Local Search Methods

- Tree search keeps unexplored alternatives on the fringe (ensures completeness)

- Local search: improve what you have until you can’t make it better

- Generally much faster and more memory efficient (but incomplete)

This slide deck courtesy of Dan Klein at UC Berkeley
Types of Search Problems

- **Planning problems:**
  - We want a path to a solution (examples?)
  - Usually want an optimal path
  - *Incremental formulations*

- **Identification problems:**
  - We actually just want to know what the goal is (examples?)
  - Usually want an optimal goal
  - *Complete-state formulations*
  - Iterative improvement algorithms
- Random restarts?
- Random sideways steps?
Continuous Problems

- Placing airports in Romania
  - States: \((x_1, y_1, x_2, y_2, x_3, y_3)\)
  - Cost: sum of squared distances to closest city

![Diagram of airport locations in Romania]
Gradient Methods

- How to deal with continuous (therefore infinite) state spaces?
- Discretization: bucket ranges of values
  - E.g. force integral coordinates
- Continuous optimization
  - E.g. gradient ascent

\[ \nabla f = \left( \frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial y_1}, \frac{\partial f}{\partial x_2}, \frac{\partial f}{\partial y_2}, \frac{\partial f}{\partial x_3}, \frac{\partial f}{\partial y_3} \right) \]

\[ x \leftarrow x + \alpha \nabla f(x) \]
What is Search For?

- Models of the world: single agents, deterministic actions, fully observed state, discrete state space

- Planning: sequences of actions
  - The path to the goal is the important thing
  - Paths have various costs, depths
  - Heuristics to guide, fringe to keep backups

- Identification: assignments to variables
  - The goal itself is important, not the path
  - All paths at the same depth (for some formulations)
  - CSPs are specialized for identification problems
Constraint Satisfaction Problems

- **Standard search problems:**
  - State is a “black box”: arbitrary data structure
  - Goal test: any function over states
  - Successor function can be anything

- **Constraint satisfaction problems (CSPs):**
  - A special subset of search problems
  - State is defined by variables $X_i$ with values from a domain $D$ (sometimes $D$ depends on $i$)
  - Goal test is a set of constraints specifying allowable combinations of values for subsets of variables

- Simple example of a *formal representation language*

- Allows useful general-purpose algorithms with more power than standard search algorithms
Example: Map-Coloring

- Variables: \( WA, NT, Q, NSW, V, SA, T \)
- Domain: \( D = \{ \text{red, green, blue} \} \)
- Constraints: adjacent regions must have different colors
  \[ WA \neq NT \]
  \[ (WA, NT) \in \{(\text{red, green}), (\text{red, blue}), (\text{green, red}), \ldots\} \]

- Solutions are assignments satisfying all constraints, e.g.:
  \[ \{WA = \text{red}, NT = \text{green}, Q = \text{red}, NSW = \text{green}, V = \text{red}, SA = \text{blue}, T = \text{green}\} \]
Example: Cryptarithmetic

- **Variables (circles):**
  \[ F T U W R O X_1 X_2 X_3 \]

- **Domains:**
  \[ \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\} \]

- **Constraints (boxes):**
  \[ \text{alldiff}(F, T, U, W, R, O) \]
  \[ O + O = R + 10 \cdot X_1 \]
  \[
  \begin{array}{cccc}
    T & W & O \\
    T & W & O \\
    F & O & U & R
  \end{array}
  \]
Example: Sudoku

- **Variables:**
  - Each (open) square

- **Domains:**
  - \{1,2,\ldots,9\}

- **Constraints:**
  - 9-way alldiff for each column
  - 9-way alldiff for each row
  - 9-way alldiff for each region
Some Hard Questions…

- Who is liable if a robot driver has an accident?
- Will machines surpass human intelligence?
- What will we do with superintelligent machines?
- Would such machines have conscious existence? Rights?
- Can human minds exist indefinitely within machines (in principle)?