Approximability

$\mu\Pi\lambda\forall$

Definitions

- Vertex Cover
- Set Cover
- MaxSat
- Clique
- Dknapsack
- TSP

Optimization problem

- Optimization problem P characterized by
  - Set of instances I
  - Function SOL that associates to any instance the set of feasible solutions
  - Measure function m that, for any feasible solution, provides its positive integer value
  - Goal, that is, either MAX or MIN
- An optimal solution is a feasible solution y^* such that

$$m(x,y^*) = \text{Goal}\{m(x,y) \mid y \in \text{SOL}(x)\}$$

- For any instance x, $m^*(x)$ denotes optimal measure

Optimization Problems

- many hard problems (especially **NP**-hard) are optimization problems
  - e.g. find *shortest* TSP tour
  - e.g. find *smallest* vertex cover
  - e.g. find *largest* clique

  - may be minimization or maximization problem
  - "opt" = value of optimal solution

Approximation Algorithms

- often happy with approximately optimal solution
  - warning: lots of heuristics
  - we want approximation algorithm with guaranteed approximation ratio of ρ
  - meaning: on every input x, output is guaranteed to have value

    at most ρ*opt for minimization

    at least opt/ρ for maximization

MINIMUM VERTEX COVER

- INSTANCE: Graph $G=(V,E)$
- SOLUTION: A subset U of V such that, for any edge (u,v), either u is in U or v is in U
- MEASURE: Cardinality of U

Three problems in one

- **Constructive problem**: given an instance, compute an optimal solution and its value
  - We will study these problems
- **Evaluation problem**: given an instance, compute the optimal value
- **Decision problem**: given an instance and an integer k, decide whether the optimal value is at least (if Goal=MAX) or at most (if Goal=MIN) k

Class NPO

- Optimization problems such that
  - I is recognizable in polynomial time
  - Solutions are polynomially bounded (in length) and recognizable in polynomial time
  - m is computable in polynomial time
- Example: MINIMUM VERTEX COVER
- **Theorem** : If P is in NPO, then the corresponding decision problem is in NP

Class PO

- NPO problems solvable in polynomial time
- Example: SHORTEST PATH

- An optimization problem P is *NP-hard* if any problem in NP is Turing reducible to P
- **Theorem**: If the decision problem corresponding to a NPO problem P is *NP-complete*, then P is *NP-hard*
  - Example: MINIMUM VERTEX COVER
- **Corollary**: If P $\neq$ NP then PO $\neq$ NPO

Evaluating versus constructing

- Decision problem is Turing reducible to evaluation problem
- Evaluation problem is Turing reducible to constructive problem
- Evaluation problem is Turing reducible to decision problem
  - Binary search on space of possible measure values
- Is constructive problem Turing reducible to evaluation problem?

MAXIMUM SATISFIABILITY

- INSTANCE: CNF Boolean formula, that is, set *C* of clauses over set of variables *V*
- SOLUTION: A truth-assignment *f* to *V*
- MEASURE: Number of satisfied clauses

Evaluating versus constructing: MAX SAT

```
begin
    for each variable v
    begin
        k := MAX SAT_eval(x);
        x_TRUE := formula obtained by setting v to TRUE in x;
        x_FALSE := formula obtained by setting v to FALSE in x;
        if MAX SAT_eval(x_TRUE) = k then
        begin
            f(v) := TRUE; x := x_TRUE
        end
        else
        begin
            f(v) := FALSE; x := x_FALSE
        end;
    return f
end.
```

Theorem: if the decision problem is NP-complete, then the constructive problem is Turing reducible to the decision problem

Performance ratio

Given an optimization problem P, an instance x and a feasible solution y, the performance ratio of y with respect to x is

$$R(x,y) = \max(m(x,y)/m^*(x),\ m^*(x)/m(x,y))$$

An algorithm is said to be an r-approximation algorithm if, for any instance x, returns a solution whose performance ratio is at most r

MINIMUM BIN PACKING

- INSTANCE: Finite set I of rational numbers $\{a_1,\ldots,a_n\}$ with $a_i \in (0,1]$
- SOLUTION: Partition $\{B_1,\ldots,B_k\}$ of I into k bins such that the sum of the numbers in each bin is at most 1
- MEASURE: Cardinality of the partition, i.e., k

