Conjuntos Disjuntos, Árboles y Gráficas

Estructuras de datos para mantener conjuntos disjuntos

- Estructura para mantener una colección $S=\{S_1,S_2,\ldots,S_k\}$ de conjuntos dinámicos disjuntos.
- Cada conjunto se identifica por un representante que es algún miembro del conjunto.
- Cada elemento del conjunto se representa con un objeto x que debe soportar las operaciones siguientes:
  - MAKE-SET(x) : nuevo conjunto con único miembro x (representante). x no puede estar en otro conjunto.
  - UNION(x,y) : une dos conjuntos dinámicos que contienen a x (S_x) y a y (S_y) como miembros en un nuevo conjunto. S_x y S_y son destruidos al terminar la operación.
  - FIND-SET(x) : regresa un apuntador al representante del único conjunto que contiene a x.

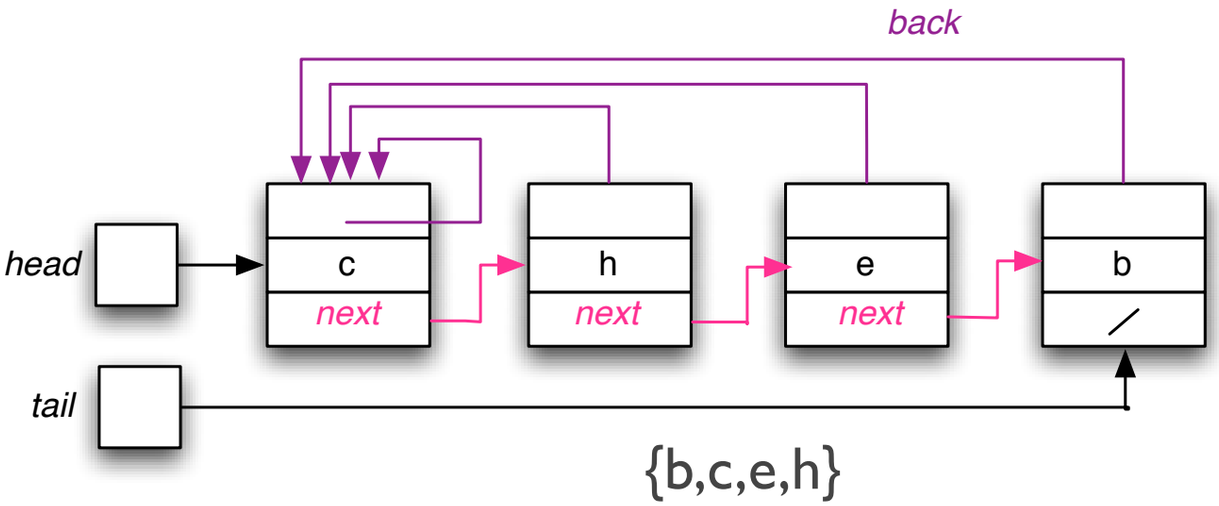
Estructuras de datos para mantener conjuntos disjuntos

- Las representaciones de conjuntos disjuntos se evalúan usando dos parámetros:
  - n : número total de operaciones MAKE-SET.
  - m: número total de operaciones MAKE-SET, UNION y FIND-SET.
- Nótese que cada operación UNION reduce el número de conjuntos disjuntos en uno.
- ¿Cuántos conjuntos disjuntos quedan después de n-1 operaciones UNION?
  - uno.
- ¿Cuál es el número máximo posible de operaciones UNION?
  - n-1.

