Introduction to Neural Networks

Johns Hopkins University
Engineering for Professionals Program

605-447/625-438

Dr. Mark Fleischer

Module 6.2: Implementation of the Simulated Annealing Algorithm

This Sub-Module …

- describes approaches for implementing the Simulated Annealing Program.
- Implementation depends on the type of problem we're trying to solve.
  - We'll look at Combinatorial Optimization Problems (COPs)
  - Continuous variable problems.
  - Touch on approaches for parallizing SA.

Metropolis Acceptance Criterion

States i and j
Objective Function Values f_i and f_j
with $\Delta f_{ji} = f_j - f_i$

$$\Pr\{\text{Accept Cand. } J\} = \begin{cases} e^{-\Delta f_{ji}^{+}/t_k} & \Delta f_{ji} > 0, j \in N(i) \\ 1 & \Delta f_{ji} \leq 0, j \in N(i) \\ 0 & \text{otherwise} \end{cases}$$

The Simulated Annealing Algorithm

1. Select a new state with a new objective function value.
2. Calculate the value of Δf.
3. If $\Delta f \leq 0$ (for a minimization problem) then the new state becomes the current state. If $\Delta f > 0$ then the new state becomes the current state via the Metropolis Acceptance Criterion. If state J is not accepted, keep state I as the current state.
4. Lower temperature and goto 1.

Requirements for SA

1. Define an appropriate set of states.
2. Define a suitable neighborhood structure.

   This entails the candidate generation mechanism.
3. Define a suitable cost/objective function.

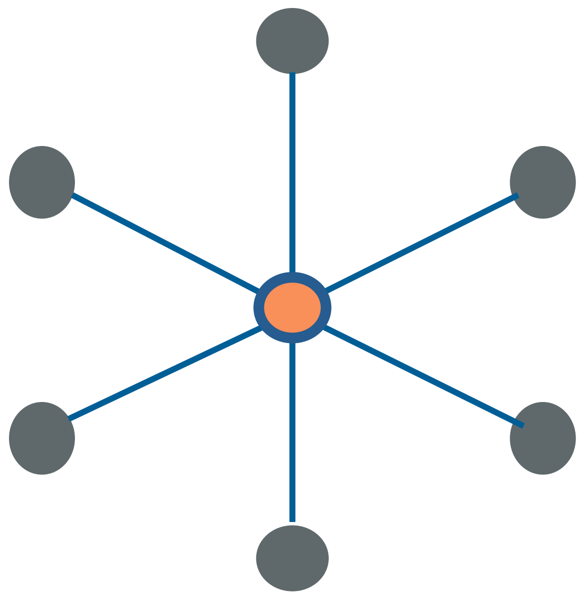
   This entails a method for calculating the change in objective function value.
4. Define a cooling schedule.
5. Define stopping criteria.

Cooling Schedules

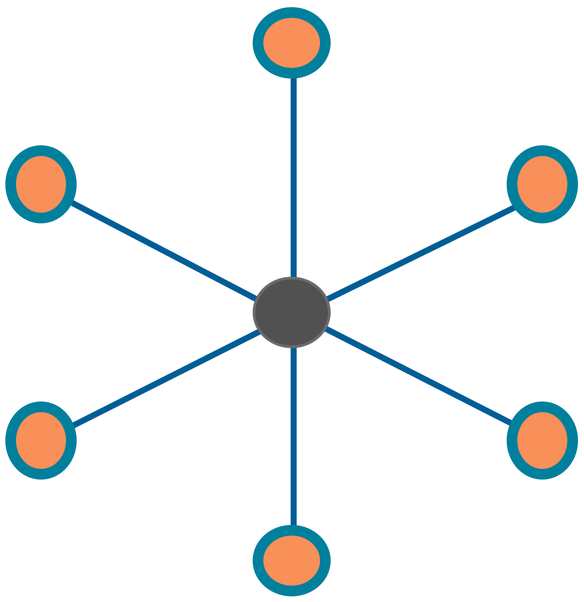
$$t_k = \frac{\gamma}{\log(c+k)}$$

$$t_k = \frac{\gamma}{c+k}$$

Combinatorial Optimization Problems (COPs)