Sequential algorithm

- Polynomial-time 2-approximation algorithm for MINIMUM BIN PACKING
  - *Next Fit* algorithm

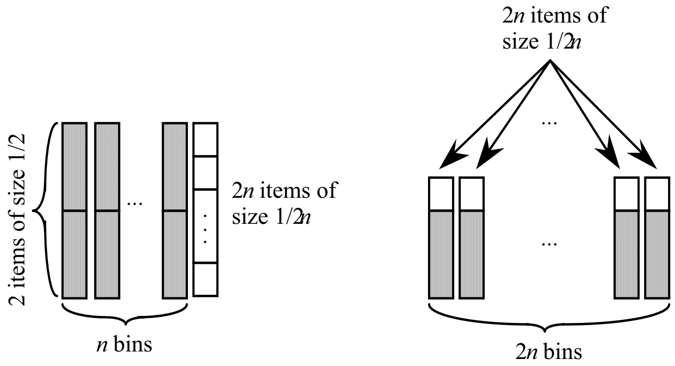
```
begin
    for each number a
        if a fits into the last open bin then assign a to this bin
        else open new bin and assign a to this bin
    return f
end.
```


Proof

- Number of bins used by the algorithm is at most $2A$, where A is the sum of all numbers
  - For each pair of consecutive bins, the sum of the number included in these two bins is greater than 1
- Each feasible solution uses at least A bins
  - Best case each bin is full (i.e., the sum of its numbers is 1)
- Performance ratio is at most 2
- **Theorem**: *First Fit Decreasing* computes solution whose measure is at most $1.5m^*(x)+1$

Tightness

- Let $I=\{1/2,1/2n,1/2,1/2n,\ldots,1/2,1/2n\}$ contain $4n$ items



optimal packing

Next Fit packing

Gavril's algorithm for vertex cover

```
begin
    U=ø;
    for any edge (u,v) do
        if (u is not in U) and (v is not in U) then
            insert u and v in U;
    return U
end.
```

- **Theorem**: Gavril's algorithm is a polynomial-time 2-approximation algorithm

MINIMUM GRAPH COLORING

- INSTANCE: Graph $G=(V,E)$
- SOLUTION: A coloring of V, that is, function f such that, for any edge (u,v), $f(u) \neq f(v)$
- MEASURE: Number of colors, i.e., cardinality of the range of f

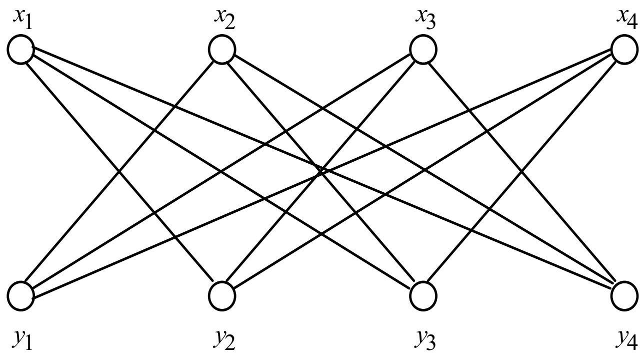
Sequential algorithm: bad

```
begin
    order V with respect to the degree;
    for each node v do
        if there exists color not used by neighbors of v then assign this
            color to v
        else create new color and assign it to v
end.
```

Example

The performance ratio is 2

Generalizing, the performance ratio is $n/2$, where n is the number of nodes



Class APX

- NPO problems P that admit a polynomial-time r–approximation algorithm, for given constant $r \geq 1$
  - P is said to be r-approximable

- Examples: MINIMUM BIN PACKING, MAXIMUM SAT, MAXIMUM CUT, MINIMUM VERTEX COVER

Class PTAS

- NPO problems P that admit a polynomial-time r–approximation algorithm, for any $r > 1$
  - Time must be polynomial in the length of the instance but not necessarily in $1/(r\text{-}1)$
    - Time complexity $O(n^{1/(r-1)})$ or $O(2^{1/(r-1)}n^3)$
  - P is said to admit a polynomial-time approximation scheme

