APPROXIMATION ALGORITHMS

- NPC decision problems often correspond to hard optimization problems of huge practical impact: we cannot give up on them!

Different approaches: (sol. exacta)

1. Non-polynomial algorithms: sometimes suitable for "small" instances. Interesting sub-category: pseudopolynomial algorithms: complexity polynomial in the values (and NOT the size of their encoding)

EXAMPLE: Using dynamic programming we can obtain an algorithm for SS with running time $O(|S| \cdot t)$ (homework: $|S| \times t$ - boolean table A: $A[i,j] = 1 \Leftrightarrow \exists S' \subseteq \{s_1, \dots s_i\} : \sum_{s \in S'} s = j$, $1 \le i \le |S|$, $0 \le j \le t$. Solution in $A[|S|, t]$. Work out RECURRENCE)

$|S| \cdot t$ is polynomial in the value t but NOT in the length of its encoding $O(\log t)$

2. **Restriction of the instance set**: solve the problem on a subset of instances for which there is a polynomial algorithm

EXAMPLE:

2-CNF-SAT $\in P$

TREE-VERTEX-COVER $\in P$

SMALL-TARGET SUBSET SUM $\in P$

$(t \in O(\log^k(\sum_{s \in S} s)))$

3. **"Intelligent" exhaustive search** (O.R. approach): e.g. **branch-and-bound**: try to get optimal solution while avoiding to examine all feasible solutions (pruning strategies).

Heuristic methods: no upper bound on worst-case running time.

Other heuristics: local-search, tabu-search, simulated annealing: may return nonoptimal solution (local optimum) (→ see OR2)

4. **APPROXIMATION ALGORITHMS**: return a feasible solution whose cost is within a given error margin from the optimal solution in worst-case polynomial time

EXAMPLE: Return, in linear time, a vertex cover of size at most twice the size of the minimum vertex cover

3 vs 4: We want hard guarantees on time and solution quality

(Sometimes a solution obtained through 4. is used as a starting solution for the heuristics of 3.)

Let Π be an optimization problem with positive integer cost function: // problemon di ott.

$\Pi \subseteq \mathcal{I} \times \mathcal{S}$, $\forall i \in \mathcal{I}: \mathcal{S}(i) = \{ s \in \mathcal{S} : i \,\Pi\, s \}$ ⇒ sol. accettabili

$c: \mathcal{S} \to \mathbb{N} - \{0\}$ and $\forall i$

$$s^*(i) = \arg \min_{\max} \{ c(s) : s \in \mathcal{S}(i) \}$$

funzione di costo

sol. ottimo

DEF: Let A_π be an algorithm such that $\forall i \in \mathcal{I} : A_\pi(i) \in \mathcal{S}(i)$. ⇒ always feasible

say that A_π is a $\rho(n)$-approximation algorithm for Π if $\forall i \in \mathcal{I}, |i| = n$:

$$\max \left\{ \frac{c(s^*(i))}{c(A_\pi(i))}, \frac{c(A_\pi(i))}{c(s^*(i))} \right\} \leq \rho(n)$$

approximation ratio

OBSERVATION 1 : The above max is always ≥ 1:
⇒ $\rho(n) \geq 1, \forall n$. Better approximation achieved for $\rho(n)$ closer to 1. ($\forall n : \rho(n) = 1$: EXACT ALGORITHM)

OBSERVATION 2 : For maximization problems the above max is always the first term, and for minimization problems the second. Equivalently :

$$c(A_\pi(i)) \begin{cases} \geq \dfrac{c(s^*(i))}{\rho(n)} & \text{MAX. PROBLEMS (fraction } \rho(n)\text{)} \\ \leq c(s^*(i)) \cdot \rho(n) & \text{MIN. PROBLEMS (factor } \rho(n)\text{)} \end{cases}$$

An approximation algorithm finds feasible solutions of "controlled quality". Also, it provides bounds on the optimal solution

In particular:

$c(S^*(i)) \leq c(A_\pi(i))\,\rho(n)$ MAX.

upper bound on $c(S^*(i))$

$c(S^*(i)) \geq \frac{c(A_\pi(i))}{\rho(n)}$ MIN

lower bound on $c(S^*(i))$

DEF We say that Π is $\rho(n)$-approximable in polynomial time if there exists a $\rho(n)$-approximation algorithm A_Π for Π running in polynomial time.

Hard optimization problems are not all equal w.r.t. approximation: some are $\rho = O(1)$-approximable (in poly. time), some only feature $\rho(n)$ with $\lim_{n\to\infty} \rho(n) = \infty$.

We can often prove that if $P \neq NP$ there cannot exist a poly-time $\rho(n)$-approximation algorithm for some function $\rho(n)$. (INAPPROXIMABILITY RESULTS).

