



Algorithms: Design
and Analysis, Part II

Advanced Union-Find

Path Compression: The
Hopcroft-Ullman Analysis

Hopcroft-Ullman Theorem

Theorem: [Hopcroft-Ullman 73] With Union by Rank and path compression, m UNION+FIND operations take $O(m \log^* n)$ time, where $\log^* n$ = the number of times you need to apply log to n before the result is ≤ 1 .

[Will focus on interesting case where $m = \Omega(n)$]

Measuring Progress

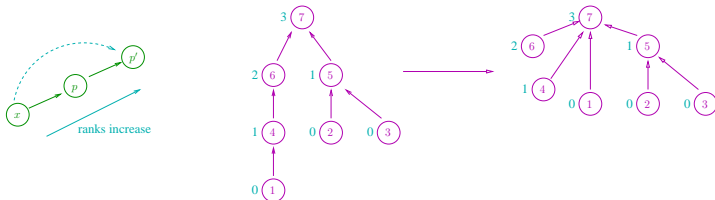
Intuition: Installing shortcuts should significantly speed up subsequent FINDs+UNIONS.

Question: How to track this progress and quantify the benefit?

Idea: Consider a non-root object x . \longrightarrow Recall: $\text{rank}[x]$ frozen


Progress measure: $\text{rank}[\text{parent}[x]] - \text{rank}[x]$

Path compression increases this progress measure: If x has old parent p , new parent $p' \neq p$, then $\text{rank}[p'] > \text{rank}[p]$.



Proof Setup

Rank blocks: $\{0\}, \{1\}, \{2, 3, 4\}, \{5, \dots, 2^4\}, \{17, 18, \dots, 2^{16}\},$
 $\{65537, \dots, 2^{65536}\}, \dots, \{\dots, n\}$



The diagram shows two blue numbers, 16 and 65536, with arrows pointing to the corresponding exponents in the rank blocks sequence. An arrow points from 16 to 2^4 in the block $\{5, \dots, 2^4\}$. Another arrow points from 65536 to 2^{65536} in the block $\{65537, \dots, 2^{65536}\}$.

Note: There are $O(\log^* n)$ different rank blocks.

Semantics: Traversal $x \rightarrow \text{parent}(x)$ is “fast progress” \iff
 $\text{rank}[\text{parent}[x]]$ in larger block than $\text{rank}[x]$

Definition: At a given point in time, call object x good if

- (1) x or x 's parent is a root OR
- (2) $\text{rank}[\text{parent}[x]]$ in larger block than $\text{rank}[x]$

x is bad otherwise.

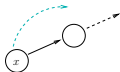
Proof of Hopcroft-Ullman

Point: Every FIND visits only $O(\log^* n)$ good nodes $[2 + \# \text{ of rank blocks} = O(\log^* n)]$

Upshot: Total work done during m operations $= O(m \log^* n)$
(visits to good objects) + total $\#$ of visits to bad nodes (need to bound globally by separate argument)

Consider: A rank block $\{k + 1, k + 2, \dots, 2^k\}$.

Note: When a bad node is visited



its parent is changed to one with strictly larger rank \Rightarrow Can only happen 2^k times before x becomes good (forevermore).

Proof of Hopcroft-Ullman II

Total work: $O(m \log^* n) + O(\text{\# visits to bad nodes})$.

$\leq n$ for each of $O(\log^* n)$ rank blocks

Consider: A rank block $\{k+1, k+2, \dots, 2^k\}$.

Last slide: For each object x with final rank in this block, $\text{\# visits to } x \text{ while } x \text{ is bad}$ is $\leq 2^k$.

Rank Lemma: Total number of objects x with final rank in this rank block is $\sum_{i=k+1}^{2^k} n/2^i \leq n/2^k$.

$\leq n$ visits to bad objects in this rank block.

Recall: Only $O(\log^* n)$ rank blocks.

Total work: $O((m+n) \log^* n)$.