- Example: MINIMUM PARTITION

The NPO world

NPO	
APX	MINIMUM BIN PACKING MAXIMUM SAT MAXIMUM CUT MINIMUM VERTEX COVER
PTAS	MINIMUM PARTITION
PO	MINIMUM PATH

Non-Approximability Results

Summary

- Gap technique
  - Examples: MINIMUM GRAPH COLORING, MINIMUM TSP, MINIMUM BIN PACKING
- The PCP theorem
  - Application: Non-approximability of MAXIMUM 3-SAT

The Gap Technique

- P_1: NPO minimization problem (same for maximization)
- P_2: NP-hard decision problem
- Function f that maps instances x of P_2 into instances $f(x)$ of P_1 such that:
  - If x is a YES-instance, then $m^*(f(x))=c(x)$
  - If x is a NO-instance, then $m^*(f(x)) \geq c(x)(1+g)$
- **Theorem:** No r-approximation algorithm for P_1 exists with $r<(1+g)$ (unless P=NP)

Proof

- A: r-approximation algorithm with $r<(1+g)$
- If x is a YES-instance, then $m^*(f(x))=c(x)$. Hence, $m(f(x),\mathsf{A}(f(x))) \leq rm^*(f(x))=rc(x)<c(x)(1+g)$
- If x is a NO-instance, then $m^*(f(x)) \geq c(x)(1+g)$. Hence, $m(f(x),\mathsf{A}(f(x))) \geq c(x)(1+g)$
- A allows to decide P_2 in polynomial time

Inapproximability of graph coloring

- NP-hard to decide whether a planar graph can be colored with 3 colors
  - Any planar graph is 4-colorable
- $f(G)=G$ where G is a planar graph
  - If G is 3-colorable, then $m^*(f(G))=3$
  - If G is not 3-colorable, then $m^*(f(G))=4=3(1+1/3)$
  - Gap: g=1/3
- **Theorem:** MINIMUM GRAPH COLORING has no r-approximation algorithm with $r<4/3$ (unless P=NP)

Inapproximability of bin packing

- NP-hard to decide whether a set of integers I can be partitioned into two equal sets
- $f(I)=(I,B)$ where B is equal to half the total sum
  - If I is a YES-instance, then $m^*(f(I))=2$
  - If I is a NO-instance, then $m^*(f(I)) \geq 3=2(1+1/2)$
  - Gap: $g=1/2$
- **Theorem:** MINIMUM BIN PACKING has no r-approximation algorithm with $r<3/2$ (unless P=NP)

MINIMUM TSP

- INSTANCE: Complete graph $G=(V,E)$, weight function on E
- SOLUTION: A tour of all vertices, that is, a permutation π of V
- MEASURE: Cost of the tour, i.e.,
  $\Sigma_{1\leq k \leq |V|-1} w(v_{\pi[k]}, v_{\pi[k+1]})+w(v_{\pi[|V|]}, v_{\pi[1]})$

Inapproximability of TSP

- NP-hard to decide whether a graph contains an Hamiltonian circuit
- For any $g>0$, $f(G=(V,E))=(G'=(V,V^2),w)$ where $w(u,v)=1$ if (u,v) is in E, otherwise $w(u,v)=1+|V|g$
  - If G has an Hamiltonian circuit, then $m^*(f(G))=|V|$
  - If G has no Hamiltonian circuit, then
    $m^*(f(G)) \geq |V|-1+1+|V|g=|V|(1+g)$
  - Gap: any $g>0$
- **Theorem:** MINIMUM TSP has no r-approximation algorithm with r>1 (unless P=NP)

The NPO world (unless P=NP)

NPO	MINIMUM TSP
APX	MINIMUM BIN PACKING MAXIMUM SAT MINIMUM VERTEX COVER MAXIMUM CUT
PTAS	MINIMUM PARTITION
PO	MINIMUM PATH

MINIMUM GRAPH COLORING? Certainly not in PTAS

Input-Dependent and Asymptotic Approximation

Summary

- Approximation algorithm for set cover
- Asymptotic approximation scheme for edge coloring

MINIMUM SET COVER

- INSTANCE: Collection C of subsets of a finite set S
- SOLUTION: A set cover for S, i.e., a subset C' of C such that every element in S belongs to at least one member of C'
- MEASURE: $|C'|$

