Disjoint Set Union

- HUGE number of other applications.
- Union(merge): Union of two sets
- Find: Find set where elements belongs
- Universe \( U = \{1, 2, \ldots, n\} \)

![Diagram showing sets and their unions]

Circled number is name of set

- Find(j): find set to which j belongs.
  - Find(8) returns 5
  - Find(5) returns 5
  - Find(2) returns 9
  - Find(4) returns 5
Initial conditions:

Union(11,3)

Union(5,9)

Note: Name of resulting set could be 5 or 9
Array Implementation

- Element j belongs to set $A_j$. Initially...

```
1 2  i  n
A 1 2  i  n
```

- Find(i): return(A[i]). $O(c)$ time

- Union(i,j): Change all i’s to j in Array A (i and j are set names). Takes $O(n)$ time

- Cost of $n$ operations: $O(n^2)$

- Cost of $m > n$ operations: $O(n^2 + m)$
Reverse tree implementation

Find(i)

Union(i, j)

Set parent of i to j

O(c) time

Find takes time is O(h)
h = height

h can be n

n find operations

Take O(n*n)
Example

Time complexity $\sum_{i=2}^{n} i$, which is $O(n^2)$

1 \quad 2 \quad \ldots \quad n

Union(1,2) \quad \text{Find(1)}

1 \quad 2 \\
3 \\
4 \\
n

Union(2,3) \quad \text{Find(1)}

1 \\
3

Union(3,4) \quad \text{Find(1)}

1 \\
2 \\
3 \\
4 \\
5 \\
n
Weighted Union (union by size)

When \( n \leq n' \)

```
1 2 3 4 5 6 7 8
```

\( \text{UNION}(1,2) \ldots \text{UNION}(3,4) \ldots \text{UNION}(5,6) \ldots \text{UNION}(7,8) \)

```
1 2 3 4 5 6 7 8
```

\( \text{UNION}(2,4) \quad \text{UNION}(6,8) \)

```
1 2 3 4 5 6 7 8
```

\( \text{UNION}(4,8) \)

```
1 2 3 4 5 6 7 8
```

```
\begin{tabular}{c|c}
\# & ht \\
1 & 1 \\
2 & 2 \\
4 & 3 \\
8 & 4 \\
\ldots & \\
\hline
2^k & k+1 \\
\end{tabular}
```
Theorem: If we have n nodes → \( h \leq \lfloor \log_2 n \rfloor + 1 \)

Proof:

- basis: \( n = 1 \), implies that \( h = 1 \).
- Ind Hypothesis: Assume true for \( n - 1 \)
- Ind Step: Prove for \( n \geq 2 \),
  
  Last union operation
  
  when \( n - m \leq m \)

- Height of “\( m \)”
  
  Since \( m < n \) then by the ind hypothesis
  
  height of \( m \) \( \leq \lfloor \log_2 m \rfloor + 1 \leq \lfloor \log_2 n \rfloor + 1 \)

- Height of “\( n - m \)” after adding new root
  
  \( \leq \lfloor \log_2 (n - m) \rfloor + 1 + 1 \leq \lfloor \log_2 (n/2) \rfloor + 2 \)
  
  \( \leq \lfloor \log_2 n - 1 \rfloor + 2 \leq \lfloor \log_2 n \rfloor + 1 \)
• No additional space is required to store the number.

  \[
  \begin{array}{c}
  \text{i} \\
  \hline
  \text{-k} \\
  \end{array}
  \quad \begin{array}{c}
  \text{i} \\
  \hline
  \end{array}
  \quad \begin{array}{c}
  \text{k nodes}
  \end{array}
  \]

• \(N\) operations
  – Union takes \(O(c)\) time.
  – Find takes \(O(\log n)\) time.
  – Total time for \(n\) Union-Find operations is \(O(n \log n)\),

• Also possible union by height but we do not do it because some changes (next slide).

• When does the worst case arise for weighted union?
Path Compression

Next find(i) will not be as expensive
Analysis is quite complex (CS230A)

- $M$ operations on $N$ elements
  - Total time $O(M\alpha(M, N))$
  - $O(M\alpha(M, N))$ is a functional inverse of Ackermann’s Function
  - $\alpha(M, N) \leq 4$ as long as $N \leq 2^{65536}$
  - Actually larger than a 20000-digit number.