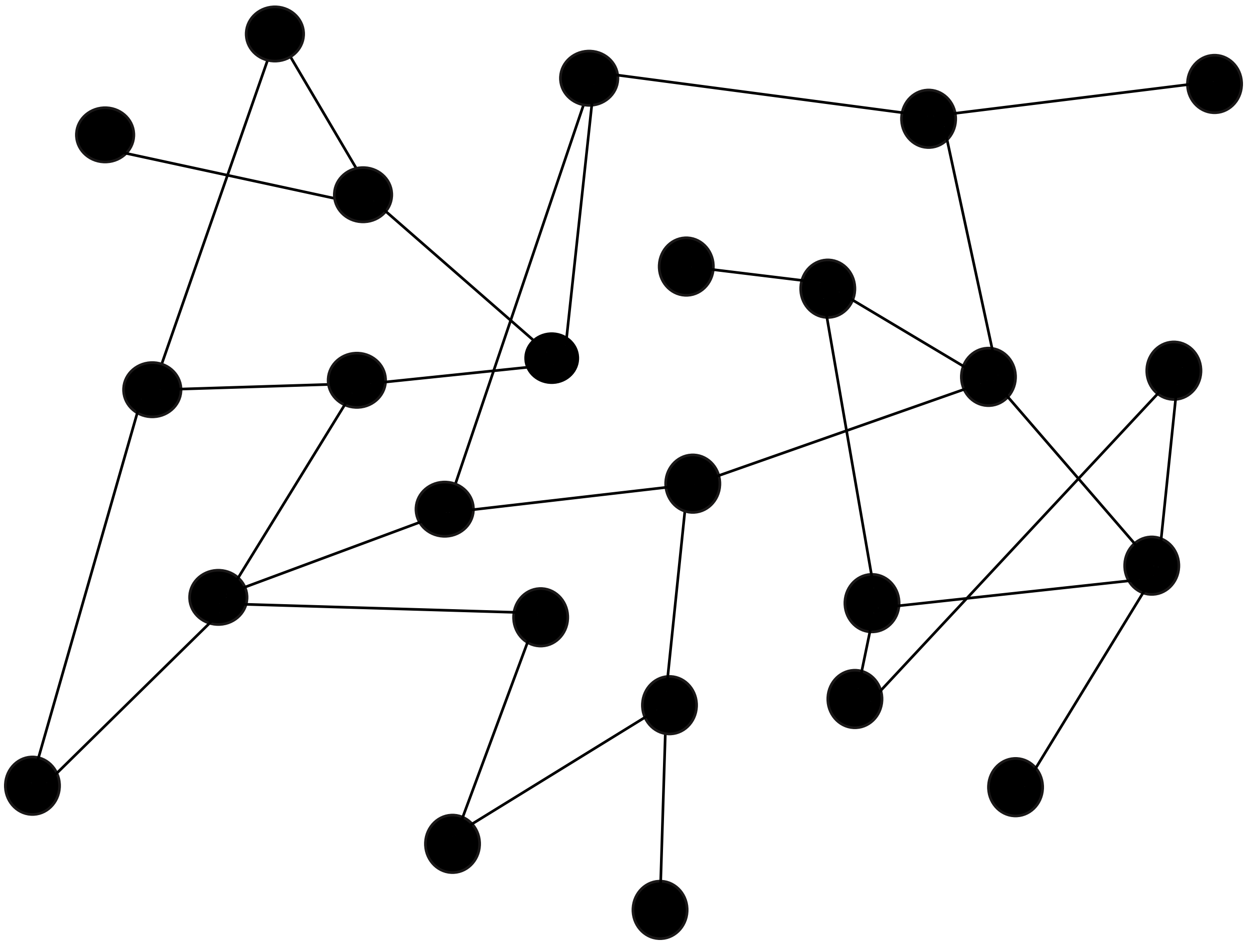


Minimum Vertex Cover



Maximum Independent Set

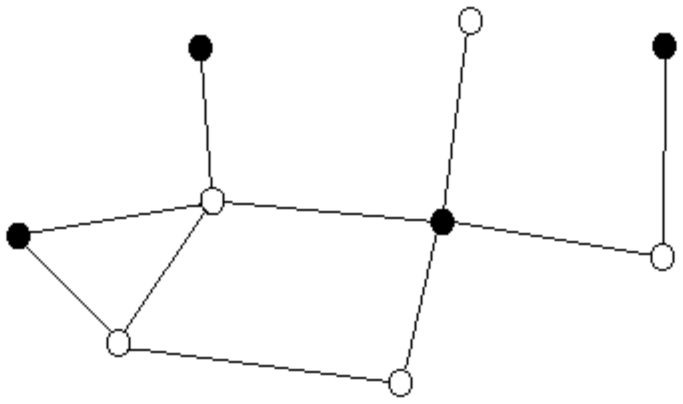
Not Always So Obvious . . .