Johnson's algorithm

- Polynomial-time logarithmic approximation algorithm for MINIMUM SET COVER

```
begin
    U:=S; C':=ø;
    for any c_i do c'_i := c_i ;
    repeat
        i:=index of c' with maximum cardinality;
        insert c_i in C';
        U := U-{elements of c'_i };
        delete all elements of c_i from all c';
    until U:=ø
end.
```


MINIMUM EDGE COLORING

- INSTANCE: Graph $G=(V,E)$
- SOLUTION: A coloring of E, that is, function f such that, for any pair of edges e_1 and e_2 that share a common endpoint, $f(e_1) \neq f(e_2)$
- MEASURE: Number of colors, i.e., cardinality of the range of f

Vizing's algorithm

- Polynomial-time algorithm to color a graph with at most D+1 colors, where D denotes the maximum degree of the graph

```
begin
    D:=maximum degree of G;
    G':=(V, E':=ø); // G' is clearly colorable with D+1 colors
    repeat
        add an edge (u,v) of E to E';
        extend coloring of G' without (u,v)  into coloring of G'
        with at most D+1 colors;
        E := E-{(u,v)};
    until E:=ø
end.
```


Asymptotic approximation scheme

- The algorithm returns an edge-coloring with at most D+1 colors
- The optimum is at least D
- Hence, performance ratio is at most
  $(D+1)/m^*(G) \leq D/D+1/m^*(G)=1+ 1/m^*(G)$
  - It implies a 2-approximation

Class F-APX

- Let F be a class of functions

- The class F -APX contains all NPO problems P that admit a polynomial-time algorithm A such that, for any instance x of P, $R(x, A(x)) \le f(|x|)$, for a given function $f \in$ F

- P is said to be $f(n)$-approximable
- A is said to be an $f(n)$-approximation algorithm

Class APTAS

- The class APTAS contains all NPO problems P that admit a polynomial-time algorithm A and a constant k such that, for any instance x of P and for any rational r, $R(x, A(x,r)) \le r + k/m^*(x)$

- The time complexity of A is polynomial in $|x|$ but not necessarily in $1/(r-1)$
- A is said to be an *asymptotic approximation scheme*
  - A is clearly a $(r+k)$-approximation algorithm

The NPO world

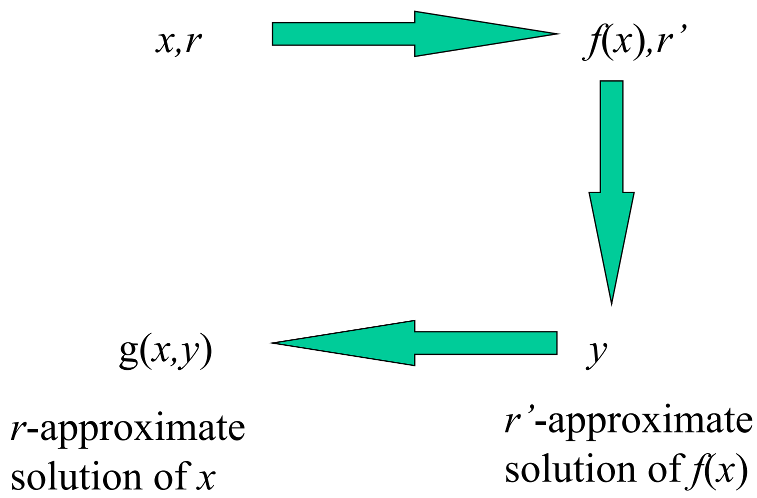
NPO	
$O(n)$-APX	MINIMUM GRAPH COLORING
$O(\log n)$-APX	MINIMUM SET COVER
APX	MAXIMUM SAT MINIMUM VERTEX COVER MAXIMUM CUT
APTAS	MINIMUM EDGE COLORING
PTAS	MINIMUM PARTITION
PO	MINIMUM PATH

Approximation Preserving Reductions

Summary

- AP-reducibility
  - L-reduction technique
- Examples: MAXIMUM CLIQUE, MAXIMUM INDEPENDENT SET, MAXIMUM 2-SAT, MAXIMUM NAE 3-SAT, MAXIMUM SAT(B)

