1 handout: slides

You think you know when you can learn, are more sure when you can write, even more when you can teach, but certain when you can program.
Optimization

Types of Problems
- LPs
- Beyond LPs

Types of Optimization Problems

- Discrete
  - finite (often small) set of values per choice
  - planning, CSPs, combinatorial optimization
  - constraints can be implicit or explicit

- Continuous
  - real values
  - constraints can be linear, quadratic, ..., non-linear
  - objective can be linear, quadratic, ..., non-linear

- Mixed discrete / continuous
  - ‘hybrid systems’
real variables, linear constraints, linear objective

- cheapest diet that meets nutrition guidelines
  - $y_{vitaminA} = 2.3x_{broccoli} + 1.7x_{carrots} \ldots$
  - $cost = 4.99x_{broccoli} + 2.67x_{carrots} \ldots$
  - minimize $cost$ subject to $y_{vitaminA} > 500 \ldots$

- max flow through network with capacity constraints
- earliest finish time subject to job durations

polynomial time (ellipsoid, Karmarkar’s), but simplex method is popular
CPLEX, Gurobi, Ip_solve
**Beyond Linear Programming**

- **convex programming**: constraints and objective are convex polynomial time
- **quadratic programming**: constraints and objective are quadratic
  - some forms are polynomial time
- **0-1 LP**: 0-1 variables, linear constraints, linear objective
  - NP-complete
- **integer linear programming**: integer variables, linear constraints and objective
  - NP-hard
- **combinatorial optimization**: variables are discrete
Combinatorial Optimization
Types of Search Problems

- Optimization
- Combination Optimization

Types of Problems

- Optimization
- Backtracking
- Depth-first Search
- DFS Order
- ILDS
- ILDS Order
- Hill-Climbing
- EOLQs

- Shortest-path (M&C, vacuum, tile puzzle)
  - want least-cost path to a goal
  - goal depth unknown
  - given operators and their costs

- Constraint satisfaction (map coloring, \( n \)-queens)
  - any goal is fine
  - maximum depth = number of variables
  - given explicit constraints on variables

- Combinatorial optimization (TSP, max-CSP)
  - want least-cost goal
  - maximum depth = number of variables
  - every leaf is a solution
A tree representation of alternatives in a small combinatorial problem.
depth-first search
child ordering
lower bounds
branch-and-bound
Depth-first Search

DFS \((node)\)
1. If is-leaf\((node)\)
2. Visit\((node)\)
3. else
4. For \(i\) from 0 to \(num\text{-}children\)
5. DFS\(\text{child}(node, i)\)
ILDS \((node, \text{allowance}, \text{remaining})\)
1. If is-leaf\((node)\)
2. Visit\((node)\)
3. else
4. If \(\text{allowance} > 0\)
   5. ILDS\((\text{child}(node, 1), \text{allowance} - 1, \text{remaining} - 1)\)
5. If \(\text{remaining} > \text{allowance}\)
   6. ILDS\((\text{child}(node, 0), \text{allowance}, \text{remaining} - 1)\)

start with ILDS\((\text{root}, \text{iteration}, \text{max-depth})\)
The second pass of ILDS visits all leaves with one discrepancy in their path from the root.
Hill-Climbing

Sol ← some random solution (probably poor quality).
Do limit times

New ← random neighbor of Sol.

If New better than Sol,
then Sol ← New.
Hill-Climbing

$Sol \leftarrow$ some random solution (probably poor quality).

Do $limit$ times

$New \leftarrow$ random neighbor of $Sol$.

If $New$ better than $Sol$,
then $Sol \leftarrow New$.

Elaborations: best neighbor (aka gradient-descent)
restarts
simulated annealing
population (GAs, ‘go with the winners’)
Please write down the most pressing question you have about the course material covered so far and put it in the box on your way out.

Thanks!