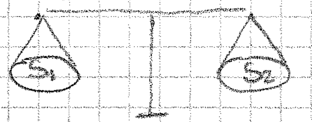
ESERCIZIO:

Date n monete, delle quali una falsa (pesa di meno) fornire un algoritmo che determini la moneta falsa usando una bilancia a due piatti (che può confrontare sottoinsiemi di monete).

Mostrare che ogni algoritmo è $\Omega(\log n)$

idea: Dato l'insieme S di monete, una pesata è un'operaz che prende due sottoinsiemi disgiunti e li confronta

$$S = S_1 \uplus S_2 \uplus S_3 \qquad \text{con } |S_1| = |S_2|$$



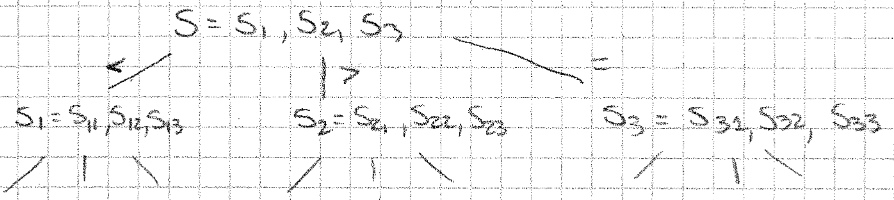
$S_1 < S_2 \rightsquigarrow S_1$

$S_1 > S_2 \rightsquigarrow S_2$

$S_1 = S_2 \rightsquigarrow S_3$

riduce la ricerca ad uno dei tre sottoinsiemi

Un qualunque algoritmo può essere rappresentato mediante un albero



→ in tutti i modi almeno uno dei sottoinsiemi ha cardinalità $\geq$ numero totale di monete $/3$

→ le foglie devono avere cardinalità 1

⇓

→ l'albero ha altezza $\geq \log_3 n$

La complessità (numero di pesate) nel caso pessimo è data dal percorso più lungo dalla radice ad una foglia (dato che variando l'imput posso ottenere ogni cammino, siccome le monete sono indistinguibili!)

$\Downarrow$

$\Omega(\log n)$

Un algoritmo ottimo divide in tre sottoinsiemi uguali.
(anche altre suddivisioni, ad esempio in due sottoinsiemi di $\frac{n}{2}$ più un eventuale resto sono asintoticamente ottime, ma concretamente peggiori)