Problem: Subset sum (optimization version)

Input: $L = [x_1, \ldots, x_n]$, a list of positive integers and some $t \in \mathbb{Z}_+$

Output: A sublist $L' = [x_{i_1}, x_{i_2}, \ldots, x_{i_m}]$ ← Not necessarily consecutive

such that

$$t - \sum_{i=1}^{m} x_{i_i} \text{ is nonnegative \& minimized}$$

That is: sum of elements in L' as close to t as possible (but does not exceed t)

Theorem (original Karp problem)

Subset sum (the decision version, Yes/No $\Sigma L' = t$) is NP-hard

NP-hardness omitted here – from vertex cover, SAT, partition, etc... Look up if interested!

Given this algorithm:

```
approx-SS(L,t)
1. sort L descending order
2. Q ← empty list
3. for a ∈ L :
4.     if ΣQ + a ≤ t :
           append a to Q
5. Return Q
```

Show that approx_SS is a 2-approximation.

Proof

Fix input L & t.

Let OPT denote optimal solution and ALG the output of approx_SS(L,t).

To show 2-approx we need to show

$$\frac{1}{2}\Sigma OPT \leq \underbrace{\Sigma ALG \leq 2\Sigma OPT}$$

$\Sigma ALG \leq \Sigma OPT \leq t$ so this is definitely true

We focus on $\frac{1}{2} \Sigma OPT \le \Sigma ALG$.

For simplicity, say L is already sorted

i.e. $L = [x_1, x_2, \ldots, x_n]$ s.t.

$$x_1 \ge x_2 \ge \cdots \ge x_n$$

Case 1: approx_SS(L,t) adds no element to Q at all i.e. ALG = [] empty list.

Can only happen if $x_i > t$ for all i.

But then $\Sigma OPT = 0$ as well and

$$\Sigma ALG = \Sigma OPT \ge \frac{1}{2} \Sigma OPT.$$

Case 2: approx_SS(L,t) adds some, but not all elements. Find i & j s.t. $1 \le i \le j \le n$ and

$$L = [x_1 \dots \underbrace{x_i}_{\substack{\uparrow \\ \text{first elt} \\ \text{in ALG}}} \overbrace{\dots \, x_j}^{\text{all in ALG}} \overbrace{x_{j+1}}^{\text{not in ALG}} \dots x_n]$$

We know:

- $x_1, x_2, \dots, x_{i-1}$ are all $> t$ (otherwise would be added)

Thus none of these in OPT either.

Case 2a

If $j = n$ then $\sum OPT = \sum_{k=i}^{n} x_k = \sum ALG$ must hold.

Case 2b

If $j < n$ then, by alg, we have

$$x_i + x_{i+1} + \dots + x_j \le t$$

$$\underbrace{x_i + x_{i+1} + \dots + x_j}_{} + \underbrace{x_{j+1}}_{} > t$$

one of these must be at least $\frac{t}{2}$.

If $x_{j+1} > \frac{t}{2}$ then $x_j \geq x_{j+1} \geq \frac{t}{2}$ imply $x_i + \ldots + x_j \geq \frac{t}{2}$. So, either way, we have:

$$x_i + \ldots + x_j \geq \frac{t}{2}.$$ Hence

$$\sum ALG \geq x_i + \ldots + x_j \geq \frac{t}{2} \geq \frac{\sum OPT}{2}$$

and we're done. $\square$

Remark

It is quite hard to find examples where $\sum ALG = \frac{1}{2} \sum OPT$, but for some small $\varepsilon > 0$

$$L = \left[\frac{t}{2} + \varepsilon, \frac{t}{2}, \frac{t}{2}\right]$$

will yield $ALG = \left[\frac{t}{2} + \varepsilon\right]$

while $OPT = \left[\frac{t}{2}, \frac{t}{2}\right]$

so eg $t = 100000$

$$L = [50\,001, 50\,000, 50000]$$

Problem Max-cut (optimization)

