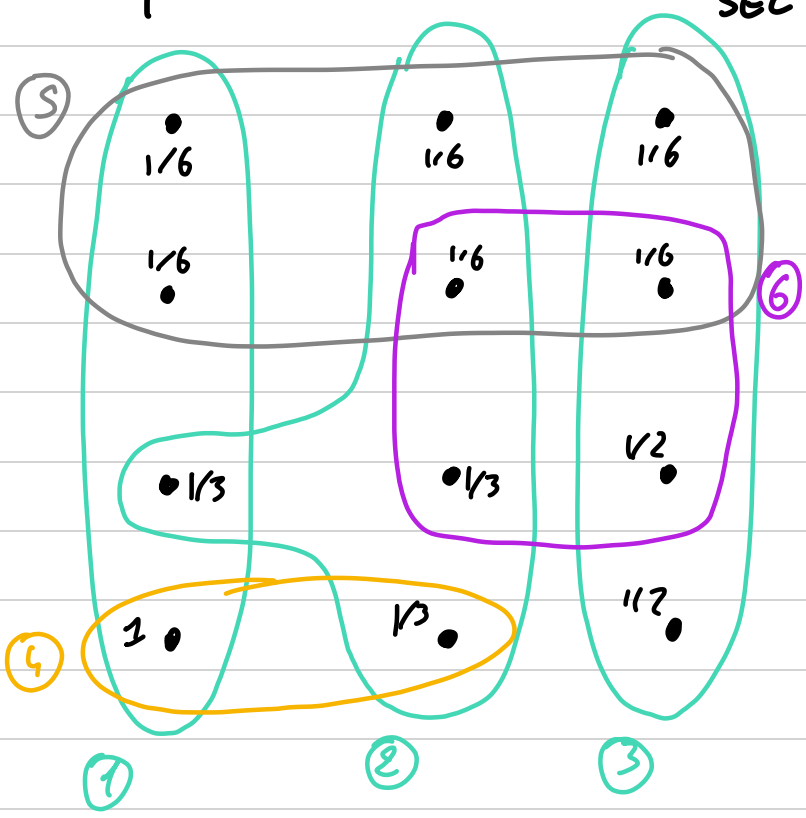
Algorithme d'approximation glouton pour SETCOVER

SETCOVER: Entrée : un ensemble X dit "à couvrir" et un ensemble de parties de X, $\mathcal{F} \subseteq \mathcal{P}(X)$
Sortie : un ensemble de parties de X, $\mathcal{C} \subseteq \mathcal{F}$ tel que $\mathcal{C}$ est une couverture de X, de taille minimale.

Définition 1 : On dit que $\mathcal{C}$ couvre X si $\bigcup_{S \in \mathcal{C}} S = X$

Exemple 2 :



Dans cet exemple une couverture optimale utilise 3 ensembles, les ensembles numérotés 1, 2 et 3

⚠ le problème est NP-complet, il n'y a donc à priori pas d'algo polynomial déterminant une couverture optimale

On considère dans l'algorithme glouton suivant

```
Greedy-SET-COVER(X, F):
    U ← X
    C ← ∅
    while U ≠ ∅
        Choisir S ∈ F qui maximise |S ∩ U|
            U ← U - S
            C ← C ∪ {S}
    retourner C
```

Idée de l'algorithme : U désigne les éléments qui restent à couvrir.
C désigne la couverture courante.
A chaque étape, on choisit l'ensemble qui ajoute le plus d'éléments, i.e celui qui maximise $|S \cap U|$

Il est clair que Greedy-SET-COVER renvoie une couverture C de X.
Soit C^* une couverture optimale de X.

Montrons que $|C| \leq (\ln|X|+1) \cdot |C^*|$

Pour cela, on introduit une valeur v^* vérifiant $|C| \leq v^* \leq (\ln|X|+1)|C^*|$

A chaque pas de l'algorithme, on distribue équitablement 1 point entre les nouveaux éléments de la couverture.

