CS 124 lecture 22 Monday, April 20, 2020

## Reductions

$A \leq_p B$

NPC problems: solve one, you can solve them all (in poly time)

Solving NPC:

1. approximation
2. randomized algorithms

   randomized polynomial time (+ coin flips)

   $RP \stackrel{?}{=} NP$
3. Restrict the inputs

   ex: 2SAT, Horn Formulas
4. Small instances?
5. Heuristics

   possible approximation algorithms we cannot prove anything about

## Local Search

space of possible solutions

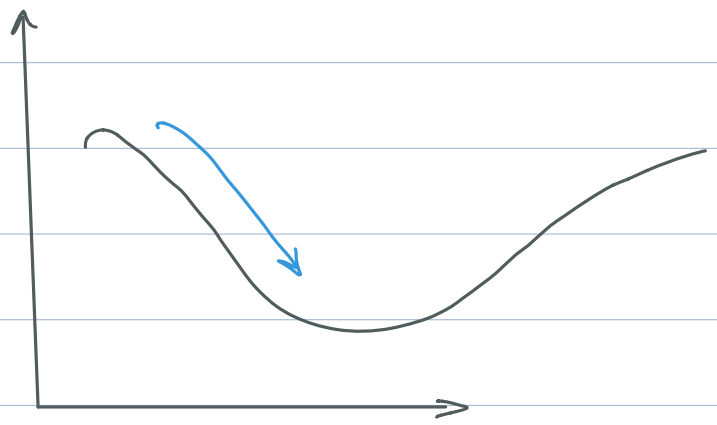
each solution has "neighbors"

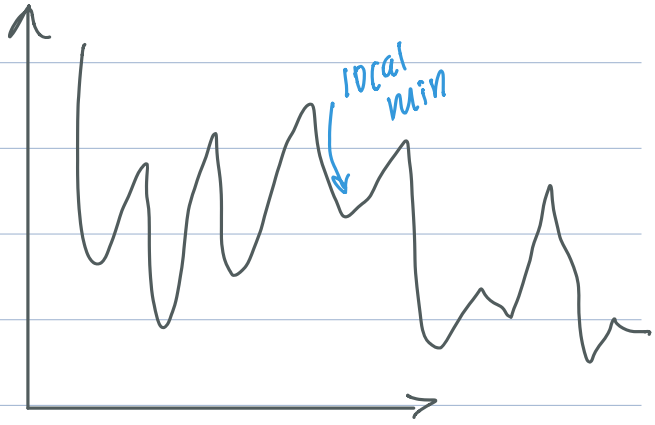
some sort of cost function

Minimize:

1. pick starting point x
2. While $\exists$ a better neighbor y s.t. $f(y) < f(x)$, go to y

   cannot $\leq$ in practice but check for cycling
3. return final solution






example: max 3SAT

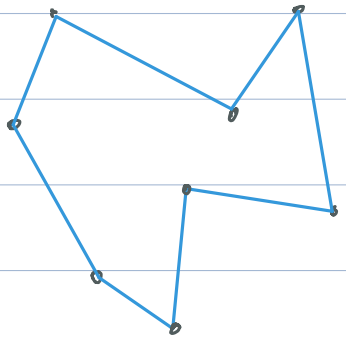
satisfy as many clauses as possible

solution: any truth assignment ($2^n$ vertices)

neighbor: 1 change of variable

↳ or satisfy a given clause

example: traveling salesperson problem



solution: ordering of cities

neighbor: swap two cities

more generally, change $k$ edges

Simplex minimize

1. pick a starting point $x$
2. while $\exists$ a better neighbor $y$ s.t. $f(y) < f(x)$, go to $y$
3. return final solution

# Programming Assignment

collection of #s

split into two sets A, B

$\min |\sum_{x \in A} x - \sum_{y \in B} y|$

★ NP-complete

DP-solution: $O(n B)$ — $n$: # of numbers, $B$: biggest

solution: two sets of numbers
neighbors: sway between two sets
} local search

random moves
- probability $p$ pick 1 and move from A to B
- prob $1-p$ pick 1 move from B to A

## Karmarkar-Karp

10 8 7 6 5

2 1 3 2 → residue

10 and 8 in different sets ≡ element of size 2

## solution: pre-partitioning

5 8 7 6 3 (8 + 7 = 15)

groups 1 2 2 3 4

→ Karmarkar-Karp → residue

## Metropolis Alg

- Pick different neighbor
- if better → go there
  if worse → go there w/ prob & how worse it is

## Simulated Annealing

Temperature → make more worse spots at the beginning

As time that goes on → "reduce temperature" and become like hill climbing

## Tabu Search

Penalty to prevent cycling (solutions that look like other solutions)

## Parallel Search (Go w/ the winners)

Synchronize - check current solutions
Remove poor performers
Clone good ones

## Genetic Algorithms

"population" of solutions
Bad solutions "die off"
Good solutions "procreate"

## Repeated Greedy (Bubblesearch)

Set cover

- normally: pick set w/ greatest uncovered coverings
- introduce randomness: best w/ prob $p$
  2nd best set w/ prob $q$
  ...

## Approximations

provable guarantees

multiplicative guarantee (approximation ratio)

- within a factor of $x$ from optimal
- $(1+\varepsilon)$ of optimal

## Vertex Cover

- special case of set cover
- $O(\log n)$ approx (for set cover)

  Greedy is $O(\log n)$ approx for VC
- 2-approx for VC

  pick an edge
  put both in VC
  remove all adjacent edges
  until done