EXAMPLE: If $P \neq NP$ then cannot exist a $\rho(|V|) = \Theta(|V|^{1-\varepsilon})$ poly-time approximation algorithm for MAX-CLIQUE, for any fixed ε

[Arora et al. 1990]

There are also "easier hard" problems which admit very "precise" approximation algorithms.

DEF: An **approximation scheme** for an optimization problem Π is a two-input algorithm

$$A_\pi(i, \varepsilon) : i \in I, \quad \varepsilon > 0$$

such that, for any fixed ε, $A_\pi(i, \varepsilon)$ is a $\rho(n) = (1+\varepsilon)$- approximation algorithm for Π.

DEF. An **Approximation Scheme** is **Polynomial-Time (PTAS)** if, for any fixed ε, $A_\pi(i, \varepsilon)$ runs in time polynomial in $n = |i|$

Observe that under the above definition, running times such as

$$O\left(n^{1/\varepsilon}\right)$$

are polynomial in n (for fixed ε)

This is not completely satisfactory: e.g., if I want to reduce the "error" ε by a factor K:

$$O\left(n^{1/\varepsilon}\right) \rightsquigarrow O\left(n^{1/(\varepsilon/K)}\right) = O\left(\left(n^{1/\varepsilon}\right)^{K}\right)!$$

Time exponential in the improvement

It would be desirable that a constant-factor improvement in quality yields a constant-factor increase in running time:

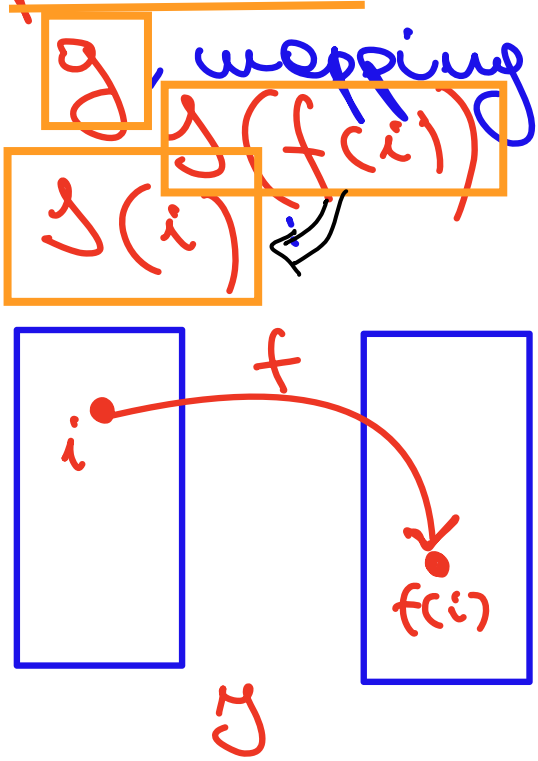
DEF An **Approximation Scheme** is **Fully-Polynomial Time (FPTAS)** if the running time of $A_\pi(i, \varepsilon)$ is polynomial both in $n = |i|$ and $\frac{1}{\varepsilon}$

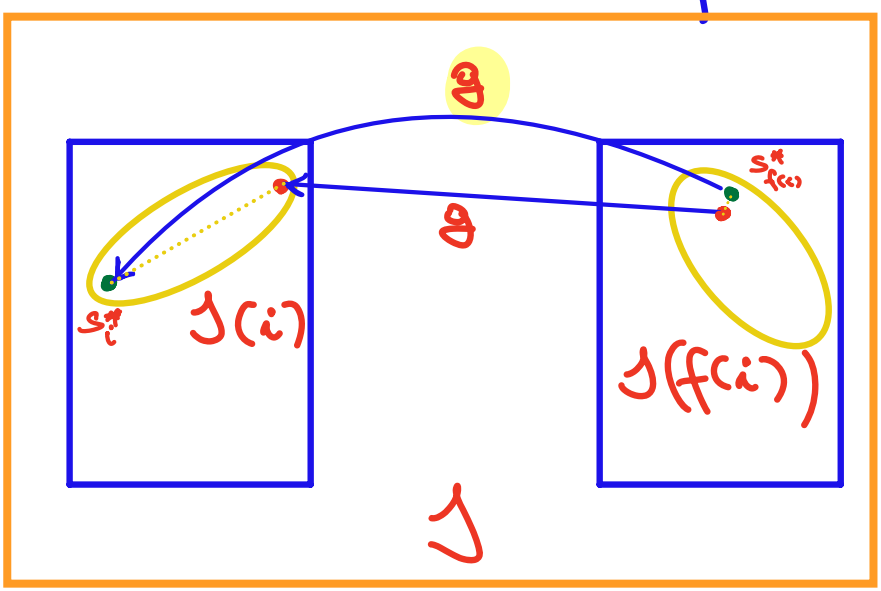
e.g. $T_{A_\pi}(n, \varepsilon) = O\left(\frac{1}{\varepsilon^2} n^3\right)$

Morale

While optimization problems associated to NPC problems are all equivalent w.r.t. exact solution, they behave differently w.r.t. approximation.

