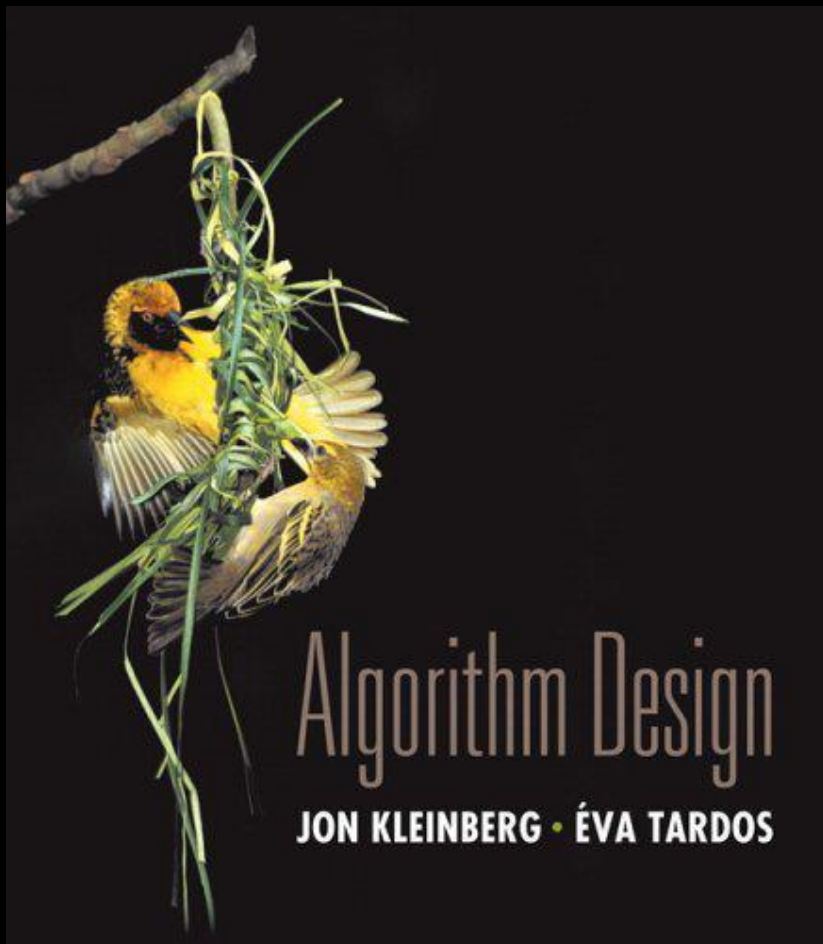


Chapter 12

Local Search



Slides by Kevin Wayne.
Copyright © 2005 Pearson-Addison Wesley.
All rights reserved.

Coping With NP-Hardness

Q. Suppose I need to solve an NP-hard problem. What should I do?

A. Theory says you're unlikely to find poly-time algorithm.

Must sacrifice one of three desired features.

- Solve problem to optimality.
- Solve problem in polynomial time.
- Solve arbitrary instances of the problem.

12.1 Landscape of an Optimization Problem

Gradient Descent: Vertex Cover

VERTEX-COVER. Given a graph $G = (V, E)$, find a subset of nodes S of minimal cardinality such that for each $u-v$ in E , either u or v (or both) are in S .

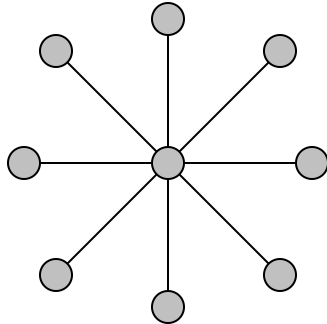
Neighbor relation. $S \sim S'$ if S' can be obtained from S by adding or deleting a single node. Each vertex cover S has at most n neighbors.

Gradient descent. Start with $S = V$. If there is a neighbor S' that is a vertex cover and has lower cardinality, replace S with S' .

