

CS540 Introduction to Artificial Intelligence

Lecture 17

Young Wu

Based on lecture slides by Jerry Zhu, Yingyu Liang, and Charles Dyer

July 10, 2020

Hill Climbing
●oooooooooo

Simulated Annealing
ooooo

Genetic Algorithm
oooooooooooo

Traveling Salesperson Example

Motivation

Search vs. Local Search

Motivation

- Some problems do not have an initial state and a goal state.
- Every state is a solution. Some states are better than others, defined by a cost function (sometimes called score function in this setting), $f(s)$.
- The search strategy will go from state to state, but the path between states is not important.
- There are too many states to enumerate, so standard search through the state space methods are too expensive.

Local Search

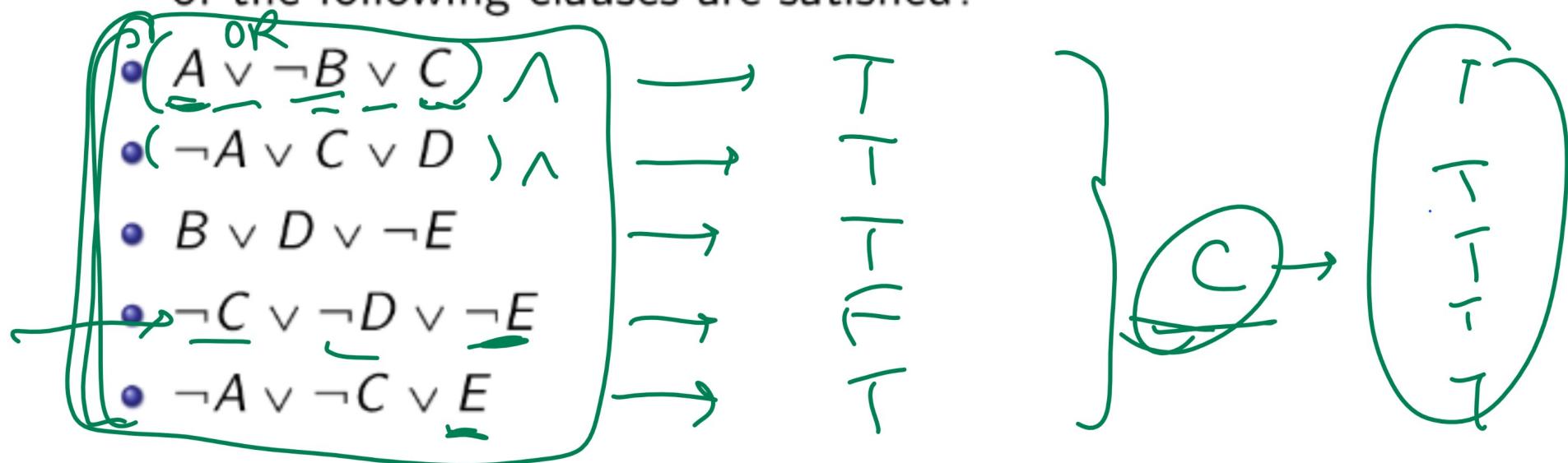
Motivation

- Local search is about searching through a state space by iteratively improving the cost to find an optimal or near-optimal state.
- The successor states are called the neighbors (sometimes move set).
- The assumption is that similar (nearby) solutions have similar costs.

Boolean Satisfiability Example 1

Quiz

- Assume all variables A, B, C, D, E are set to True. How many of the following clauses are satisfied?



Boolean Satisfiability Example 2

Quiz

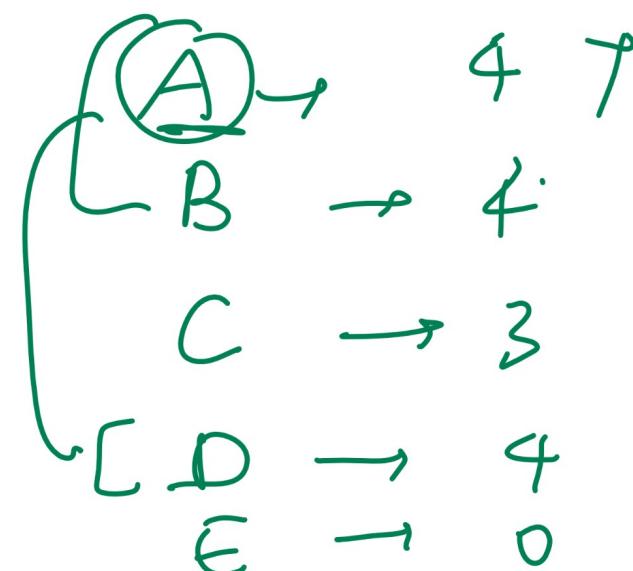
- Assume all variables A, B, C, D, E are set to True. Which one of the variables should be changed to False to maximize the number of clauses satisfied?
 $C \leftrightarrow D$
- $A \vee \neg B \vee C$
- $\neg A \vee C \vee D$
- $B \vee D \vee \neg E$
- $\neg C \vee \neg D \vee \neg E$
- $\neg A \vee \neg C \vee E$

Boolean Satisfiability Example 3

Quiz

- Assume all variables A, B, C, D, E are set to True. Which one of the variables should be changed to False to maximize the number of clauses satisfied?

