Heuristics
Local Search

CSE 473
University of Washington

Admissable Heuristics

- f(x) = g(x) + h(x)
- g: cost so far
- h: underestimate of remaining costs

Where do heuristics come from?

Relaxed Problems

- Derive admissible heuristic from **exact** cost of a solution to a **relaxed** version of problem
  For transportation planning, relax requirement that car has to stay on road → Euclidean dist
- Cost of optimal soln to relaxed problem ≤ cost of optimal soln for real problem

Simplifying Integrals

vertex = formula
goal = closed form formula without integrals
arcs = mathematical transformations

$$\int \qquad {}^{n+} \qquad n \top$$

heuristic = number of integrals still in formula

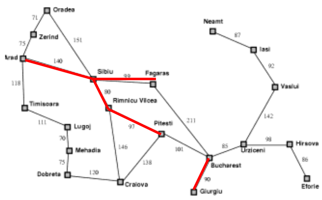
what is being relaxed?

Traveling Salesman Problem

- Problem Space

  States = partial path (not nec. connected)
  Operator = add an edge
  Start state = empty path
  Goal = complete path

- Heuristic?



What can be Relaxed?

Heuristics for eight puzzle

7	2	3
5	1	6
8	3	

→

1	2	3
4	5	6
7	8	

start **goal**

- What can we relax?

Importance of Heuristics

7	2	3
4	1	6
8	5	

- h1 = number of tiles in wrong place
- h2 = distances of tiles from correct loc

D	IDS	A*(h1)	A*(h2)
2	10	6	6
4	112	13	12
6	680	20	18
8	6384	39	25
10	47127	93	39
12	364404	227	73
14	3473941	539	113
18		3056	363
24		39135	1641

Need More Power!

Performance of Manhattan Distance Heuristic

8 Puzzle	< 1 second
15 Puzzle	1 minute
24 Puzzle	65000 years

Need even better heuristics!

Adapted from Richard Korf presentation

Pattern Databases

[Culberson & Schaeffer 1996]

- Pick any subset of tiles
  - E.g., 3, 7, 11, 12, 13, 14, 15
- Precompute a table
  - Optimal cost of solving just these tiles
  - For all possible configurations
    - 57 Million in this case
  - Use breadth first search back from goal state
    - State = position of just these tiles (& blank)

Adapted from Richard Korf presentation

Using a Pattern Database

- As each state is generated
  - Use position of chosen tiles as index into DB
  - Use lookup value as heuristic, h(n)

  Admissible?

Adapted from Richard Korf presentation

Combining Multiple Databases

- Can choose another set of tiles
  - Precompute multiple tables
- How combine table values?

- E.g. Optimal solutions to Rubik's cube
  - First found w/ IDA* using pattern DB heuristics
  - Multiple DBs were used (dif subsets of cubies)
  - Most problems solved optimally in 1 day
  - Compare with ***574,000 years*** for IDDFS

Adapted from Richard Korf presentation

Drawbacks of Standard Pattern DBs

- Since we can only take ***max***
  - Diminishing returns on additional DBs

- Would like to be able to ***add*** values

Adapted from Richard Korf presentation

Disjoint Pattern DBs

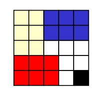
1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	

- Partition tiles into disjoint sets
  - For each set, precompute table
    - E.g. 8 tile DB has 519 million entries
    - And 7 tile DB has 58 million
- During search
  - Look up heuristic values for each set
  - ***Can add values without overestimating!***

  Manhattan distance is a special case of this idea where each set is a single tile

Performance

- 15 Puzzle: 2000x speedup *vs* Manhattan dist
  - IDA* with the two DBs shown previously solves 15 Puzzles optimally in 30 milliseconds
- 24 Puzzle: 12 million x speedup *vs* Manhattan
  - IDA* can solve random instances in 2 days.
  - Requires 4 DBs as shown
    - Each DB has 128 million entries
  - Without PDBs: 65000 years



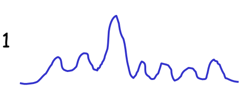
Local Search Algorithms

- In many optimization problems, the path to the goal is irrelevant; the goal state itself is the solution
-
- State space = set of "complete" configurations
- Find configuration satisfying constraints, e.g., n-queens
- In such cases, we can use local search algorithms that keep a single "current" state, try to improve it
-

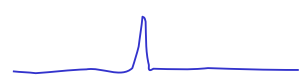
Hill Climbing

"Gradient ascent"

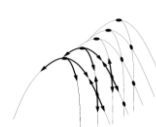
- Idea
  - Always choose best child; no backtracking
  - Beam search with |queue| = 1
- Problems?
  - Local maxima



  - Plateaus



  - Diagonal ridges

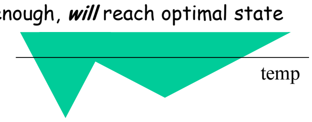


Stochastic Hill Climbing

- Randomly disobeying heuristic
- Random restarts

Simulated Annealing

- Objective: avoid local minima
- Technique:
  - For the most part use hill climbing
  - When no improvement possible
    - Choose random neighbor
    - Let a be the decrease in quality
    - Move to neighbor with probability $e^{-a/T}$
  - Reduce "temperature" (T) over time
- Pros & cons
  - Optimal?
  - If T decreased slowly enough, ***will*** reach optimal state
- Widely used
- See also
  - WalkSAT



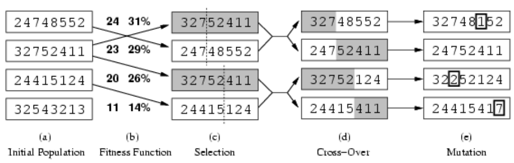
Local Beam Search

- Idea
  - Best first but only keep N best items on priority queue
- Evaluation
  - Complete?
  - Time Complexity?
  - Space Complexity?

Genetic Algorithms

- Start with random population
  - Representation serialized
  - States are ranked with "fitness function"
- Produce new generation
  - Select random pair(s):
    - probability ~ fitness
  - Randomly choose "crossover point"
    - Offspring mix halves
  - Randomly mutate bits

Genetic algorithms



- Fitness function: number of non-attacking pairs of queens (min = 0, max = 8 × 7/2 = 28)
-
- 24/(24+23+20+11) = 31%
-
- 23/(24+23+20+11) = 29% etc

Genetic algorithms