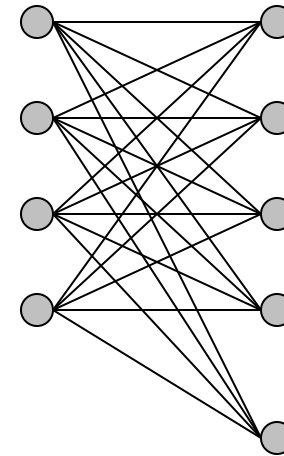
Remark. Algorithm terminates after at most n steps since each update decreases the size of the cover by one.

Gradient Descent: Vertex Cover

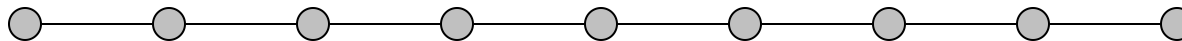
Local optimum. No neighbor is strictly better.



optimum = center node only
local optimum = all other nodes



optimum = all nodes on left side
local optimum = all nodes on right side



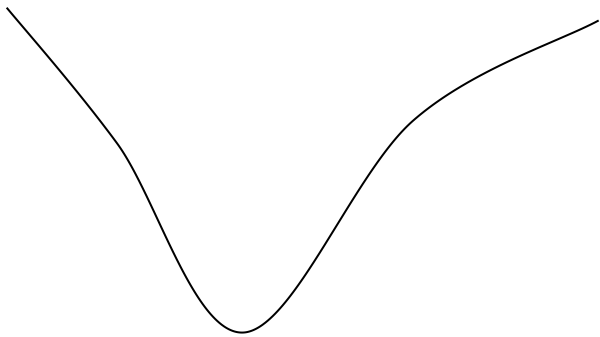
optimum = even nodes
local optimum = omit every third node

Local Search

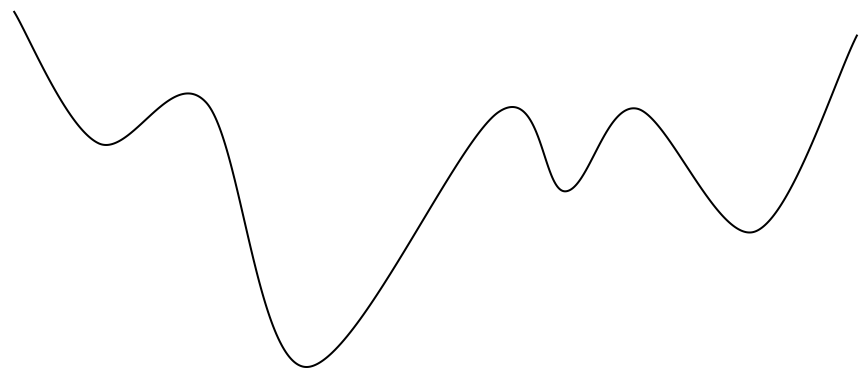
Local search. Algorithm that explores the space of possible solutions in sequential fashion, moving from a current solution to a "nearby" one.

Neighbor relation. Let $S \sim S'$ be a neighbor relation for the problem.

Gradient descent. Let S denote current solution. If there is a neighbor S' of S with strictly lower cost, replace S with the neighbor whose cost is as small as possible. Otherwise, terminate the algorithm.



A funnel



A jagged funnel

12.4 Maximum Cut

Maximum Cut

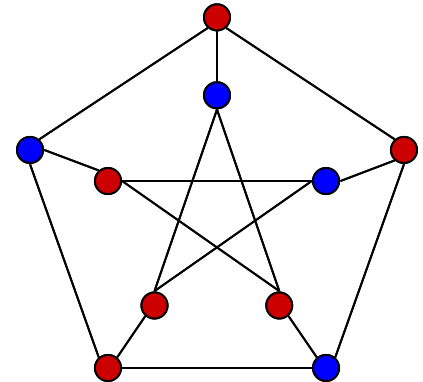
Maximum cut. Given an undirected graph $G = (V, E)$ with positive integer edge weights w_e , find a node partition (A, B) such that the total weight of edges crossing the cut is maximized.

$$w(A, B) \doteq \sum_{u \in A, v \in B} w_{uv}$$

Toy application.

