Recall
- approx. scheme \( \forall \varepsilon > 0 \)
  \[ A(I, \varepsilon) \geq (1 - \varepsilon) \text{OPT}(I) \]
for max. problem.

- PTAS - run time poly. in \( n \) for fixed \( \varepsilon \)
- FPTAS - "fully poly. time approx. scheme"
  - run time is poly. in \( n \) and \( \frac{1}{\varepsilon} \)

We've seen PTAS - packing unit squares
  - bin packing.

Today: FPTAS

**Knapsack Problem**

Given objects \( 1 \ldots n \) each with size \( s_i \in \mathbb{N} \) and profit \( p_i \in \mathbb{N} \), given knapsack capacity \( B \)

find \( K \subseteq \{1, \ldots, n\} \), \( \sum s_i \leq B \)

and maximize \( \sum_{i \in K} p_i \)

[decision version] is \text{NP}-complete.

Ex. Show [Subset Sum] reduces to Knapsack - or Partition

A pseudo-polynomial time alg. for Knapsack.

Dynamic Programming:

idea: for each profit sum, find min. size.
subproblems:
Find $S(i, p) = \min_i$ size for subset of items $\{i_1 \cdots i_p\}$
of profit exactly $p$.

\[ p = 1 \cdots n \quad \text{max}_i \pi_i \]

\[ P = \text{upper bound on total profit}. \]

Alg.

- Initialize $A, p = 1 \cdots P$, $S(1, p) = \{ s_i \text{ if } P = P_i \}$, $\infty$ otherwise.
- for $i = 2 \cdots n$
  - for $p = 1 \cdots P$
    - for $i \in \text{set} \quad i \in \text{set} \quad i \in \text{set}$
      - $S'(i, p) = \min \{ S(i-1, p), S(i-1, p-P_i) + s_i \}$
    - return max $p$ such that $S(n, p) \leq B = \max_\text{profit of subset of size } B$.

run time:

\[ O(nP) = O(n^2, \max_i \pi_i) \]

pseudo-poly. time — run time depends on $P_i$'s, not on size of $P_i$'s which is $\log(P_i)$ = # bits.

Some NP-complete problems don't have pseudo-poly. time alg's (unless $P = NP$).

- e.g. TSP — still NP-complete with 0-1 weights ( = ham cycle )

Two notions of "easy" NP-complete problem

1. a pseudo-poly. time algorithm
2. an FPTAS
An FPTAS for knapsack (use above pseudo-poly. time alg.)

Idea: if $p_i$'s are small (few bits)
then pseudo-poly. time alg. is poly. time $O(n^2 \max_i p_i)$

So, round $p_i$'s to have few bits.

Alg: Given $\varepsilon$ (want approx $\ge (1-\varepsilon) \cdot$OPT.)

Given $s_1, \ldots, s_n, p_1, \ldots, p_n, B$.

For each $i$, let $p'_i \le \left\lfloor \frac{p_i}{\varepsilon} \right\rfloor$ to be chosen.

Run dyn prog. alg. on $p'_i$ — run time $O(n^2 \max_i p'_i)$

Suppose result gives set $K(t) \subseteq \{1, \ldots, n\}$

$K(t)$ is feasible i.e. $\sum_{i \in K(t)} s_i \le B$.

We need to analyze $P(K(t)) = \sum_{i \in K(t)} p_i$

Compared to $P(K^*) = \text{OPT}$

$P_i - t < t p'_i \le P_i$

$\sum_{i \in K(t)} p_i \ge \sum_{i \in K(t)} t p'_i \ge \sum_{i \in K^*} t p'_i \ge \sum_{i \in K^*} (P_i - t) = \sum_{i \in K^*} P_i - t |K^*|$

because $K(t)$ is opt for $p'_i$

$\ge \text{OPT} - t |K^*| \ge \text{OPT} - t \cdot n = \text{OPT} \left(1 - \frac{t \cdot n}{\text{OPT}}\right)$

note: $\text{OPT} \ge \max_i p_i$

want $\ge \text{OPT} \left(1 - \varepsilon\right)$

so choose $t$ s.t. $\varepsilon = \frac{t \cdot n}{\max_i p_i}$ i.e. $t = \frac{\varepsilon \cdot \max_i p_i}{n}$
Run-time \( O(n^3 \max p_i) \)

\[ p_i' = \left( \frac{p_i}{\varepsilon} \right) = \left( \frac{n \cdot p_i}{\varepsilon \cdot \max p_i} \right) \leq \frac{n}{\varepsilon} \]

So run-time \( O(n^3 \cdot \frac{1}{\varepsilon}) \) — FPTAS

Best known alg. has run time \( O(n \log \frac{1}{\varepsilon} + \left( \frac{1}{\varepsilon} \right)^4) \).

Idea: separate into large profit items (use above technique) and small (pack afterwards).

We showed for knapsack: pseudo-polynomial time alg. \( \longrightarrow \) FPTAS.

Not known in general.

But converse true in general:

FPTAS \( \Rightarrow \) pseudo-polynomial time alg.