Reducibility and NPO problems



AP-reducibility

- P_1 is AP-reducible to P_2 if two functions f and g and a constant $c \geq 1$ exist such that:
  - For any instance x of P_1 and for any r, $f(x,r)$ is an instance of P_2
  - For any instance x of P_1, for any r, and for any solution y of $f(x,r)$, $g(x,y,r)$ is a solution of x
  - For any fixed r, f and g are computable in polynomial time
  - For any instance x of P_1, for any r, and for any solution y of $f(x,r)$, if $R(f(x,r),y) \leq r$, then $R(x,g(x,y,r)) \leq 1+c(r-1)$

Basic properties

- **Theorem:** If P_1 is AP-reducible to P_2 and P_2 is in APX, then P_1 is in APX
  - If A is an r-approximation algorithm for P_2 then
    $g(x,\mathsf{A}(f(x,r)),r)$
    is a $(1+c(r-1))$-approximation algorithm for P_1
- **Theorem:** If P_1 is AP-reducible to P_2 and P_2 is in PTAS, then P_1 is in PTAS
  - If A is a polynomial-time approximation scheme for P_2 then
    $g(x,\mathsf{A}(f(x,r'),r'),r')$
    is a polynomial-time approximation scheme for P_1, where $r'=1+(r-1)/c$

Basic properties

- **Theorem:** If P_1 is AP-reducible to P_2 and P_2 is in APX, then P_1 is in APX
  - If A is an r-approximation algorithm for P_2 then
    $g(x,A(f(x,r)),r)$
    is a $(1+c(r-1))$-approximation algorithm for P_1
- **Theorem:** If P_1 is AP-reducible to P_2 and P_2 is in PTAS, then P_1 is in PTAS
  - If A is a polynomial-time approximation scheme for P_2 then
    $g(x,A(f(x,r'),r'),r')$
    is a polynomial-time approximation scheme for P_1, where $r'=1+(r-1)/c$

L-reducibility

- P_1 is L-reducible to P_2 if two functions f and g and two constants a and b exist such that:
  - For any instance x of P_1, $f(x)$ is an instance of P_2
  - For any instance x of P_1, and for any solution y of $f(x)$, $g(x,y)$ is a solution of x
  - f and g are computable in polynomial time
  - For any instance x of P_1, $m^*(f(x)) \leq am^*(x)$
  - For any instance x of P_1 and for any solution y of $f(x)$, $|m^*(x)-m(x,g(x,y))| \leq b|m^*(f(x))-m(f(x),y)|$

Basic property of L-reductions

- **Theorem:** If P_1 is L-reducible to P_2 and P_2 is in PTAS, then P_1 is in PTAS
  - Relative error in P_1 is bounded by ab times the relative error in P_2
- However, in general, it is not true that if P_1 is L-reducible to P_2 and P_2 is in APX, then P_1 is in APX
  - The problem is that the relation between r and r' may be non-invertible

Inapproximability of clique

- **Theorem:** MAXIMUM 3-SAT is L-reducible to MAXIMUM CLIQUE
  - $f(C,U)$ is the graph $G(V,E)$ where $V=\{(l,c) : l$ is in clause $c\}$ and $E=\{((l,c),(l',c')) : l \neq l'$ and $c \neq c'\}$
  - $g(C,U,V')$ is a truth-assignment t such that $t(u)$ is true if and only if a clause c exists for which (u,c) is in V'
  - $a=b=1$
    - t satisfies at least $|V'|$ clauses
    - optimum measures are equal
- **Corollary:** MAXIMUM CLIQUE does not belong to APX

Inapproximability of independent set

- **Theorem:** MAXIMUM CLIQUE is AP-reducible to MAXIMUM INDEPENDENT SET
  - $f(G=(V,E)) = G^c=(V,V^2-E)$, which is called the complement graph
  - $g(G,U)=U$
  - $c=1$
    - Each clique in G is an independent set in G^c
- **Corollary:** MAXIMUM INDEPENDENT SET does not belong to APX