- n activities, m people.
- Each person wants to participate in two of the activities.
- Schedule each activity in the morning or afternoon to maximize number of people that can enjoy both activities.

Real applications. Circuit layout, statistical physics.



Maximum Cut

Single-flip neighborhood. Given a partition (A, B) , move one node from A to B , or one from B to A if it improves the solution.

Greedy algorithm.

MAX – CUT – LOCAL (G, w)

```
1:  $(A, B) \leftarrow$  random cut.
2: while there exists an improving node  $v$  do
3:   if  $v \notin A$  then
4:      $A \leftarrow A \cup \{v\}$ .
5:      $B \leftarrow B - \{v\}$ .
6:   else
7:      $B \leftarrow B \cup \{v\}$ .
8:      $A \leftarrow A - \{v\}$ .
9:   end if
10: end while
11: return  $(A, B)$ .
```

Maximum Cut: Local Search Analysis

Theorem. Let (A, B) be a locally optimal partition and let (A^*, B^*) be optimal partition. Then $w(A, B) \geq \frac{1}{2} \sum_e w_e \geq \frac{1}{2} w(A^*, B^*)$.

↑
weights are nonnegative

Pf.

- Local optimality implies that for all $u \in A$: $\sum_{v \in A} w_{uv} \leq \sum_{v \in B} w_{uv}$

Adding up all these inequalities yields:

$$2 \sum_{\{u,v\} \subseteq A} w_{uv} \leq \sum_{u \in A, v \in B} w_{uv} = w(A, B)$$

- Similarly $2 \sum_{\{u,v\} \subseteq B} w_{uv} \leq \sum_{u \in A, v \in B} w_{uv} = w(A, B)$

- Now,

each edge counted once

$$\begin{aligned} \downarrow \\ \sum_{e \in E} w_e &= \underbrace{\sum_{\{u,v\} \subseteq A} w_{uv}}_{\leq \frac{1}{2} w(A, B)} + \underbrace{\sum_{u \in A, v \in B} w_{uv}}_{w(A, B)} + \underbrace{\sum_{\{u,v\} \subseteq B} w_{uv}}_{\leq \frac{1}{2} w(A, B)} \leq 2w(A, B) \quad \blacksquare \end{aligned}$$

Maximum Cut: Big Improvement Flips

Local search. Within a factor of 2 for MAX-CUT, but not poly-time!

Big-improvement-flip algorithm. Only choose a node which, when flipped, increases the cut value by at least $\frac{2\varepsilon}{n} w(A, B)$

Claim. Upon termination, big-improvement-flip algorithm returns a cut (A, B) with $(2 + \varepsilon) w(A, B) \geq w(A^*, B^*)$.

Pf idea. Add $\frac{2\varepsilon}{n} w(A, B)$ to each inequality in original proof.

Claim. Big-improvement-flip algorithm terminates after $O(\varepsilon^{-1} n \log W)$ flips, where $W = \sum_e w_e$.

- Each flip improves cut value by at least a factor of $(1 + \varepsilon/n)$.
- After n/ε iterations the cut value improves by a factor of 2.
- Cut value can be doubled at most $\log W$ times.

↑
if $x \geq 1$, $(1 + 1/x)^x \geq 2$

Maximum Cut: Context

Theorem. [Sahni-Gonzales 1976] There exists a $\frac{1}{2}$ -approximation algorithm for MAX-CUT.

