Disjoint Sets

Disjoint Sets operationer

MAKE-SET(x):

Opret $\{x\}$ som en mængde.

UNION(x, y):

Slå $\{a, b, c, \dots, x\}$ og $\{h, i, j, \dots, y\}$ sammen til $\{a, b, c \dots, h, i, j, \dots, x, y\}$.

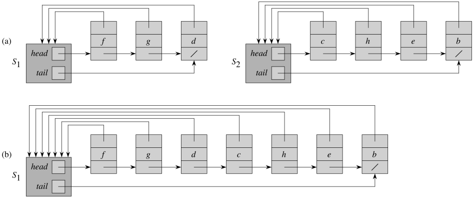
FIND-SET(x):

Returner en ID for mængden indeholdende x.

(NB: Vi har ingen krav til ID'en. Skal blot være den samme for alle x i samme mængde.)

Datastruktur for Disjoint Sets

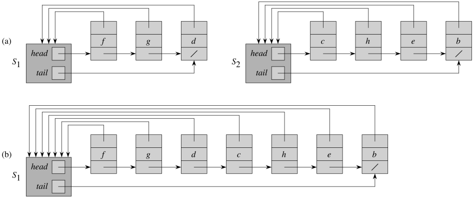
En simpel løsning:



- $\textsc{Make-Set}(x)$: opret ny liste.
- $\textsc{Union}(x, y)$: slå lister sammen, behold een header, ændrer alle header-pointere i den anden liste.
- $\textsc{Find-Set}(x)$: returner pointer til header.

Datastruktur for Disjoint Sets

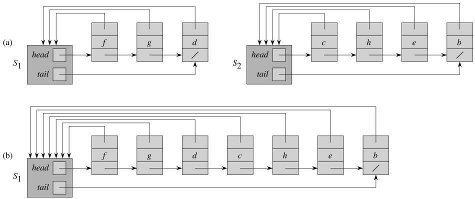
Køretid (efter n MAKE-SET):



- MAKE-SET(x): opret ny liste: $O(1)$.
- UNION(x, y): slå lister sammen, behold een header, ændre alle header-pointere i den anden liste: $O(n)$.
- FIND-SET(x): returner pointer til header: $O(1)$.

Datastruktur for Disjoint Sets

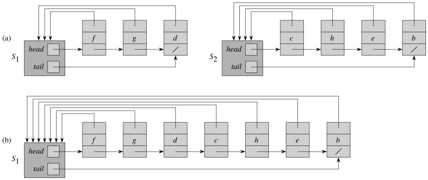
Køretid (efter n MAKE-SET):



- MAKE-SET(x): opret ny liste: $O(1)$.
- UNION(x, y): slå lister sammen, behold een header, ændre alle header-pointere i den anden liste: $O(n)$.
- FIND-SET(x): returner pointer til header: $O(1)$.

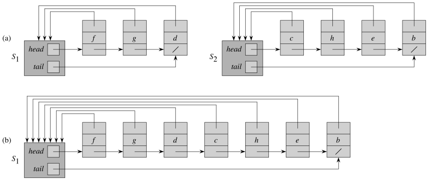
Naiv analyse: n MAKE-SET, op til $n-1$ UNION, og m FIND-SET koster $O(m+n^2)$.

Datastruktur for Disjoint Sets



- $\text{Make-Set}(x)$: opret ny liste: $O(1)$.
- $\text{Union}(x, y)$: slå lister sammen, behold header af længste liste, ændre alle header-pointere i korteste liste: $O(n)$.
- $\text{Find-Set}(x)$: returner pointer til header: $O(1)$.

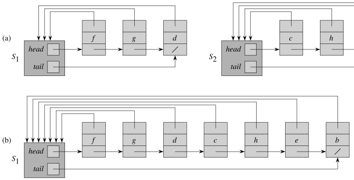
Datastruktur for Disjoint Sets



- $\textsc{Make-Set}(x)$: opret ny liste: $O(1)$.
- $\textsc{Union}(x, y)$: slå lister sammen, behold header af længste liste, ændre alle header-pointere i korteste liste: $O(n)$.
- $\textsc{Find-Set}(x)$: returner pointer til header: $O(1)$.

En knude kan kun ændre sin header-pointer $k = \log n$ gange, da størrelsen af dens mængde hver gang vokser mindst en faktor to ($1 \cdot 2^k \leq n \Leftrightarrow k \leq \log n$).

Datastruktur for Disjoint Sets

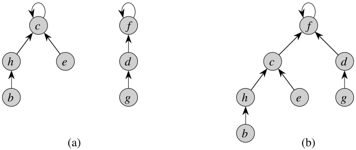


- $\textsc{Make-Set}(x)$: opret ny liste: $O(1)$.
- $\textsc{Union}(x, y)$: slå lister sammen, behold header af længste liste, ændre alle header-pointere i korteste liste: $O(n)$.
- $\textsc{Find-Set}(x)$: returner pointer til header: $O(1)$.

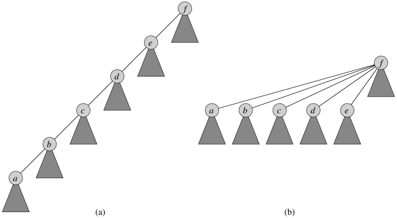
En knude kan kun ændre sin header-pointer $k = \log n$ gange, da størrelsen af dens mængde hver gang vokser mindst en faktor to ($1 \cdot 2^k \leq n \Leftrightarrow k \leq \log n$).

Bedre analyse: n $\textsc{Make-Set}$, op til $n - 1$ $\textsc{Union}$, og m $\textsc{Find-Set}$ koster $O(m + n \log n)$.

En anden datastruktur

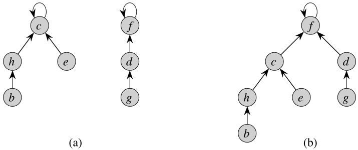


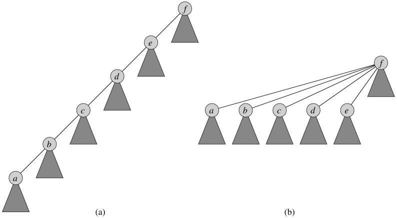
(a) (b)



(a) (b)

En anden datastruktur





Union by rank + path compression $\Rightarrow$ meget tæt på $O(m+n)$ tid. Mere præcist $O(m \cdot \alpha(n))$.

Analyse: i et senere kursus.