Heuristic Optimization
Lecture 1

Algorithm Engineering Group
Hasso Plattner Institute, University of Potsdam

14 April 2015

Optimization

Goal:
- Find $z \in X$ such that $f(z) \leq f(x)$ for all $x \in X$ (minimization)
- Find $z \in X$ such that $f(z) \geq f(x)$ for all $x \in X$ (maximization)

Optimization examples

Linear programming
- $X$ is the set of all vectors $x \in \mathbb{R}^n$ with $Ax \leq b$ and $x \geq 0$
- $f(x) = c^\top x$
- **Goal:** find $x \in X$ such that $f(x)$ is minimal

Example: Schedule production levels of a product to minimize total cost subject to resource constraints.
- Simplex algorithm
- Interior point methods

Convex optimization
- $X$ is the set of all vectors $x \in \mathbb{R}^n$ with some constraints,
- $f(t x + (1-t)y) \leq tf(x) + (1-t)f(y)$ for all $0 \leq t \leq 1$.
- **Goal:** find $x \in X$ such that $f(x)$ is minimal

Example: Find the receiver location among a set of interfering transmitters that maximizes signal to noise ratio.
- Subgradient method
- Cutting plane method
Optimization examples

Find the shortest route between two cities

- \( X \) is the set of feasible paths
- \( f \) measures the length of a path
- **Goal**: find \( x \in X \) such that \( f(x) \) is minimal

**Example**: Navigation software.

- Dijkstra’s algorithm
- Bellman-Ford

The black-box scenario

Suppose we know nothing (or almost nothing) about the function

- \( f(x) \) measures some complex (e.g., industrial) process
- \( f(x) \) value depends on the result of an expensive simulation
- process of assigning \( f \)-values to \( X \) is noisy/unpredictable

How should we approach these problems?

Heuristic Optimization

**Approaches**

- Take a best guess at a good solution and “live with it”
- Try each possible solution and keep the best
- **Start with a good guess and then try to improve it iteratively**

**Heuristic Optimization**

- Can be inspired by *human problem solving*
  - Common sense, rules of thumb, experience
- Can be inspired by *natural processes*
  - Evolution, annealing, swarming behavior
- Typically rely on a source of *randomness* to make decisions
- General purpose, robust methods
- Easy to implement
- **Can be challenging to analyze and prove rigorous results**

Some success stories

**NASA**

- communication antennas on ST-5 mission (evolutionary algorithm)
- deployed on spacecraft in 2006

Some success stories

**Boeing**
- 777 GE engine: turbine geometry evolved with a genetic algorithm


---

**Oral B**
- Cross-action toothbrush design optimized by Creativity Machine (evolutionary algorithm)


---

**Nutech Solutions**
- Improved car frame for GM (genetic algorithms, neural networks, simulated annealing, swarm intelligence)

**BMW**
- Optimized acoustic and safety parameters in car bodies (simulated annealing, genetic and evolutionary algorithms)


---

**Hitachi**
- Improved nose cone for N700 bullet train (genetic algorithm)

Some success stories

Merck Pharmaceutical Company
- discovered first clinically-approved antiviral drug for HIV (Isentress) using AutoDock software (uses a genetic algorithm)

REFERENCE:

Heuristics

Assumptions
1. Solutions encoded as length-$n$ bitstrings (elements of $\{0, 1\}^n$),
2. want to maximize some $f: \{0, 1\}^n \rightarrow \mathbb{R}$.

Random Search
Choose $x$ uniformly at random from $\{0, 1\}^n$;
while stopping criterion not met do
  Choose $y$ uniformly at random from $\{0, 1\}^n$;
  if $f(y) \geq f(x)$ then $x \leftarrow y$;
end

Random(ized) Local Search (RLS)
Choose $x$ uniformly at random from $\{0, 1\}^n$;
while stopping criterion not met do
  $y \leftarrow x$;
  Choose $i$ uniformly at random from $\{1, \ldots, n\}$;
  $y_i \leftarrow (1 - y_i)$;
  if $f(y) \geq f(x)$ then $x \leftarrow y$;
end

Local Optima

How to deal with local optima?
- Restart the process when it becomes trapped (ILS)
- Accept disimproving moves (MA, SA)
- Take larger steps (EA, GA)
**Simple Randomized Search Heuristics**

**Metropolis Algorithm**

Choose \( x \) uniformly at random from \( \{0, 1\}^n \);

while stopping criterion not met do

\[
y \leftarrow x;
\]

Choose \( i \) uniformly at random from \( \{1, \ldots, n\} \);

\[
y_i \leftarrow (1 - y_i);
\]

if \( f(y) \geq f(x) \) then \( x \leftarrow y \);

else \( x \leftarrow y \) with probability \( e^{(f(x)-f(y))/T} \);

end

Method developed for generating sample states of a thermodynamic system (1953)

- \( T \) is fixed over the iterations

**Simulated Annealing**

Choose \( x \) uniformly at random from \( \{0, 1\}^n \);

while stopping criterion not met do

\[
y \leftarrow x, t \leftarrow 0;
\]

Choose \( i \) uniformly at random from \( \{1, \ldots, n\} \);

\[
y_i \leftarrow (1 - y_i);
\]

if \( f(y) \geq f(x) \) then \( x \leftarrow y \);

else \( x \leftarrow y \) with probability \( e^{(f(x)-f(y))/T_t} \);

\[
t \leftarrow t + 1;
\]

end

Heating and controlled cooling of a material to increase crystal size and reduce their defects.

- High temperature \( \Rightarrow \) many random state changes
- Low temperature \( \Rightarrow \) system prefers “low energy” states (high fitness)
- Idea is to carefully settle the system down over time to its lowest energy state (highest fitness) by cooling
- \( T_t \) is dependent on \( t \), typically decreasing.

**Evolutionary Algorithms**

- Allow larger jumps
- Long (destructive) jumps should be rare

\((1+1)\) EA

Choose \( x \) uniformly at random from \( \{0, 1\}^n \);

while stopping criterion not met do

\[
y \leftarrow x;
\]

foreach \( i \in \{1, \ldots, n\} \) do

- With probability \( 1/n \), \( y_i \leftarrow (1 - y_i) \);

end

if \( f(y) \geq f(x) \) then \( x \leftarrow y \);

end