Intuition: The idea of reducibility can be extended to optimization problems using an extra function g, mapping each element of $\mathcal{J}(i)$ and $\mathcal{J}(f(i))$ into an element of $\mathcal{J}(i)$.





Function g maps optimal solutions in $\mathcal{J}(f(i))$ into optimal solutions in $\mathcal{J}(i)$, however the mapping does not necessarily preserve the approximation quality!

A TWO-APPROXIMATION ALGORITHM FOR (MINIMUM) VERTEX COVER

Optimization version: given undirected $G=(V,E)$, determine its vertex cover $V^* \subseteq V$ $(\forall \{u,v\} \in E: (u \in V^*) \vee (v \in V^*))$ of minimum size.

spesso, bastano alg. semplici con approcci greedy

We will prove that a simple greedy approach yields a good quality solution

in analisi, collegheremo costo VC a costo di sol. di problema + semplice

IDEA: 1. GREEDY CHOICE: consider an arbitrary edge $\{u,v\}$ and ADD BOTH u and v to the vertex cover. (OBS: NOT INTUITIVE!)

→ e.g. primo in ordine

2. CLEAN-UP: remove all covered edges from E: $E' = E - \{e \in E : \{u,v\} \cap e \neq \emptyset\}$

```
APPROX-VERTEX-COVER (<G = (V,E)>)
  V' ← ∅
  E' ← E
  while (E' ≠ ∅) do
     * select {u,v} ∈ E', arbitrary * GC
     V' ← V' ∪ {u} ∪ {v}
     E' ← E' - {e ∈ E' : {u,v} ∩ e ≠ ∅} c.u.
  return V'
```

⇒ corretto = ritorna sol. feasible

The algorithm correctly returns a vertex cover (I eliminate an edge only when it is covered by some selected node)

Running time: $O(|V|+|E|) = O(|\langle G=(V,E)\rangle|)$
(every time I select edge $\{u,v\}$, I eliminate the two adjacency lists of u and v.

TOUGH PART OF THE ANALYSIS: approximation ratio.

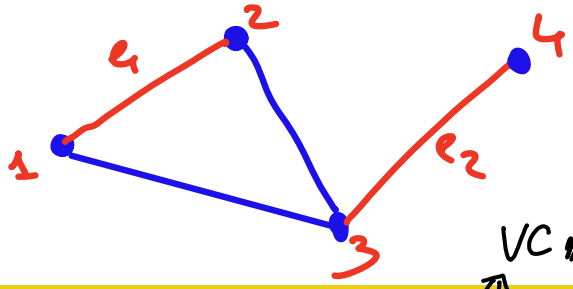
Let A be the set of edges selected by A_V_C (G) at each iteration of the while loop.
↳ accumula tutti lati selezionati

PROPERTY: $\forall e_1 \neq e_2 \in A : e_1 \cap e_2 = \emptyset$
The edges in A do not share endpoints (since at each iteration all edges sharing one endpoint with the selected edge are removed)

A set of edges A not sharing endpoints is called a matching
facile da ottenere in grafo

FACT: If a graph $G=(V,E)$ contains a matching $A \subseteq E$, then for any vertex-cover $V' \subseteq V$ we have $|V'| \geq |A|$

PROOF: There must be at least one endpoint of each edge e in A in V', or otherwise e would not be covered



V' must contain 1 or 2 AND 3 or 4

COROLLARY: $|V^*| \geq |A|$ (VC minimo)