Theorem. [Goemans-Williamson 1995] There exists an 0.878567-approximation algorithm for MAX-CUT.

$$\min_{0 \leq \theta \leq \pi} \frac{2}{\pi} \frac{\theta}{1 - \cos \theta}$$

Theorem. [Håstad 1997] Unless $P = NP$, no $16/17$ approximation algorithm for MAX-CUT.

$$\uparrow$$

0.941176

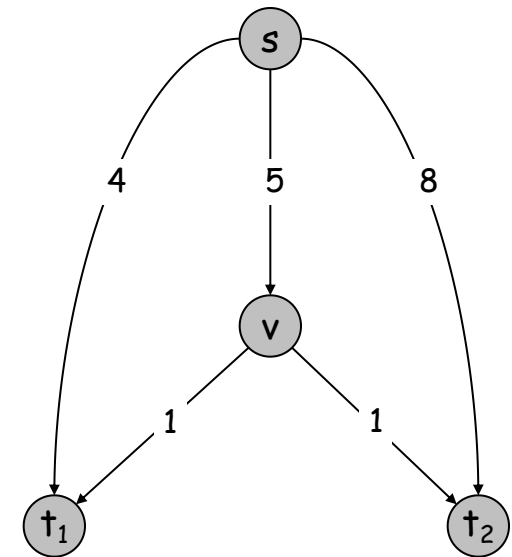
12.7 Nash Equilibria

Multicast Routing

Multicast routing. Given a directed graph $G = (V, E)$ with edge costs $c_e \geq 0$, a source node s , and k agents located at terminal nodes t_1, \dots, t_k . Agent j must construct a path P_j from node s to its terminal t_j .

Fair share. If x agents use edge e , they each pay c_e / x .

1	2	1 pays	2 pays
outer	outer	4	8
outer	middle	4	$5 + 1$
middle	outer	$5 + 1$	8
middle	middle	$5/2 + 1$	$5/2 + 1$



Nash Equilibrium

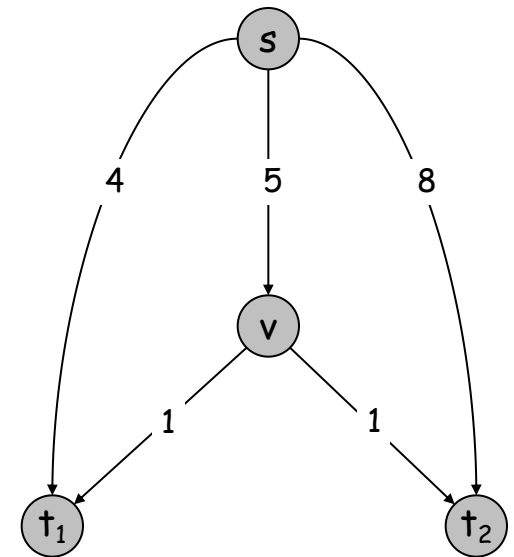
Best response dynamics. Each agent is continually prepared to improve its solution in response to changes made by other agents.

Nash equilibrium. Solution where no agent has an incentive to switch.

Fundamental question. When do Nash equilibria exist?

Ex:

- Two agents start with outer paths.
- Agent 1 has no incentive to switch paths (since $4 < 5 + 1$), but agent 2 does (since $8 > 5 + 1$).
- Once this happens, agent 1 prefers middle path (since $4 > 5/2 + 1$).
- Both agents using middle path is a Nash equilibrium.



Nash Equilibrium and Local Search

Local search algorithm. Each agent is continually prepared to improve its solution in response to changes made by other agents.

Analogies.

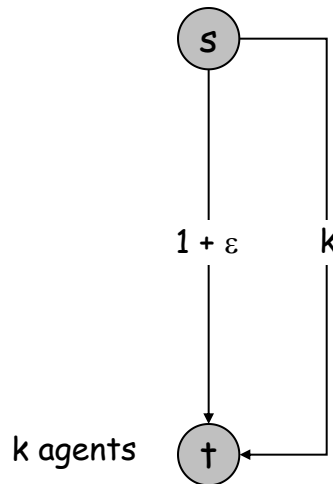
- Nash equilibrium : local optimum.
- Best response dynamics : local search algorithm.
- Unilateral move by single agent : local neighborhood.

Contrast. Best-response dynamics need not terminate since no single objective function is being optimized.