Input: graph $G = (V, E)$

Output: A maximum cut (S, t) of G

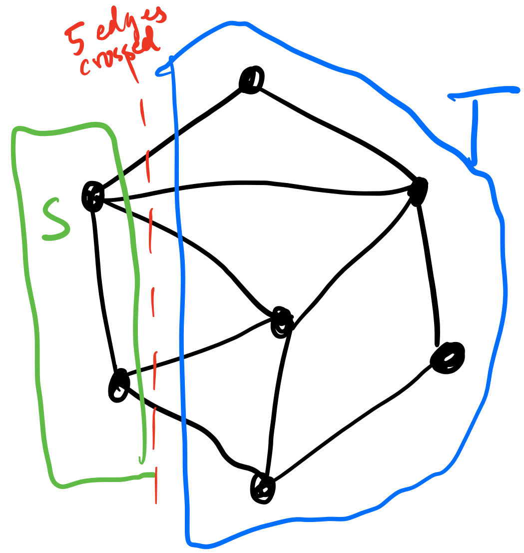
def A cut of $G = (V, E)$ is a pair (S, T) such that $S \subseteq V$ & $T = V \setminus S$.

The size of the cut is

$$|\{\{s,t\} \in E \mid s \in S \text{ \& } t \in T\}|.$$

A maximum cut is a cut of max size.

Ex



cut of size 5

```
approx_cut(G=(V,E))
    S = ∅, T = ∅
    FOR v ∈ V .
        If v has more neighbors in S
              than in T
            T ← T ∪ {v}
        else
            S ← S ∪ {v}
    RETURN (S,T)
```

Show: approx_cut(G) is a 2-approx. of max-cut.

Proof

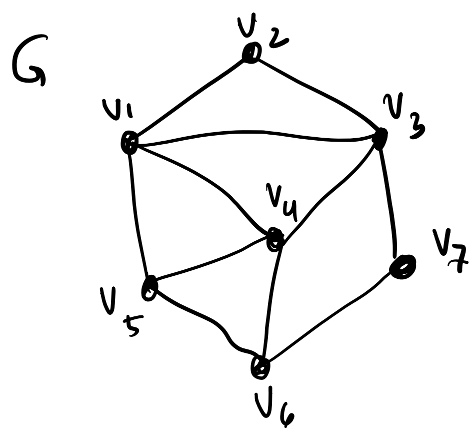
ALG = size of cut by algorithm

OPT = true maximum.