Algorithm A_V_C(G) returns a vertex cover V', with $|V'| = 2|A|$

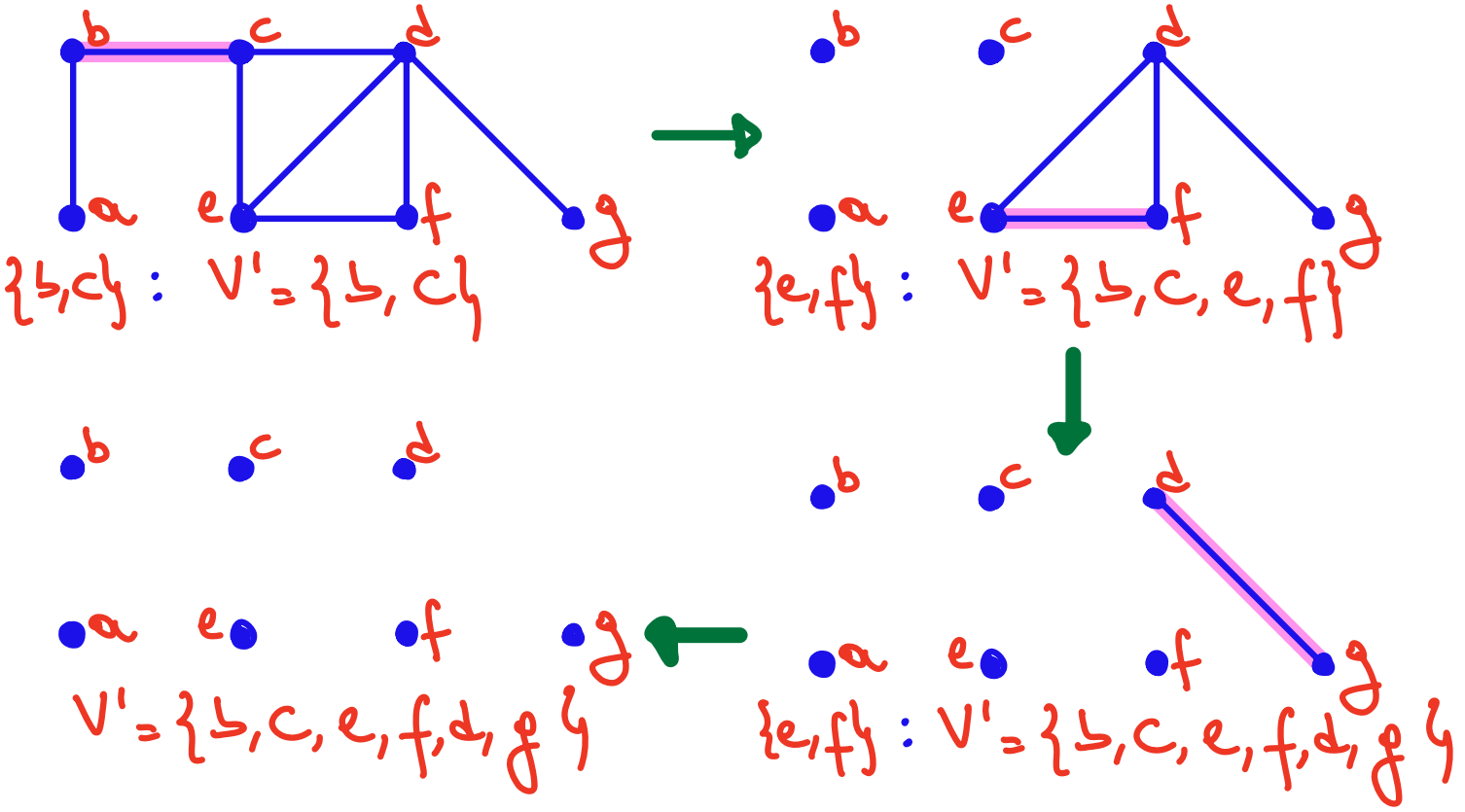
We have $\rho = \frac{|V'|}{|V^*|} \leq \frac{2|A|}{|A|} = 2$

IMPORTANT: To prove an upper bound to ρ, we always need:

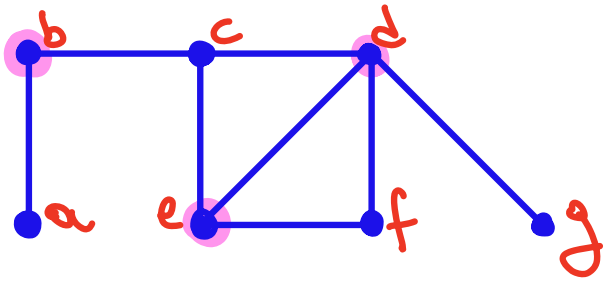
- a lower bound to $c(s^*)$ MINIMIZATION
- an upper bound to $c(s^*)$ MASSIMIZATION

We can show that the bound on the approximation ratio is tight by exhibiting an instance for which

$$c(A_\pi(i)) = \rho \cdot c(S^*(i)) \quad (\text{MIN})$$



$|V'| = 6$ but :



$V^* = \{b, d, e\}$

$\Rightarrow |V'| = 2|V^*|$ (tight approximation!)

LESSONS LEARNED: To get a bound on the approximation we need to establish explicit bounds (lower or upper) on $C(s^*)$.
For VC, we establish such a (lower) bound by relating $|V^*|$ (cannot be computed) to the size of an easily obtainable construct (a matching A of G).
We will see that this approach can be used for other problems.

OBSERVATION: The matching A constructed implicitly by $A\_V\_C(G)$ is maximal, in the sense that $\forall e \in E: A \cup \{e\}$ is not a matching - (PROVE IT!)

EXERCISE: Prove that the set of all nodes forming the edges of a maximal matching are a vertex cover