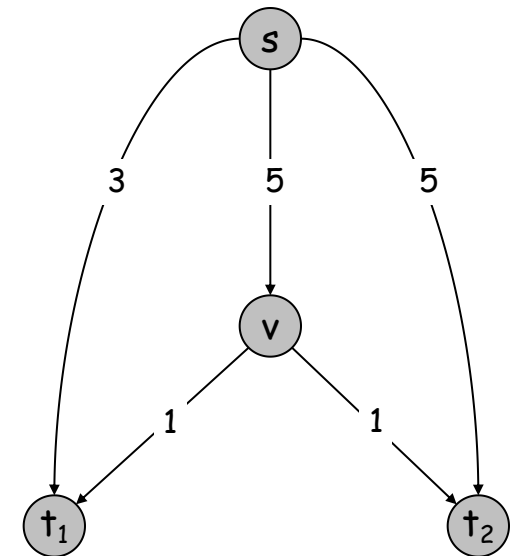
Socially Optimum

Social optimum. Minimizes total cost to all agent.

Observation. In general, there can be many Nash equilibria. Even when its unique, it does not necessarily equal the social optimum.



Social optimum = $1 + \varepsilon$
Nash equilibrium A = $1 + \varepsilon$
Nash equilibrium B = k



Social optimum = 7
Unique Nash equilibrium = 8

Price of Stability

Price of stability. Ratio of best Nash equilibrium to social optimum.

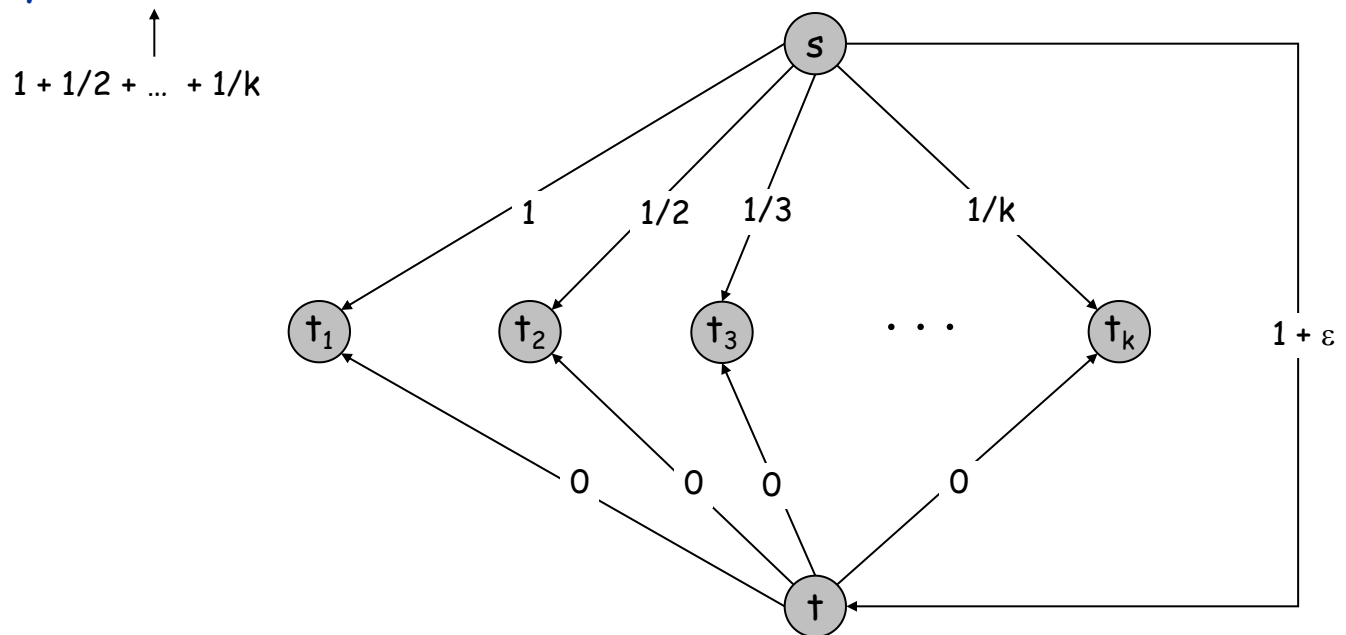
Fundamental question. What is price of stability?