Adding Insult to Injury. . .

$2^{1,000,000}$? Yikes!

Implementation Issues

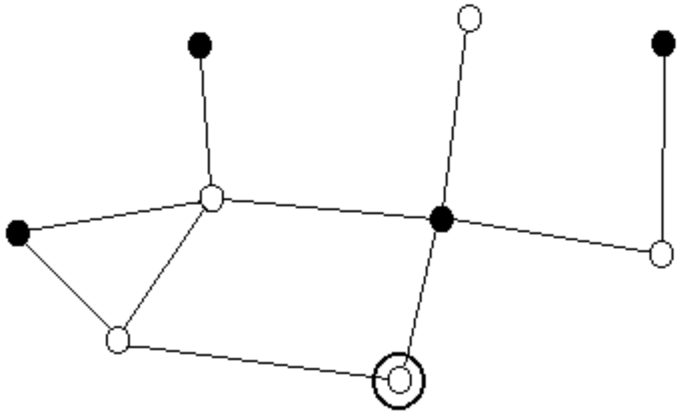


A good objective function is simply
the number of nodes in the current set.

Current Solution = 4
Maximum Set Size = 5

How do we get to better solutions
without doing a lot of work?

Select a node, any node…



But this violates independent set constraints!
Use some penalty function.
Penalty function should decrease
objective function value
and
be based on those elements which violate constraints.

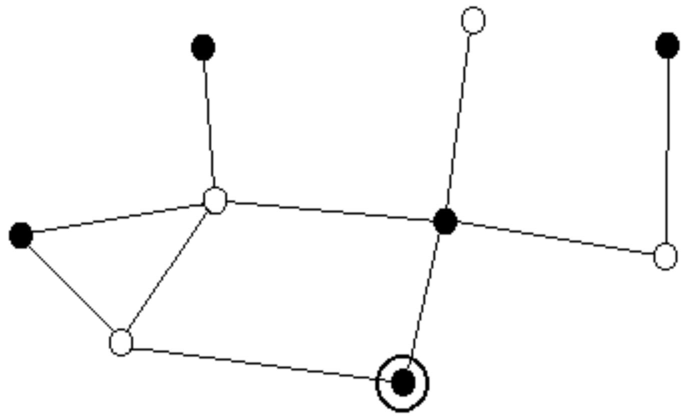
Objective Functions

What are the elements which violate constraints?

edges

f(G) = number of nodes – g(edges)