Sur l'exemple 2 : on choisit S, puis S_2, puis S_3, puis (S_1 ou S_4)
la distribution des points est indiquée sur le schéma

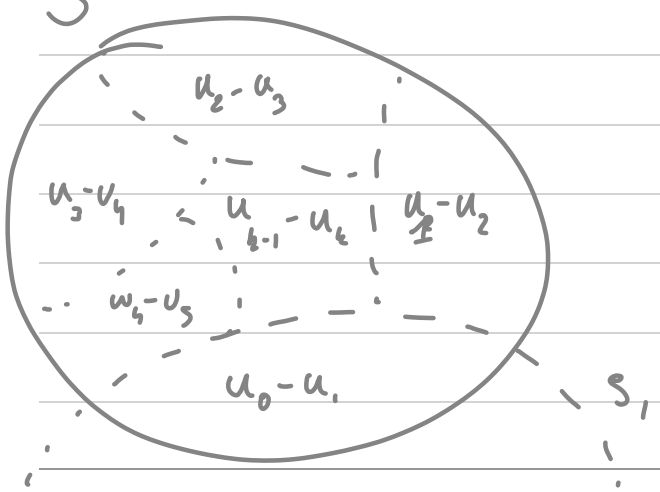
Pour tout $x \in X$, on note c_x le nombre de pts reçus. $c_x = \dfrac{1}{|S_i - (S_1 \cup S_2 \cdots \cup S_{i-1})|}$

On conclut alors $\boxed{v^* = \sum_{S \in C^*} \sum_{x \in S} c_x}$

On a $v^* \geq \sum_{x \in X} c_x = |C|$ (on distribue 1 pt par étape)

Démontrons mtn que pour $S \in \mathcal{F}$, $\sum_{x \in S} c_x \leq H(|S|)$

Soit $S \in \mathcal{F}$.
On pose $u_i = |S - (S_1 \cup S_2 \cup \cdots \cup S_i)|$
u_i est le nb d'éléments non couverts de S à la $i$ème étape



Remarquons d'abord que $|S_i - (S_1 \cup S_2 \cdots \cup S_{i-1})|$
$\geq |S - (S_1 \cup S_2 \cdots \cup S_{i-1})| = u_{i-1}$
car sinon l'algorithme aurait choisi S au lieu de S_i.

On a $$\sum_{x\in S} c_x \leq \sum_{i=1}^{k} (u_{i-1} - u_i) \cdot \frac{1}{|S_i - (S_1 \cup S_2 \cdots \cup S_{i-1})|}$$

$$\leq \sum_{i=1}^{k} (u_{i-1} - u_i) \cdot \frac{1}{u_{i-1}}$$

$$= \sum_{i=1}^{k} \sum_{j=u_i+1}^{u_{i-1}} \frac{1}{u_{i-1}}$$

$$\leq \sum_{i=1}^{k} \sum_{j=u_{i+1}}^{u_{i-1}} \frac{1}{j}$$

$$= \sum_{i=1}^{k} H(u_{i-1}) - H(u_i)$$

$$= H(u_0) - H(u_k)$$
$$\leq H(|S|)$$

On en déduit finalement

$$v^* \leq \sum_{S \in C^*} H(|S|)$$

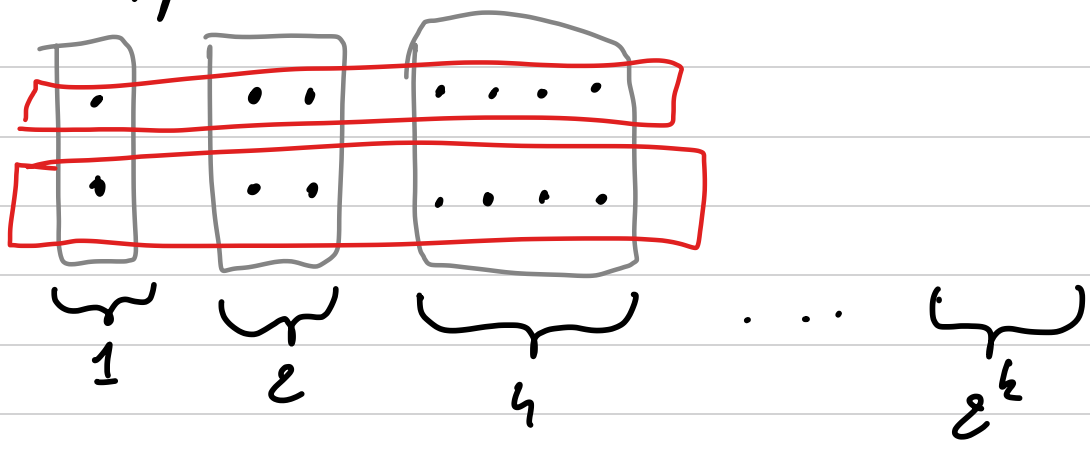
$$\leq |C^*| \cdot H(|X|)$$

$$\leq |C^*| \cdot (\ln |X| + 1)$$ ← série-intégrale

ce qui achève la démonstration.

Le pire cas est atteint

petit exemple



toujours une solution optimale de taille 2.

l'algorithme glouton en trouve une de taille $k+1$, or $n = 2^{k+2} - 2$