.

Crossover

- Given two nodes:

$$X_1 = a_1, X_2 = a_2, \ldots, X_m = a_m$$

$$X_1 = b_1, X_2 = b_2, \ldots, X_m = b_m$$

- Select i at random.
- Form two offspring:

$$X_1 = a_1, \ldots, X_i = a_i, X_{i+1} = b_{i+1}, \ldots, X_m = b_m$$

$$X_1 = b_1, \ldots, X_i = b_i, X_{i+1} = a_{i+1}, \ldots, X_m = a_m$$

- Note that this depends on an ordering of the variables.
- Many variations are possible.

Lecture Overview

1 Recap

2 More SLS Variants

3 Planning

4 State-Based Representation

5 Feature-Based Representation

Planning

- To a large extent, CSPs and search have been focused on atemporal problems.
  - Before, when solving a CSP, we wanted to find a single state of the world that satisfied all of our constraints
- How should an agent reason and act when it will take a sequence of actions to achieve a goal?
  - Now we want to find a *sequence* of states of the world that get us from our initial state to the goal state

Planning: Assumptions

- We make the following assumptions about the world:
  - the world is deterministic
  - the agent knows what state it is in and understands the consequences of its actions
  - the agent is the only thing that changes the state of the world
  - time passes in discrete steps
  - goals are predicates of states ("goal tests")

Goals

- Achievement goals: conditions on the final state we want to achieve
- Maintenance goals: conditions that we want to have remain true throughout the plan
- Transient goals: conditions that we want to achieve before the final achievement goal, but which do not need to remain true
- Resource goals: functions we want to minimize in the pursuit of our achievement goals (e.g., energy expended; distance traveled.)

Lecture Overview

1 Recap

2 More SLS Variants

3 Planning

4 State-Based Representation

5 Feature-Based Representation

State-Based Representation of a Planning Domain

- The domain is characterized by states, actions and goals
  - note: a given action may not be possible in all states
- Key issue: representing the way we transition from one state to another by taking actions
- A state-based representation has no structure
  - there's no sense in which we can say that states a and b are more similar than states a and z
- Thus, we can't do better than a tabular representation:

Starting state	Action	Resulting state
⋮	⋮	⋮

Problems with the Tabular Representation

Problems:

- Usually too many states for a tabular representation to be feasible
- Small changes to the model can mean big changes for the representation
  - e.g., if we added another feature, all the states would change
- There may be structure and regularity to the actions, and to the states themselves. If there is, there's no way to capture it with this representation.

Lecture Overview

1 Recap

2 More SLS Variants

3 Planning

4 State-Based Representation

5 Feature-Based Representation

Feature-Based Representation

- Features helped us to represent CSPs more compactly than states could. Can they help us here?
- Note that our problem is trickier than the original CSP problem:
  - we used to be looking for a single variable assignment, out of the space of all variable assignments, that satisfies our constraints
- Now we are looking for a sequence of variable assignments that
  - begins at the initial state
  - proceeds from one state to another by taking valid actions
  - ends up at a goal
- This means that instead of having one variable for every feature, we must instead have one variable for every feature at each time step, indicating the value taken by that feature at that time step!