New Combination of Nodes, New Objective Function Value

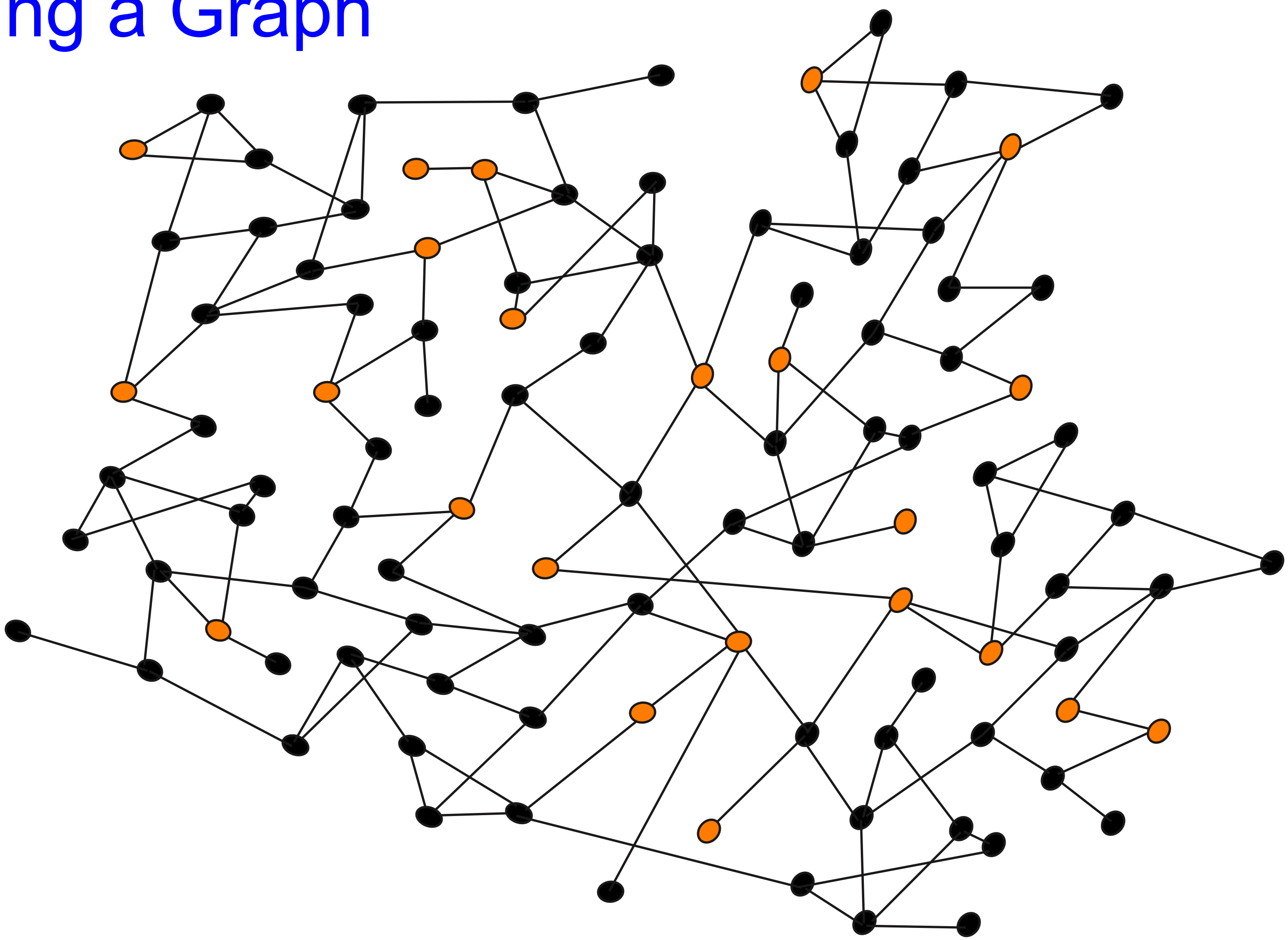


Current Objective Function Value:

From $$f_{\text{indep set}}(V',G) = v' - \lambda E_1(V') \quad = \quad 4 - 0$$

to $$f_{\text{indep set}}(V',G) = v' - \lambda E_1(V') = 5 - \lambda 1$$

Annealing a Graph



E.g. the Stationary Probability

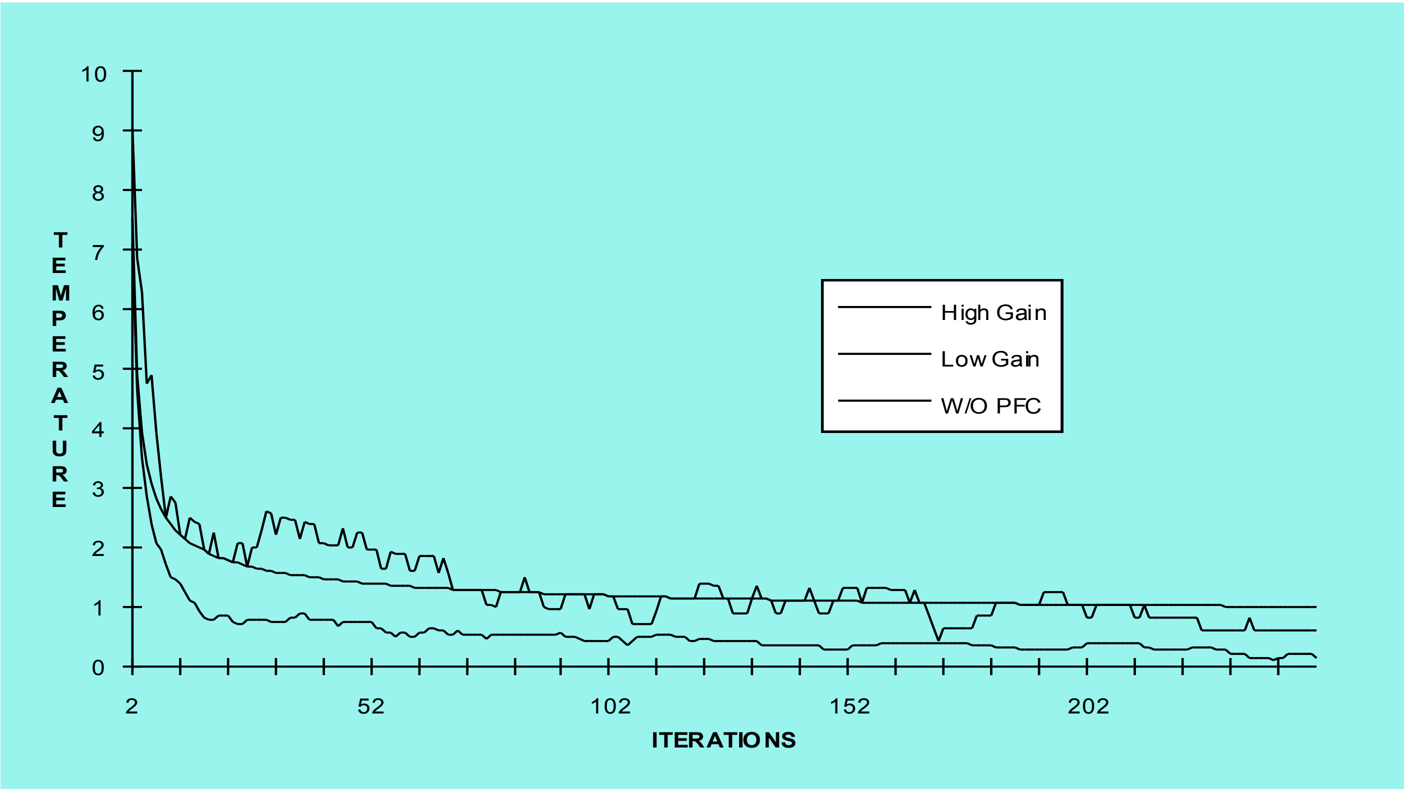
$$\pi_i(t) = \frac{e^{-f_i/t}}{\sum_{i=1}^{s} e^{-f_i/t}}.$$

Boltzmann Distribution Function

Joint Distribution

$$
\begin{aligned}
\pi_{i_1 \cdots i_p}(t) &= \prod_{m=1}^{p} \pi_{i_m}(t) \\
&= \prod_{m=1}^{p} \frac{e^{-f_{i_m}/t}}{\sum_{i_m=1}^{s} e^{-f_{i_m}/t}} \\
&= \frac{e^{-\left(\sum_{m=1}^{p} f_{i_m}\right)/t}}{\prod_{m=1}^{p} \sum_{i_m=1}^{s} e^{-f_{i_m}/t}} \\
&= \frac{e^{-f_{i_1 \cdots i_p}/t}}{\sum_{i_1 \cdots i_p}^{s^p} e^{-f_{i_1 \cdots i_p}/t}}
\end{aligned}
$$

Non-Monotonic Cooling



Journal of Heuristics Vol. 1 No. 2 p.245

Experimental Results and Observations

Number of Processors	Gain Setting	Objective Function Values: With PFC $\overline{F}_{\min}$	$s_{\mathbf{F}}^2$	No PFC $\overline{G}_{\min}$	$s_{\mathbf{G}}^2$	z-value
5	1	0.683	0.303	0.541	0.057	1.290
5	5	0.212	0.057	0.536	0.536	-2.300
5	10	0.232	0.090	0.541	0.057	-4.419
10	1	0.488	0.560	0.388	0.040	0.709
10	5	0.130	0.027	0.324	0.047	-3.899
10	10	0.110	0.020	0.438	0.062	-6.261

Table 5.1 : Efficacy of PFC by Candidate Generation

What About Continuous Variable Problems?

- Picking candidate solutions is main issue.
- SA doesn't work well on continuous variable problems.

Recursive Intensification

- Accelerates convergence to optima, experimentally.
- Increases probability of never converging to the optima.