Ex: Price of stability = $\Theta(\log k)$.

Social optimum. Everyone takes bottom paths.

Unique Nash equilibrium. Everyone takes top paths.

Price of stability. $H(k) / (1 + \varepsilon)$.



Finding a Nash Equilibrium

Theorem. The following algorithm terminates with a Nash equilibrium.

BEST – RESPONSE – DYNAMICS(G, c, k)

```
1: for  $j = 1$  to  $k$  do
2:    $P_j \leftarrow$  any path for agent  $j$ .
3: end for
4: while not a Nash equilibrium do
5:    $j \leftarrow$  some agent who can improve by switching paths.
6:    $P_j \leftarrow$  better path for agent  $j$ .
7: end while
8: return  $(P_1, P_2, \dots, P_k)$ .
```

Pf. Consider a set of paths P_1, \dots, P_k .

- Let x_e denote the number of paths that use edge e .
- Let $\Phi(P_1, \dots, P_k) = \sum_{e \in E} c_e \cdot H(x_e)$ be a potential function.
- Since there are only finitely many sets of paths, it suffices to show that Φ strictly decreases in each step.

\swarrow $H(0) = 0, H(k) = \sum_{i=1}^k \frac{1}{i}$

Finding a Nash Equilibrium

Pf. (continued)

- Consider agent j switching from path P_j to path P_j' .
- Agent j switches because

$$\underbrace{\sum_{f \in P_j' - P_j} \frac{c_f}{x_f + 1}}_{\text{newly incurred cost}} < \underbrace{\sum_{e \in P_j - P_j'} \frac{c_e}{x_e}}_{\text{cost saved}}$$

- Φ increases by $\sum_{f \in P_j' - P_j} c_f [H(x_f + 1) - H(x_f)] = \sum_{f \in P_j' - P_j} \frac{c_f}{x_f + 1}$
- Φ decreases by $\sum_{e \in P_j - P_j'} c_e [H(x_e) - H(x_e - 1)] = \sum_{e \in P_j - P_j'} \frac{c_e}{x_e}$
- Thus, net change in Φ is negative. ▀

Bounding the Price of Stability

Claim. Let $C(P_1, \dots, P_k)$ denote the total cost of selecting paths P_1, \dots, P_k . For any set of paths P_1, \dots, P_k , we have

$$C(P_1, \dots, P_k) \leq \Phi(P_1, \dots, P_k) \leq H(k) \cdot C(P_1, \dots, P_k)$$

Pf. Let x_e denote the number of paths containing edge e .

- Let E^+ denote set of edges that belong to at least one of the paths.

$$C(P_1, \dots, P_k) = \sum_{e \in E^+} c_e \leq \underbrace{\sum_{e \in E^+} c_e H(x_e)}_{\Phi(P_1, \dots, P_k)} \leq \sum_{e \in E^+} c_e H(k) = H(k) C(P_1, \dots, P_k)$$

Bounding the Price of Stability

Theorem. There is a Nash equilibrium for which the total cost to all agents exceeds that of the social optimum by at most a factor of $H(k)$.

Pf.

- Let (P_1^*, \dots, P_k^*) denote set of socially optimal paths.
- Run best-response dynamics algorithm starting from P^* .
- Since Φ is monotone decreasing $\Phi(P_1, \dots, P_k) \leq \Phi(P_1^*, \dots, P_k^*)$.

$$C(P_1, \dots, P_k) \leq \Phi(P_1, \dots, P_k) \leq \Phi(P_1^*, \dots, P_k^*) \leq H(k) \cdot C(P_1^*, \dots, P_k^*)$$

\uparrow
 previous claim
 applied to P

\uparrow
 previous claim
 applied to P*

Summary

Existence. Nash equilibria always exist for k -agent multicast routing with fair sharing.

Price of stability. **Best** Nash equilibrium is never more than a factor of $H(k)$ worse than the social optimum.

Fundamental open problem. **Find any** Nash equilibria in poly-time.