Inapproximability of 2-satisfiability

- **Theorem:** MAXIMUM 3-SAT is L-reducible to MAXIMUM 2-SAT
  - f transforms each clause x **or** y **or** z into the following set of 10 clauses where i is a new variable:
    - x, y, z, i, **not** x **or not** y, **not** x **or not** z, **not** y **or not** z, x **or not** i, y **or not** i, z **or not** i
  - $g(C,t)$=restriction of t to original variables
  - $a=13$, $b=1$
    - $m^*(f(x))=6|C|+m^*(x) \leq 12m^*(x)+m^*(x)=13m^*(x)$
    - $m^*(f(x))-m(f(x),t) \leq m^*(x)-m(x,g(C,t))$
- **Corollary:** MAXIMUM 2-SAT is not in PTAS

MAXIMUM NOT-ALL-EQUAL SAT

- INSTANCE: CNF Boolean formula, that is, set C of clauses over set of variables V
- SOLUTION: A truth-assignment f to V
- MEASURE: Number of clauses that contain both a false and a true literal

Inapproximability of NAE 2-satisfiability

- **Theorem:** MAXIMUM 2-SAT is L-reducible to MAXIMUM NAE 3-SAT
  - f transforms each clause x **or** y into new clause x **or** y **or** z where z is a new global variable
  - $g(C,t)$=restriction of t to original variables
  - $a=1$, $b=1$
    - z may be assumed false
    - each new clause is not-all-equal satisfied iff the original clause is satisfied
- **Corollary:** MAXIMUM NAE 3-SAT is not in PTAS

Inapproximability of MAXIMUM SAT(B)

- Standard reduction
  - If a variable y occurs h times, create new h variables $y[i]$
  - Substitute ith occurrence with $y[i]$
  - Add (**not** $y[i]$ **or** $y[i+1]$) and (**not** $y[h]$ **or** $y[1]$)
- Not useful: deleting one new clause may increase the measure arbitrarily
  - The cycle of implications can be easily broken
  - If we add all possible implications (that is, we use a clique), then no the number of occurrences is not bounded and there is no linear relation between optimal measures

Expander graphs

- A graph $G=(V,E)$ is an expander if, for every subset S of the nodes, the corresponding cut has measure at least

$$\min\{|S|, |V-S|\}$$

  - A cycle is not an expander
  - A clique is an expander (but has unbounded degree)
- **Theorem:** A constant n_0 and an algorithm A exist such that, for any $k > n_0$, A(k) constructs in time polynomial in k a 14-regular expander graph E_k with k nodes.

AP-reduction through expanders

- We may assume that h is greater than n_0 (it suffices to replicate any clause n_0 times)
- For any i and j, if (i,j) is an edge of E_h then add (**not** $y[i]$ **or** $y[j]$) and (**not** $y[j]$ **or** $y[i]$)
- Globally, we have $m + 14N$ clauses where N is the sum of the hs
- Each variable occurs in exactly 28 new clauses and 1 old clause: hence, $B=29$
  - Starting from $B=29$, it is possible to arrive at $B=3$

Proof

- **Claim:** Any solution must satisfy all new clauses (that is, gives the same value to all copies of the same variable)
  - From the expansion property, if we change the truth value of the copies in the smaller set we do not loose anything
- $a=85$
  - $m^*(f(x)) = 14N+m^*(x) \geq 42m+m^*(x) \geq 85m^*(x)$
- $b=1$
  - $m^*(x)-m(x,t)=14N+m^*(f(x))-14N-m(f(x),t)=m^*(f(x))-m(f(x),t)$

Other inapproximability results

- **Theorem:** MINIMUM VERTEX COVER is not in PTAS
  - Reduction from MAXIMUM 3-SAT(3)
- **Theorem:** MAXIMUM CUT is not in PTAS
  - Reduction from MAXIMUM NAE 3-SAT
- **Theorem:** MINIMUM GRAPH COLORING is not in APX
  - Reduction from variation of independent set

The NPO world if P ≠ NP

NPO Poly-APX	MINIMUM TSP MAXIMUM INDEPENDENT SET MAXIMUM CLIQUE MINIMUM GRAPH COLORING
APX	MINIMUM BIN PACKING MAXIMUM SATISFIABILITY MINIMUM VERTEX COVER MAXIMUM CUT
PTAS	MINIMUM PARTITION
PO	MINIMUM PATH

Approximation Algorithms

- Example approximation algorithm:
  – Recall:

  Vertex Cover (VC): given a graph G, what is the *smallest* subset of vertices that touch every edge?