• $\neg A \vee \neg B \vee \neg C$	$\rightarrow F$
• $\neg A \vee \neg B \vee \neg D$	$\rightarrow F$
• $\neg A \vee \neg C \vee \neg D$	$\rightarrow F$
• $\neg A \vee \neg B \vee \neg D$	$\rightarrow F$
• $\neg B \vee \neg C \vee \neg D$	$\rightarrow F$



Hill Climbing (Valley Finding)

Description

- Start at a random state.
- Move to the best neighbor state (one of the successors).
- Stop when all neighbors are worse than the current state.
- The idea is similar to gradient descent.

Hill Climbing Algorithm

- Input: state space S and cost function f .
- Output: $s^* \in S$ that minimizes $f(s)$.
- Start at a random state s_0 .
- At iteration t , find the neighbor that minimizes f .

$$s_{t+1} = \arg \min_{s \in s'(s_t)} f(s)$$

- Stop when none of the neighbors have a lower cost.

stop if $f(s_{t+1}) \leq f(s_t)$

Random Restarts

Discussion

- A simple modification is picking random initial states multiple times and finding the best among the local minima.

First Choice Hill Climbing

Discussion

- If there are too many neighbors, randomly generate neighbors until a better neighbor is found.
- This method is called first choice hill climbing.

Simulated Annealing

Description

- Each time, a random neighbor is generated.
- If the neighbor has a lower cost, move to the neighbor.
- If the neighbor has a higher cost, move to the neighbor with a small probability.
- Stop until bored.
- It is a version of Metropolis-Hastings Algorithm.

Acceptance Probability

Definition

- The probability of moving to a state with a higher cost should be small.
- ① Constant: $p = 0.1$
- ② Decreases with time: $p = \frac{1}{t}$
- ③ Decreases with time and as the energy difference increases:
$$p = \exp\left(-\frac{|f(s') - f(s)|}{\text{Temp } (t)}\right)$$
- The algorithm corresponding to the third idea is called simulated annealing. Temp should be a decreasing in time (iteration number).

Temperature

Definition

- Temp represents temperature which decreases over time. For example, the temperature can change arithmetically or geometrically.

$\text{Temp } (t + 1) = \max \{ \text{Temp } (t) - 1, 1 \}$, $\text{Temp } (0) = \text{large}$

$\text{Temp } (t + 1) = 0.9 \text{Temp } (t)$, $\text{Temp } (0) = \text{large}$

- High temperature: almost always accept any s' .
- Low temperature: first choice hill climbing.

Simulated Annealing

Algorithm

- Input: state space S , temperature function Temp , and cost function f .
- Output: $s^* \in S$ that minimizes $f(s)$.
- Start at a random state s_0 .
- At iteration t , generate a random neighbor s' , and update the state according to the following rule.

$$s_{t+1} = \begin{cases} s' & \text{if } f(s') < f(s_t) \\ s' & \text{with probability } \exp\left(-\frac{|f(s') - f(s_t)|}{\text{Temp}(t)}\right) \\ s_t & \text{otherwise} \end{cases}$$

Simulated Annealing Performance

Discussion

- Use hill-climbing first.
- Neighborhood design is the most important.
- In theory, with infinitely slow cooling rate, SA finds global minimum with probability 1.

Genetic Algorithm

Description

- Start with a fixed population of initial states.
- Find the successors by:
 - ➊ Cross over.
 - ➋ Mutation.

Reproduction Probability

Definition

- Each state in the population has probability of reproduction proportional to the fitness. Fitness is the opposite of the cost: higher cost means lower fitness. Use F to denote the fitness function, for example, $F(s) = \frac{1}{f(s)}$ is a valid fitness function.

$$p_i = \frac{F(s_i)}{\sum_{j=1}^N F(s_j)}, i = 1, 2, \dots, N$$

- A pair of states are selected according to the reproduction probabilities (using CDF inversion).

Cross Over

Definition

- The states need to be encoded by strings.
- Cross over means swapping substrings.
- For example, the children of 10101 and 01010 could be the same as the parents or one of the following variations.