Thm [Garey & Johnson '78]

If a problem has an FPTAS, (some technical assumptions)

then there’s a pseudo-polynomial time alg.
Status of NP-complete problems w.r.t. approximation

Harder
- $O(\log n)$-factor (Set Cover)
- constant factor (e.g., Vertex Cover, Euclidean TSP)
- PTAS (e.g., packing unit squares, bin-packing)

Easier
- FPTAS (e.g. knapsack)

Positive results - give approx. alg.
Negative results - “hardness”

An easy example:

Polynomial 2-approx for general TSP $\Rightarrow$ P = NP

This was on Assignment 1.

There are also reductions that preserve good approximation examples

- Polynomial $k$-approx for Ind. Set $\Rightarrow$ polynomial $k$-approx.
  - for clique - this is easy
- PTAS for Ind. Set $\Rightarrow$ PTAS for Clique - same proof
- PTAS for Ind. Set $\Rightarrow$ PTAS for Max 3-SAT.
  - proof outline in Lecture 15

**Lemma** constant factor approx for Clique $\Rightarrow$ PTAS for clique

**Proof.** Suppose we have an alg. that returns a clique of size $\geq \frac{1}{k} \text{OPT}$

Design a PTAS. Input graph $G, \varepsilon$

Want clique of size $\geq (1 - \varepsilon) \text{OPT}$

Idea: “duplicate” $G$ to leverage the $\frac{1}{k}$-approx.
Status of NP-complete problems w.r.t. approximation

Harder
- $O(\log n)$-factor (Set Cover)
- constant factor (e.g., Vertex Cover, Euclidean TSP)
- PTAS (e.g., packing unit squares, bin-packing)

Easier
- FPTAS (e.g., knapsack)

Positive results - give approx. alg.
Negative results - "hardness"

An easy example:
Poly-time 2-approx for general TSP $\Rightarrow$ P = NP
This was on Assignment 1

There are also reductions that preserve good approximation examples
- Poly-time k-approx for Ind. Set $\Leftrightarrow$ poly time k-approx.
  for clique - this is easy
- PTAS for Ind. Set $\Leftrightarrow$ PTAS for Clique - same proof
- PTAS for Ind. Set $\Rightarrow$ PTAS for Max 3-SAT
  - proof outline in Lecture 15
- Constant factor approx for Clique $\Rightarrow$ PTAS for Clique

Breakthrough result 1992
PTAS for Max 3-SAT $\Rightarrow$ P = NP

Thus none of above possible either (unless P = NP)
Lemma constant factor approx for clique \( \Rightarrow \) PTAS for clique

Proof. Suppose we have an alg. that returns a clique of size \( \geq \frac{1}{2} \text{OPT} \) (same proof will work for \( \frac{1}{k} \))

Design a PTAS. Input graph \( G, \varepsilon \)

Want clique of size \( \geq (1-\varepsilon) \text{OPT} \)

Idea: "duplicate" \( G \) to leverage the \( \frac{1}{2} \)-approx.

Define the \( k \)-th power of \( G = (V, E) \), \( G^k \) as follows:

vertices of \( G^k = k \)-tuples \( (v_1, \ldots, v_k) \) \( v_i \in V \)

so \( n^k \) vertices

edge \( (u_1, \ldots, u_k) \) to \( (v_1, \ldots, v_k) \) iff

\( \forall i \; u_i = v_i \; \text{or} \; (u_i, v_i) \in E \)

e.g.

\[
\begin{array}{c}
\text{G1} \\
\text{2} \\
\text{3}
\end{array}
\]

\( G^2 \) has 9 vertices

\( (1,1), (1,2), (1,3), (2,1), \ldots \)

Is there edge \( ((1,1), (1,2)) \)? \( \text{YES} \)

" " \( ((1,1), (1,3)) \)? \( \text{NO} \)

How many edges?
Claim: \( A \) has a clique of size \( t \) iff \( G^k \) has a clique of size \( t^k \).

Furthermore, from a clique of size \( t^k \) in \( G^k \), we can find a clique of size \( t \) in \( G \).

Thus \( \text{OPT}(G^k) = (\text{OPT}(G))^k \).

Plan for PTAS: run the \( \frac{1}{2} \) approx alg. on \( G^k \) for appropriate value of \( k \).

Our approx. alg. gives \( A(G^k) \geq \frac{1}{2} \text{OPT}(G^k) \) in \( G \) we get a clique of size

\[
\left( A(G^k) \right)^{\frac{1}{k}} \geq \left( \frac{1}{2} \text{OPT}(G^k) \right)^{\frac{1}{k}} = \left( \frac{1}{2} \right)^{\frac{1}{k}} \text{OPT}(G)
\]

We want \( \left( \frac{1}{2} \right)^{\frac{1}{k}} \geq (1 - \varepsilon) \)

\[
2^{\frac{1}{k}} \leq \frac{1}{1 - \varepsilon} \quad \frac{1}{k} \leq \log \frac{1}{1 - \varepsilon} \quad k \geq \frac{1}{\log \frac{1}{1 - \varepsilon}}
\]