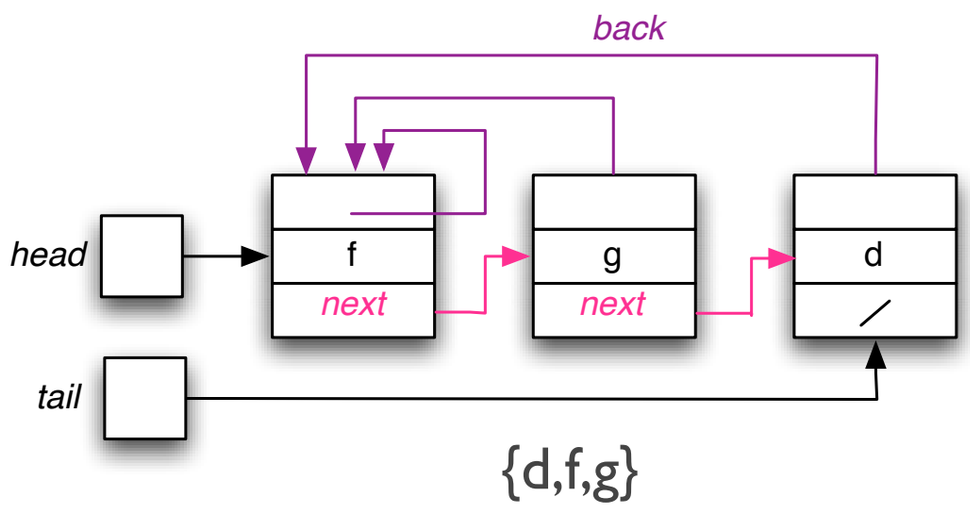
Representación de conjuntos disjuntos con listas ligadas

- El primer elemento de la lista sirve como representante del conjunto.
- Cada objeto en la lista ligada contiene:
  - un miembro del conjunto,
  - un apuntador next al objeto que contiene al siguiente miembro,
  - un apuntador back para regresar al representante.
- Cada lista ligada mantiene:
  - un apuntador head al representante,
  - un apuntador tail al final de la lista.
- Dentro de cada lista los objetos pueden aparecer en cualquier orden.

Representación de conjuntos disjuntos con listas ligadas

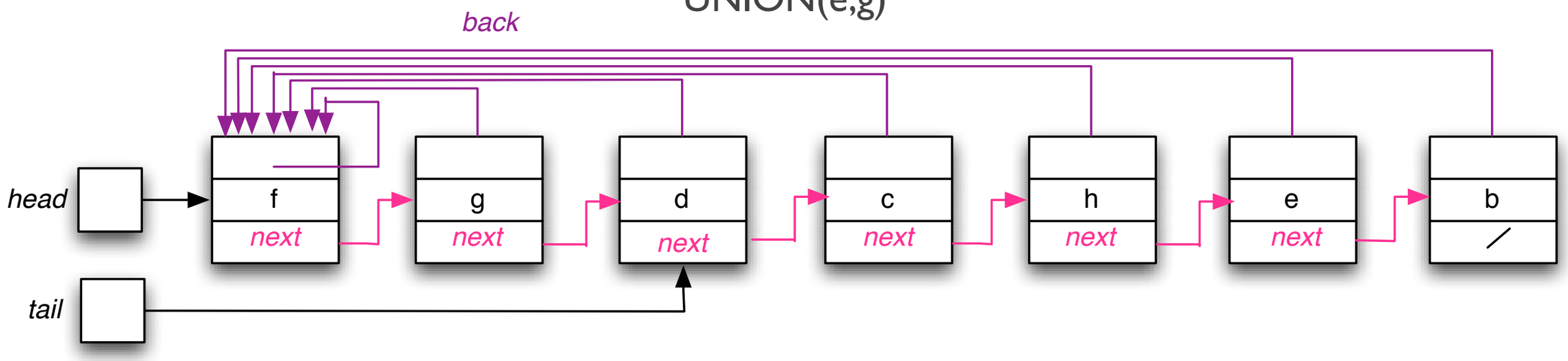


{b,c,e,h}



{d,f,g}

UNION(e,g)