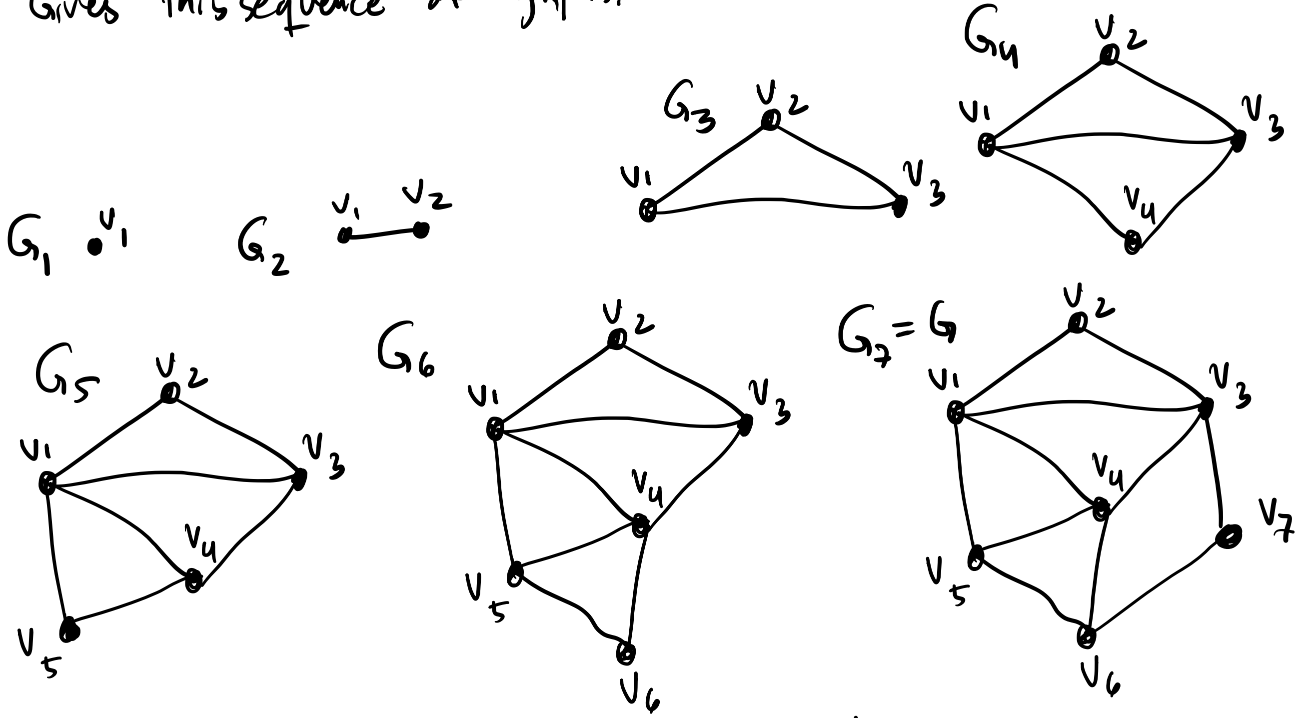
Maximization problem so show $\frac{1}{2}OPT \leq ALG$

Fix an order $v_1, \dots, v_n$ of G's vertices.

Consider induced subgraphs $G_1, \dots, G_n = G$ on these vertices. Example:



Gives this sequence of graphs:



Suppose the FOR-loop goes through the vertices in this order.

Let (S_i, T_i) denote the sets S & T after the i:th iteration, and $|(S_i, T_i)|$ the size of this cut in G_i.

We now show (S_i, T_i) is a 2-approx of max cut in G_i, for every $i = 1, 2, \ldots$

For $i = 1$, G_1 is a single vertex graph so $(S_1, T_1) = (\{v_1\}, \emptyset)$ (v_1 has no neighbors in G_1).

Sps. $|(S_i, T_i)| \geq \frac{1}{2} \underbrace{OPT(G_i)}_{\text{max cut in } G_i}$ for some $i \geq 1$.

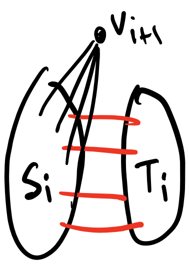
The algorithm ensures that

$$|(S_{i+1}, T_{i+1})| \geq |(S_i, T_i)| + \frac{1}{2} \underbrace{\deg_{G_{i+1}}(v_{i+1})}_{\text{degree of } v_{i+1} \text{ in } G_{i+1}}$$

Apply Induction hypothesis:

$$|(S_i, T_i)| + \frac{1}{2} \deg_{G_{i+1}}(v_{i+1}) \geq \frac{1}{2} OPT(G_i) + \frac{1}{2} \deg_{G_{i+1}}(v_{i+1})$$

$$\geq \frac{1}{2} \Big(\underbrace{OPT(G_i) + \deg_{G_{i+1}}(v_{i+1})}_{\text{best case if all of } v_{i+1}\text{'s neighbours in same set}} \Big) \geq \frac{1}{2} OPT(G_{i+1})$$



$OPT(G_i)$

By induction,

$$ALG = |(S_n, T_n)| \geq \frac{1}{2} OPT(G_n) = \frac{1}{2} OPT$$

$\square$