- Classes represented by reversed trees
- Union : add a semi-class as child of the root of the other
  Union by rank : we take as child that with a smaller rank
  Rank : accurate enough upper bound of the heigth of each node.
- Find : we have to cross a path of elements in the class
  Path compression : when traversing the path (by the way, after that when going back through the stack of pending cells) we set all the nodes in it as children of the root of the class.

Analysis of MFU with union by rank and path compression

- A simple Ackermann's function
  - $f^{(i)} = \underbrace{f \circ \dots \circ f}_{i}$ $\quad A_0(j) = j+1$; $A_k(j) = A_{k-1}^{(j+1)}(j)$.
  - $A_4(1) \gg 10^{80}$!!
- Pseudo-inverse : $\alpha(n) = \min\{k \mid A_k(1) \geq n\}$ $\quad \alpha(n) \leq 4$ for any reasonable n.
- Ranks of nodes
  rank(n) can increase while n is the root of a class ; it increases by 1 when we add a subclass with the same rank.
  $p(n) \neq n \Rightarrow$ rank$(n) <$ rank$(p(n))$.
  Remark : rank(n) could be greater than the heigth of the corresponding subclass due to the fact that path compression removes some subtrees without changing rank at all (even so, union by rank still works !!)
  rank$(n) \leq N+1$ (we start assigning rank$(n) := 0$) N is the full size of the family.
- We replace a call to Union by 2 Find and a Link
  Link is the particular case of Union where we have as inputs the representatives of the two classes.
  Trivially, an amortized cost $O(f(n))$ for MFL gives us also the same cost for MFU.
- Potential function Φ (each nodes contributes by $\phi(n)$)
  Each root contributes by $\alpha(N) \cdot$ rank(n). For nodes that are not roots, we define
  $\text{Level}(n) = \max\{k \mid \text{rank}(p(n)) \geq A_k(\text{rank}(n))\}$
  $0 \leq \text{Level}(n)$; $\text{Level}(n) < \alpha(N)$; Level(n) increases over time.
  (rank$(n) \geq 1$) $\text{Iter}(n) = \max\{i \mid \text{rank}(p(n)) \geq A^{(i)}_{\text{level}(n)}(\text{rank}(n))\}$
  $1 \leq \text{Iter}(n)$; $\text{Iter}(n) \leq \text{rank}(n)$; Iter(n) only can decrease when Level(n) increases.
  Each n with rank$(n)=0$ contributes to Φ by 0.
  Each n with rank$(n)>0$ contributes by $(\alpha(N) - \text{level}(n))\,\text{rank}(n) - \text{iter}(n)$
- Lemma 1 : i) $0 \leq \phi(n)$ using the obtained bounds for level(n) and iter(n).
  ii) $\phi_n \leq \alpha(N) \cdot \text{rank}(n)$, in this case we use their lower bounds.
  Moreover, this disequality is strict when n is not a root and rank$(n) > 0$.

- Amortized costs of operations

1) If n is not a root, then after LINK or FIND $\phi(n)$ does not increase. Moreover, if $rank(n) > 0$ and either $level(n)$ or $iter(n)$ changes after the operation, then $\phi(n)$ strictly decreases.

Proof: When $level(n)$ increases, $iter(n)$ can only decrease by $rank(n) - 1$.

2) The amortized cost of MAKE is $O(1)$.

3) The amortized cost of LINK is $O(\alpha(N))$. (We add x as child of y)
   - Only the potential of x, y, and the other children of y can change.
     - None of these other children can have its potential increase.
     - $\phi(x) = \alpha(N) \cdot rank(x)$ and becomes $\phi'(x) = 0$ when $rank(x) = 0$, or $\phi'(x) < \alpha(N) \cdot rank(x)$ as stated in Lemma 1.
     - Since y was and remains a root, the only change in $\phi(y)$ would come when $rank(y)$ is increased by 1, and then $\phi'(y) = \phi(y) + \alpha(N)$
     - The possible increase of Φ when applying a LINK is $\alpha(N)$

4) The amortized cost of FIND is $O(\alpha(N))$
   - If the ascending path from the input node contains s nodes, then s is the actual cost of the operation.
   - No node's potential increases: applying the lemma and the fact that the rank of the root does not change.
   - At least the potential of $\max(0, s - (\alpha(N)+2))$ nodes decreases.

     We consider the nodes in the path with $rank(x) > 0$ that are followed in the way up (possibly not immediately) by some y that is not a root and satisfying $level(x) = level(y)$.

     There are at least $s - (\alpha(N)+2)$ such nodes.

     For each such node $\phi'(x) < \phi(x)$, by applying the lemma after making several quite bright reasonings

Any collection of m operations, either MFL or MFU, producing a classification of a set S with $|S| = N$, takes a time $O(m\,\alpha(N))$.