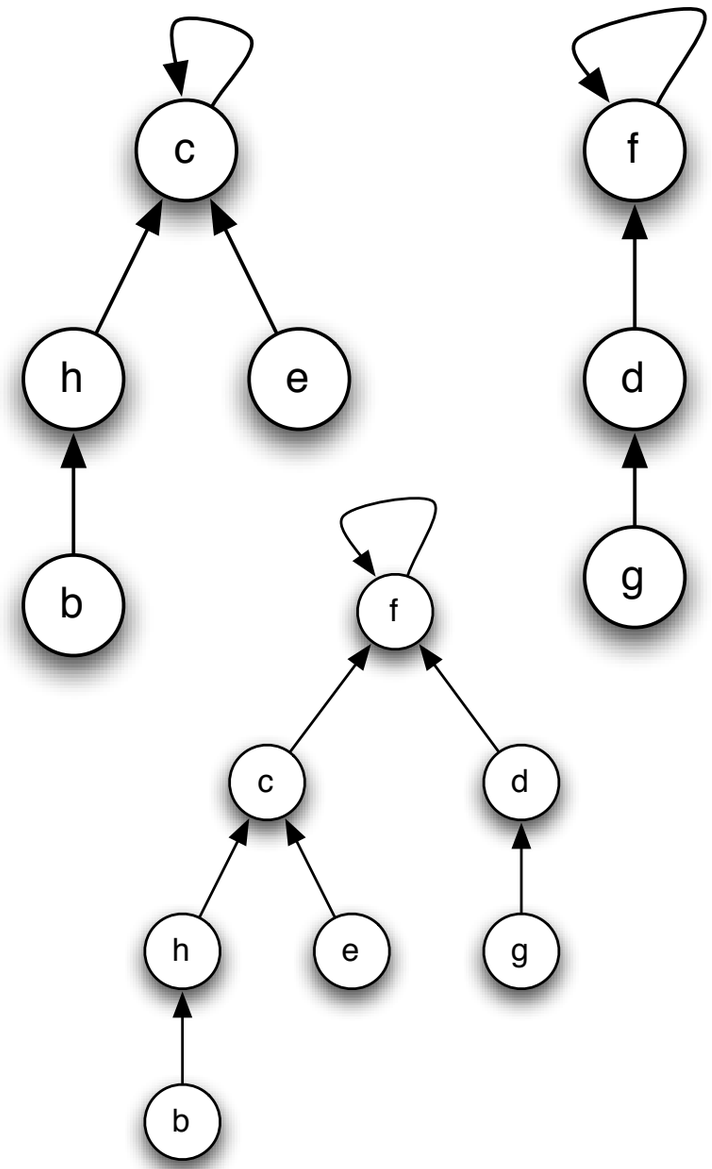
{b,c,d,e,f,g,h}

Representación de conjuntos disjuntos con listas ligadas

- ¿Cuál es el tiempo de ejecución para MAKE-SET(x)?
  - O(1): crear la lista nueva con x como único elemento.
- ¿Y de FIND-SET?
  - O(1): regresar el apuntador back de x a su representante.
- Implementación simple de UNION(x,y), ¿cuánto tiempo toma?
  - O(x): actualizar el apuntador del objeto original al representante de la lista x (que puede ser la más larga).
- Implementación con heurística de peso de UNION.
- Usando la representación de listas ligadas y una heurística de peso para UNION, una secuencia de m operaciones MAKE-SET, UNION, FIND-SET, de las cuáles n son MAKE-SET, toma O(m+n log n) tiempo de ejecución.

