EXERCISE 4 Show that if $P \neq NP$ there cannot exist an FPTAS for T_TSP

EXERCISE 5 Provide a greedy-based, fast, 2-approximation algorithm for the optimization version of Subset-Sum

Recall: Instance is $\langle S, t \rangle$

$S = \langle x_1, x_2, \ldots, x_n \rangle$, $x_i \leq t$ w.l.o.g.

Determine

$$S^* = \arg\max \{ |S'| : S' \subseteq S, \sum_{s \in S'} s \leq t \}$$

Greedy approach based on a specific ordering of the elements of S: Keep selecting elements in the order while the sum stays $\leq t$.

EXERCISE 6 Consider the following greedy algorithm returning an independent set of a graph $G=(V,E)$:

GREEDY_IS $(G=(V,E))$
$V' \leftarrow \emptyset$
$W \leftarrow V$
while $(W \neq \emptyset)$ do
* select arbitrary $v \in W$ *
$V' \leftarrow V' \cup \{v\}$
$W \leftarrow W - \{v\} - \{u \in W : (u,v) \in E\}$
return V'

Prove that:

1. V' is an IS of G

2. $\forall v \in V : (v \in V') \vee (\exists u \in V' : \{u,v\} \in E)$
(V' is maximAL ($\neq$ maximUM) in the sense that $\forall v \in V - V'$: $V' \cup \{v\}$ is not independent)

3. $\rho_{GIS} \leq \Delta = \max\{\deg(v) : v \in V\}$

EXERCISE 7 Given n jobs $J = \{1, 2, ---, n\}$ we wish to run them on m identical machines. Jobs have durations $t_1, t_2, ---, t_n$. We want to partition the jobs among the m machines so to minimize the maximum execution time of a single machine.

Formally: feasible solution:

schedule $\mathcal{P} = \{M_1, M_2, ---, M_m\}$:

$$M_i \cap M_j = \phi \qquad \bigcup_{k=1}^{m} M_k = \{1, ---, n\}$$

Execution time of M_i: $T_i = \sum_{j \in M_i} t_j$

Objective function:

minimize $T = \max \{T_i : 1 \leq i \leq m\}$

(1) Prove that the decision version of the problem, LOAD BALANCING

$$\begin{cases} I: \langle \{t_1, t_2, ---, t_n\}, m, K \rangle \\ Q: \exists \text{ schedule } \mathcal{P} \text{ with } T \leq K? \end{cases}$$

$\in$ NPH

(2) Let T^* be the cost of the optimal solution. Prove that

$$T^* \geq \max\left\{ \left(\sum_{j=1}^{n} t_j\right)/m,\ \max_{1\leq j\leq n}\{t_j\} \right\}$$

(3) Give a 2-approximation algorithm for optimal LOAD BALANCING

---

(1) We prove

PARTITION $\leq_p$ LOAD BALANCING

$$\begin{cases} I: \langle S \rangle,\ S \subset \mathbb{N}^+,\ \text{finite} \\ Q: \exists\, S_1, S_2 \subset S,\ (S_1 \cup S_2 = S) \wedge (S_1 \cap S_2 = \phi): \\ \qquad \sum_{s\in S_1} s = \sum_{s \in S_2} s\ ? \end{cases}$$

2

$$\boxed{T^{*} \geq \max\left\{\left(\sum_{j=1}^{n} t_j\right)/m,\ \max_{1\leq j\leq n}\{t_j\}\right\}}$$

③ Greedy algorithm : allocate next job to the less loaded machine :