(11010, 00101) , (10010, 01101)

(10110, 01001) , (10100, 01011)

Mutation

Definition

- The states need to be encoded by strings.
- Mutation means randomly updating substrings. Each character is changed with small probability q , called the mutation rate.
- For example, the mutated state from 000 could stay the same or be one of the following.

one of 001, 010, 100, with probability $q(1 - q)^2$

one of 011, 101, 110, with probability $q^2(1 - q)$

and 111, with probability q^3

Cross Over, Modifications

Definition

- The previous cross over method is called 1 point cross over.
- It is also possible to divide the string into N parts. The method is called N point cross over.
- It is also possible to choose each character from one of the parents randomly. The method is called uniform cross over.

Mutation, Modifications

Definition

- For specific problems, there are ways other than flipping bits to mutate a state.
- 1 Two-swap: ABCDE to EBCDA
 - 2 Two-interchange: ABCDE to EDCBA

Fitness Example 1

Quiz

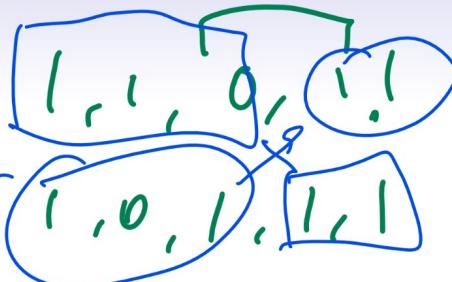
- Fall 1999 Final Q5
- Which ones (multiple) of the following states have the highest reproduction probability?

- The fitness function is $f(x) = 5x_1 + 3x_2x_3 - x_4 + 2x_5$.

- A: (1, 1, 0, 1, 1) → $5 \cdot 1 + 3 \cdot 1 \cdot 0 - 1 + 2 \cdot 1 = 6$ ✓
- B: (0, 1, 1, 0, 1) ✓
- C: (1, 1, 0, 0, 0) ✓
- D: (1, 0, 1, 1, 1) ✓
- E: (1, 0, 0, 0, 0) ✓

$$P(A) = \frac{6}{6+5+5+6+5} = P(D)$$

$$P(C) = P(E) = \frac{5}{25}$$

 x_3 and x_4

Fitness Example 2

child 1 → Quiz
(, 1, 0, 1, 1
(, 0, 1, 1)

- Which one of the following states have the highest reproduction probability? The fitness function is

$$f(x) = \min\{t \in \{1, 2, 3, 4, 5, 6\} : x_t = 1\} \text{ with } x_6 = 1.$$

- A: (0, 0, 1, 0, 0) → 3
- B: (0, 1, 0, 0, 1) → 2
- C: (0, 0, 1, 1, 0) → 3
- D: (0, 0, 0, 1, 0) → 4
- E: (0, 0, 0, 0, 0) → 6

Fitness Example 3

Quiz

- Which one of the following states have the highest reproduction probability? The fitness function is

$$f(x) = \max\{t \in \{0, 1, 2, 3, 4, 5\} : x_t = 1\} \text{ with } x_0 = 1.$$

- A: (0, 0, 1, 0, 0) $\rightarrow 3$
- B: (0, 1, 0, 0, 1) $\leftarrow 5$
- C: (0, 0, 1, 1, 0) $\rightarrow 4$
- D: (0, 0, 0, 1, 0) $\rightarrow 4$
- E: (0, 0, 0, 0, 0) $\rightarrow 0$

Genetic Algorithm, Part I

Algorithm

- Input: state space S represented by strings s and cost function f or fitness function F .
- Output: $s^* \in S$ that minimizes $f(s)$.
- Randomly generate N solutions as the initial population.

s_1, s_2, \dots, s_N

- Compute the reproduction probability.

$$p_i = \frac{F(s_i)}{\sum_{j=1}^N F(s_j)}, i = 1, 2, \dots, N$$

Genetic Algorithm, Part II

Algorithm

- Randomly pick two states according to p_i , say s_a, s_b .
Randomly select a cross over point c , swap the strings.

$$s'_a = s_a [0 \dots c) \, s_b [c \dots m)$$

$$s'_b = s_b [0 \dots c) \, s_a [c \dots m)$$

- Randomly mutate each position of each state s_i with a small probability (mutation rate).

$$s'_i [k] = \begin{cases} s_i [k] & \text{with probability } 1 - q \\ \text{random} & \text{with probability } q \end{cases}, k = 1, 2, \dots, m$$

- Repeat with population s' .

Variations

Discussion

- Parents can survive.
- Use ranking instead of $F(s)$ to compute reproduction probabilities.
- Cross over random bits instead of chunks.

Genetic Algorithm Performance

Discussion

- Use hill-climbing first.
- State design is the most important.
- In theory, cross over is much more efficient than mutation.