Algorithms in Nature

Optimization

What Is Optimization?

- Selecting an element from a defined set to maximize (or minimize) a given criteria

Input set is usually well defined

A formula to evaluate potential solution is often easy to employ

In many cases optimization involves a search over a set of potential solutions

Formal definiton

- A mathematical formulation
  - *Given*: a function f : $\boldsymbol{A} \rightarrow \mathbf{R}$ from some set A to the real numbers
  - *Sought:* an element x_0 in A such that $f(x_0) \leq f(x)$ for all x in A ("minimization") or such that $f(x_0) \geq f(x)$ for all x in A ("maximization").

Challenges

- Problem / solution can be:
  - High dimensional
  - With lots of local optimum
  - Residing in a huge search space

Examples

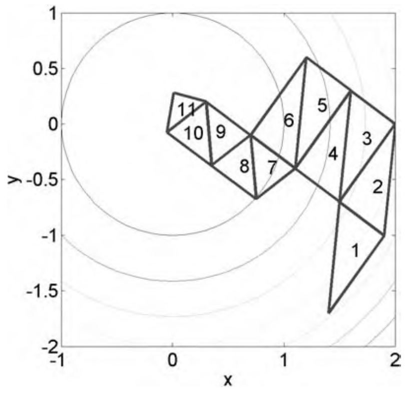
- Minimize the costs of shipping from production facilities to warehouses
- Traveling salesman
- Place sensors in manner to maximize useful information
- Determine the times to administer a sequence of drugs for maximum therapeutic effect
- Find the best red-yellow-green signal timings in an urban traffic network
- Etc …

Minimum-seeking algorithms

1. **Exhaustive Search = Brute Force**

2. **Analytical Optimization** $\nabla f(\mathbf{x}) = 0, \ \dfrac{\partial^2 f}{\partial x} - ?$

3. **Nelder-Mead downhill Simplex Method**



4. **Optimization based on search methods (the coordinate search method, the steepest descent algorithm, Newton's method, etc …)**

Major directions

- Linear Programming, a type of convex programming, studies the case in which the objective function *f* is linear and the set of constraints is specified using only linear equalities and inequalities.
- Nonlinear Programming studies the general case in which the objective function or the constraints or both contain nonlinear parts.

Major directions

- Integer Programming studies linear programs in which some or all variables are constrained to take on integer values. This is not convex, and in general much more difficult than regular linear programming.

Major directions

- Heuristics and Meta-heuristics make few or no assumptions about the problem being optimized. Usually, heuristics do not guarantee that any optimal solution need be found. On the other hand, heuristics are used to find approximate solutions for many complicated optimization problems.