  – **NP**-complete

Approximation Algorithms

- Approximation algorithm for VC:
  – pick an edge (x, y), add vertices x and y to VC
  – discard edges incident to x or y; repeat.
- Claim: approximation ratio is 2.
- Proof:
  – an optimal VC must include at least one endpoint of each edge considered
  – therefore 2*opt ≥ actual

Approximation Algorithms

- diverse array of ratios achievable
- some examples:
  - (min) Vertex Cover: 2
  - MAX-3-SAT (find assignment satisfying largest # clauses): 8/7
  - (min) Set Cover: ln n
  - (max) Clique: $n/\log^2 n$
  - (max) Knapsack: $(1 + \varepsilon)$ for any $\varepsilon > 0$

Approximation Algorithms

(max) Knapsack: $(1 + \varepsilon)$ for any $\varepsilon > 0$

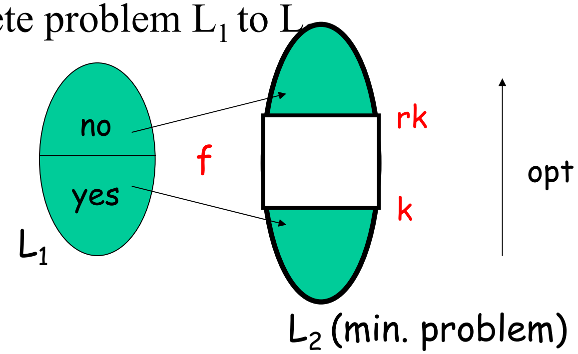
- called Polynomial Time Approximation Scheme (PTAS)
  - algorithm runs in poly time for every fixed $\varepsilon>0$
  - poor dependence on ε allowed
- If all **NP** optimization problems had a PTAS, almost like **P** = **NP** (!)

Approximation Algorithms

- A job for complexity: How to explain failure to do better than ratios on previous slide?
  - just like: how to explain failure to find poly-time algorithm for SAT...
  - first guess: probably NP-hard
  - what is needed to show this?
- "gap-producing" reduction from **NP**-complete problem L_1 to L_2

Approximation Algorithms

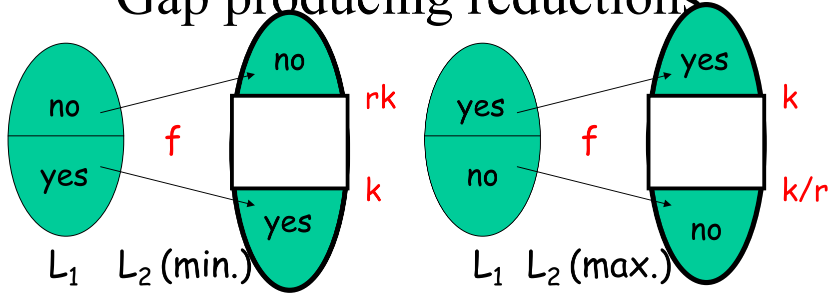
- "gap-producing" reduction from **NP**-complete problem L_1 to L_2



Gap producing reductions

- r-gap-producing reduction:
  - f computable in poly time
  - $x \in L_1 \Rightarrow opt(f(x)) \leq k$
  - $x \notin L_1 \Rightarrow opt(f(x)) > rk$
  - for max. problems use "$\geq k$" and "$< k/r$"
- Note: target problem is not a language
  - **promise problem** (yes $\cup$ no *not* all strings)
  - "promise": instances always from (yes $\cup$ no)

Gap producing reductions

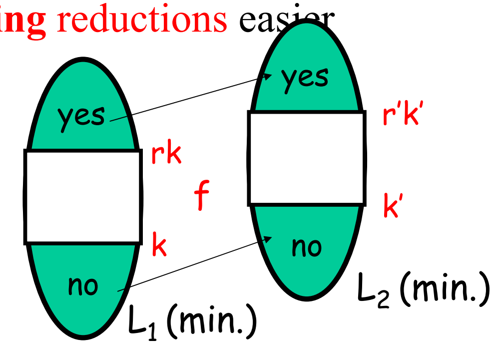


- Main purpose:
  - r-approximation algorithm for L_2 distinguishes between f(yes) and f(no); can use to decide L_1
  - "**NP**-hard to approximate to within r"

