Lecture01b: Search

CS540 1/18/18

Announcements

Class Piazza site going up
- Accessible through class web site
- Mechanisms for discussing reading assignments (mandatory)

Off-campus students: send email to cs540@cs.colostate.edu
- Dejan will respond

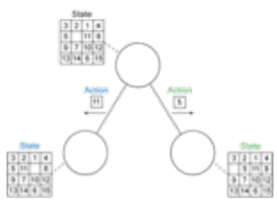
First reading assignment: R. Qu, et. al. *A Survey of Methodologies and Automated System Development for Examination Timetabling,* Journal of Scheduling, 12(1):55-89.
- Comment on Piazza by Thursday
- Warning: long paper

Search Topics

Applications
- Combinatorial Optimization
- Scheduling
- Planning

Algorithms
- Stochastic Local Search
- … and others

Analyses
- Phase Transitions
- Structural Analysis
- Statistical Models



https://www.ibm.com/developerworks/library/j-ai-search/

Combinatorial Problems

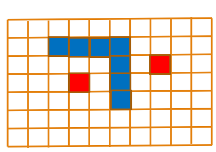
Applications
- Traveling Salesman Problem (TSP)
- Satisfiability (SAT)
- Planning
- Scheduling
- Packet Routing
- …

Informal Definition

Find an ordering, grouping or assignment of a discrete, finite set of objects that satisfies given conditions (from Hoos & Stutzle)

Example Search Problems

1. Satisfiability (SAT)
   - In CNF
   - Example: (A v ~B) ^ (B v ~C) ^ (C v ~A) ^ (~A v ~B v ~C)
   - Simple examples, clean formulation
2. Blocks World Path Planning
   - Harder examples, but closer to our project domain
   - Not as simple to formalize
   - Example: create formation with red blocks next to each other
   - Formalize:
     - No two blocks in one square
     - Blocks move one square at a time



Decision vs Optimization Problems

Decision
- Solution satisfies logical condition (e.g. SAT)
- For a given problem instance
  - Does a feasible solution exist? (decision variant)
  - What is a feasible solution? (search variant)

Optimization
- Add an objects function *f* that quantifies the quality of the solution
- "Best" can minimize or maximize *f*
- For a given problem instance
  - Does a solution exist with a given value of *f* (or better)? (decision variant)
  - What is a solution with optimal *f*? (search variant)
  - What is the optimal value of *f*? (evaluation variant)

Decision vs Optimization: Examples

1. Satisfiability
   - Decision:
     - Is there an assignment of truth values that satisfies the CNF equation?
   - Optimization:
     - Let *f* be the number of satisfied clauses
     - Find an assignment that maximizes *f*
2. Blocks World Path Planning
   - Decision
     - Is there a path that puts the red blocks together?
   - Optimization
     - Let *f* be the number of steps in a path
     - Find a shortest path that puts the red blocks together

Approaches to Search

Dimension #1: Solution Formation

- Construction
  - Start with NULL solution
  - Add to it
- Perturbation
  - Start with complete (but possibly incorrect) solution
  - Modify it

Construction vs Perturbation Example

- SAT: (A v ~B) ^ (B v ~C) ^ (C v ~A) ^ (~A v ~B v ~C)
- Constructive:
  - {}, {A}, {A, B}, {A,B,C} (failure)
  - {}, {~A}, {~A,~B}, {~A, ~B, ~C} (success)
- Perturbation
  - {A, B, C} : f = 3, {~A,B,C} : f = 3, {~A, ~B, C} : f = 3, {~A, ~B, ~C} : f =4

Approaches to Search (II)

Dimension #2: Search Space Traversal

- Systematic
  - Guarantee completeness
  - Example: breadth-first, depth first, etc.
- Local Search
  - Traversal based on information in current state
  - Example: Greedy Search

Definitions

Completeness

- Given enough time, all feasible points in search space are explored

Any-time Property

- A solution is always available
- More time may produce better solution

Stochasticity

- Randomness is part of the search strategy
- Well, often pseudo-random…

When to use which approach?

Rules of Thumb:

- Systematic Search
  - When solution quality guarantees demand it
    - … and sufficient time is available
  - For decision problems
    - E.g. satisfiability
- Local Search
  - When strong heuristics are available
  - When time is tight
  - When the anytime property is important
  - When the search space is really big…

http://idioms.languagesystems.edu/2013/05/rule-of-thumb.html

How to Measure Success

How do you know if your search algorithm is good?

Classic ``strawman" algorithms:

- Greedy
  - Start with null solution
  - Maximize heuristic *f* at every choice to generate solution
- Iterative Sampling (Langley 92)
  - Start with null solution
  - At every step, randomly add to growing solution
  - If dead-end, restart
  - *Claim: this is a complete algorithm. Why?*

CS540 Advanced Search Algorithms

Systematic Search Algorithms
- Randomized choices

Local Search Algorithms
- Randomized iterative improvement
- Simulated annealing
- Tabu search
- Iterated Tabu search
- Dynamic local search
- Variable neighborhood search

Hybrid Search Algorithms
- GRASP
- Squeky Wheel Optimization

Randomized Systematic Search

Randomization + restarts (Gomes, Selman & Kautz, 1998)

Start with a complete constructive search algorithm and a heuristic measure *f*
- Either constructive or perturbation

At choice points
- Select randomly from heuristically equivalent options
- Those with H% of the best (according to *f*)
- *Cutoff* parameter limits the number of backtracks
- Add bookkeeping to preclude repetition

Observations about RSS

Distribution of search costs across trials "heavy tailed"
- Many fast trials
- A few really slow ones
- Hypothesis: bad trials for poor early choices
- Therefor, restarts are importante

Benefits
- Robust
- Easy to parallelize

Costs
- Hard to analyze & ensure completeness
- More parameters