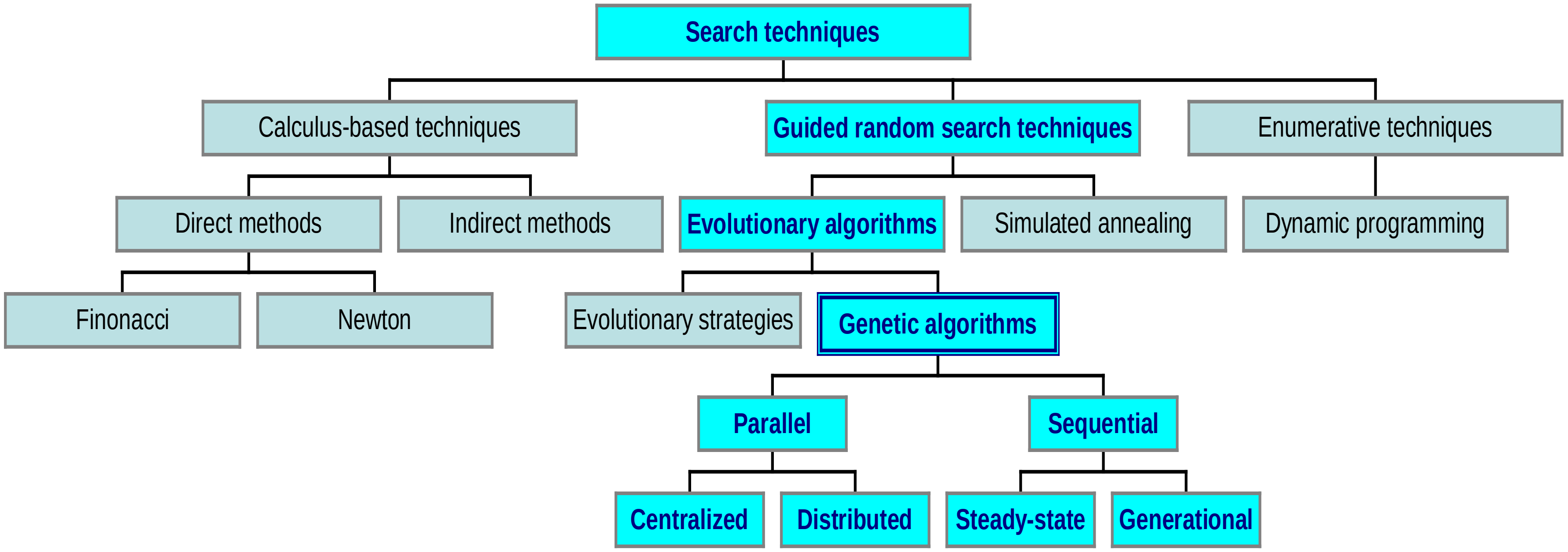
Heuristics

- Heuristic Algorithms
  - Heuristic algorithms are not guaranteed to find the optimal solution.
  - Heuristic algorithms do not even necessarily have a bound on how bad they will perform.
  - However, in practice, heuristic algorithms (heuristics for short) are often successful.

Why Not use Exact Methods?

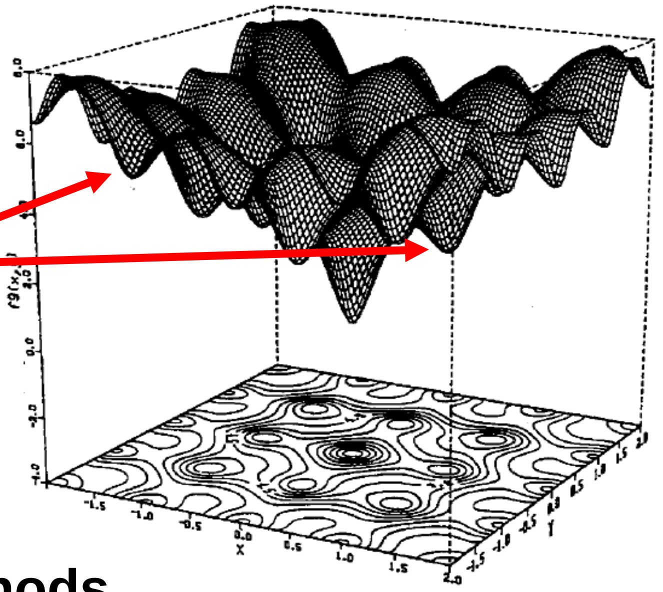
- Run time
  - Depending on the problem, run time of exact methods can be expontnetial
- The link between the real-world problem and the formal problem may be weak
  - Sometimes you cannot properly formulate a mathematical problem that captures all aspects of the real-world problem. If the problem you solve is not the right problem, it might be just as useful to have one (or more) heuristic solutions, rather than the optimal solution for the formal problem.

Classes of Search Techniques



Biologically motivated optimization algorithms

Several local searches can converge to a local minimum!



Natural optimization methods

Not the panacea, but …

Simulated annealing
(Kirkpatrick et al., 1983)

Particle swarm optimization
(Parsopoulos and Vrahatis, 2002)

Genetic algorithms
(Holland, 1975)

Evolutionary algorithms
(Schwefel, 1995)

No derivatives, large search spaces, “nature-based”