Gap preserving reductions

- gap-producing reduction difficult (more later)
- but **gap-preserving** reductions easier

Warning: many reductions **not** gap-preserving



Gap preserving reductions

- Example **gap-preserving** reduction:
  - reduce MAX-k-SAT with gap ε
  - to MAX-3-SAT with gap ε'
  - "MAX-k-SAT is NP-hard to approx. within $\varepsilon \Rightarrow$ MAX-3-SAT is NP-hard to approx. within ε'"

  constants
- **MAXSNP** (PY) – a class of problems reducible to each other in this way
  - PTAS for **MAXSNP**-complete problem iff PTAS for all problems in **MAXSNP**

MAX-k-SAT

- Missing link: first gap-producing reduction
  - history's guide

    it should have something to do with SAT
- Definition: MAX-k-SAT with gap ε
  - instance: k-CNF φ
  - YES: some assignment satisfies **all** clauses
  - NO: no assignment satisfies more than $(1 - \varepsilon)$ fraction of clauses

Proof systems viewpoint

- k-SAT **NP**-hard $\Rightarrow$ for any language $L \in$ **NP** proof system of form:
  - given x, compute reduction to k-SAT: φ_x
  - expected proof is satisfying assignment for φ_x
  - verifier picks random clause ("local test") and checks that it is satisfied by the assignment

$$x \in L \Rightarrow \Pr[\text{verifier accepts}] = 1$$
$$x \notin L \Rightarrow \Pr[\text{verifier accepts}] < 1$$

Proof systems viewpoint

- MAX-k-SAT with gap ε **NP**-hard $\Rightarrow$ for any language $L \in$ **NP** proof system of form:
  - given x, compute reduction to MAX-k-SAT: φ_x
  - expected proof is satisfying assignment for φ_x
  - verifier picks random clause ("local test") and checks that it is satisfied by the assignment

$$x \in L \Rightarrow \Pr[\text{verifier accepts}] = 1$$
$$x \notin L \Rightarrow \Pr[\text{verifier accepts}] \le (1 - \varepsilon)$$

  - can repeat $O(1/\varepsilon)$ times for error < ½

Proof systems viewpoint

- can think of reduction showing k-SAT NP-hard as designing a proof system for **NP** in which:
  - verifier only performs local tests

- can think of reduction showing MAX-k-SAT with gap ε NP-hard as designing a proof system for **NP** in which:
  - verifier only performs local tests
  - invalidity of proof* evident all over: "holographic proof" and an ε fraction of tests notice such invalidity

PCP

- Probabilistically Checkable Proof (PCP) permits novel way of verifying proof:
  - pick random local test
  - query proof in specified k locations
  - accept iff passes test

- fancy name for a NP-hardness reduction

PCP

- **PCP[r(n),q(n)]**: set of languages L with p.p.t. verifier V that has (r, q)-restricted access to a string "proof"
  - V tosses O(r(n)) coins
  - V accesses proof in O(q(n)) locations
  - (completeness) $x \in L \Rightarrow \exists$ proof such that
    $\Pr[V(x, \text{proof}) \text{ accepts}] = 1$
  - (soundness) $x \notin L \Rightarrow \forall$ proof*
    $\Pr[V(x, \text{proof*}) \text{ accepts}] \leq \frac{1}{2}$

PCP

- Two observations:
  - **PCP[1, poly n] = NP**
    proof?
  - **PCP[log n, 1] $\subset$ NP**
    proof?

The PCP Theorem (AS, ALMSS):

PCP[log n, 1] = NP.

PCP

Corollary: MAX-k-SAT is **NP**-hard to approximate to within some constant ε.
- using PCP[log n, 1] protocol for, say, VC
- enumerate all $2^{O(\log n)} = \text{poly}(n)$ sets of queries
- construct a k-CNF φ_i for verifier's test on each
  - note: k-CNF since function on only k bits
- "YES" VC instance $\Rightarrow$ all clauses satisfiable
- "NO" VC instance $\Rightarrow$ every assignment fails to satisfy at least ½ of the $\varphi_i \Rightarrow$ fails to satisfy an $\varepsilon = (\frac{1}{2})2^{-k}$ fraction of clauses.