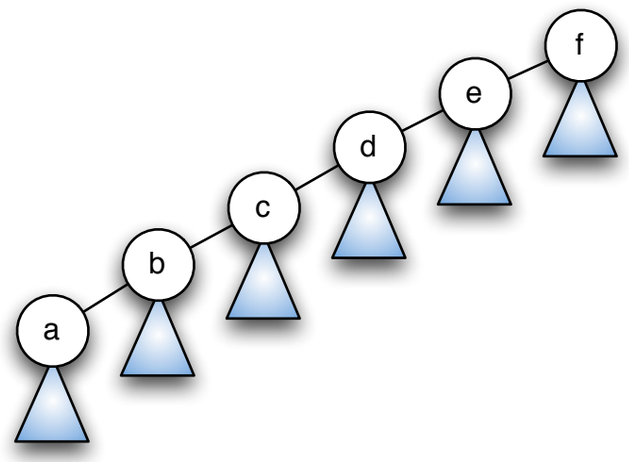
Representación de conjuntos disjuntos con árboles

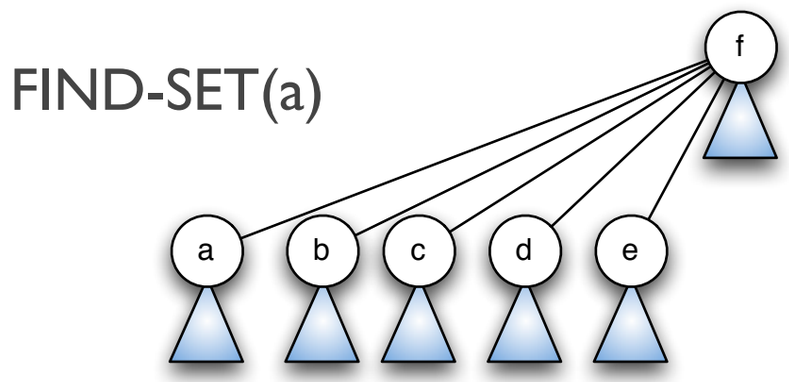
- Árboles enraizados.
- Cada nodo contiene un miembro y cada árbol representa un conjunto.
- Cada miembro apunta a su padre.
- Por si mismos no representan grandes ventajas: se usan dos heurísticas.
- Unión por rango.
- Camino por compresión.
- MAKE-SET crea un árbol nuevo con un sólo nodo.
- FIND-SET sigue los apuntadores a predecesores hasta encontrar la raíz.
- UNION hace la raíz de un árbol apuntar a la raíz de otro.



Heurísticas: unión por rango

- Unión por rango (union by rank)
  - similar a la unión por peso con listas ligadas.
  - hacer al árbol con menos nodos apuntar a la raíz del árbol con más nodos.
  - para cada nodo se mantiene su rango (número de aristas en el camino más largo entre el nodo y sus nodos dependientes).
- Compresión por camino (path compression)
  - usado durante FIND-SET para hacer que cada nodo visitado apunte directamente a la raíz.





Representación de conjuntos disjuntos con árboles

```
MAKE-SET(x)

1  p[x] ← x
2  rank[x] ← 0
```

```
FIND-SET(x)

1  if x ≠ p[x]
2       then p[x] ← FIND-SET(p[x])
3  return p[x]
```

```
UNION(x, y)

1  LINK(FIND-SET(x), FIND-SET(y))
```

```
LINK(x, y)

1  if rank[x] > rank[y]
2       then p[y] ← x
3       else p[x] ← y
4            if rank[x] = rank[y]
5                 then rank[y] ← rank[y] + 1
```

Representación de conjuntos disjuntos con árboles

- La operación UNION con una heurística de unión por rango tiene un tiempo de ejecución de $O(m \log n)$.
- La operación FIND-SET con una heurística de compresión de caminos tiene un tiempo de ejecución de $\Theta(n+f(1+\log n))$.
  - para n operaciones MAKE-SET, n-1 operaciones UNION y f operaciones FIND-SET.
- Cuando ámbas heurísticas son usadas, el tiempo de ejecución en el peor caso es $O(m\alpha(n))$, donde $\alpha(n)$ es una función que crece muy lento.
- Se ha probado que para cualquier aplicación de conjuntos disjuntos $\alpha(n) \leq 4$ (ver Cormen et. al, sección 21.4)
- Se puede ver el tiempo de ejecución con ambas heurísticas como líneal en m para todas las situaciones prácticas.