Master Theorem

The master method depends on the following theorem.

Theorem 4.1 (Master theorem)
Let $a \geq 1$ and $b > 1$ be constants, let $f(n)$ be a function, and let $T(n)$ be defined on the nonnegative integers by the recurrence

$$T(n) = aT(n/b) + f(n) ,$$

where we interpret n/b to mean either $\lfloor n/b \rfloor$ or $\lceil n/b \rceil$. Then $T(n)$ has the following asymptotic bounds:

1. If $f(n) = O(n^{\log_b a - \epsilon})$ for some constant $\epsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$.
2. If $f(n) = \Theta(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a} \lg n)$.
3. If $f(n) = \Omega(n^{\log_b a + \epsilon})$ for some constant $\epsilon > 0$, and if $af(n/b) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$. ■

Conjunto dinámico (dynamic set)

Dado un universo $\boldsymbol{U}$={0,1,…,u-1} de valores de claves posibles.

Sea ***S*** un subconjunto de elementos cuyos claves están en $\boldsymbol{U}$={0,1,…,u-1} (sin repetición). La estructura de datos de implementación debe soportar las siguientes operaciones.

- CREATE-SET(***S***)
- MEMBER(***S***,key) $\rightarrow \{true, false\}$
- INSERT(***S***,key)
- DELETE(***S***,key)
- MINIMUM(***S***) $\rightarrow U$
- MAXIMUM(***S***) $\rightarrow U$
- PREDECESSOR(***S***,key) $\rightarrow U$
- SUCCESSOR(***S***,key) $\rightarrow U$

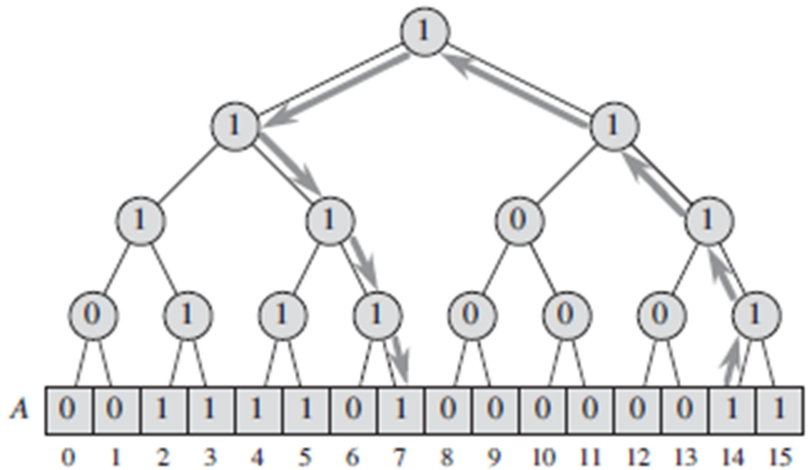
Direccionamiento directo

Utiliza un arreglo A de bits de dimensión u cuyos índices van de 0 a u-1. A[i] = i ⇔ i∈S

A	0	0	1	1	1	1	0	1	0	0	0	0	0	0	1	1
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

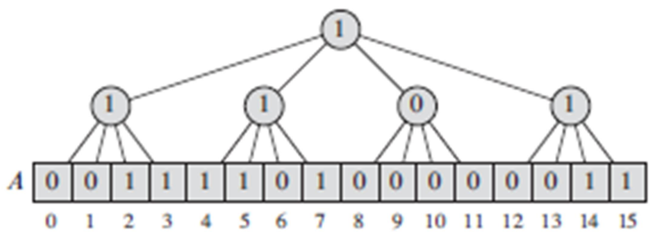
- CREATE-SET(***S***) O(u)
- MEMBER(***S***,key) O(1)
- INSERT(***S***,key) O(1)
- DELETE(***S***,key) O(1)
- MINIMUM(***S***) O(u)
- MAXIMUM(***S***) O(u)
- PREDECESSOR(***S***,key) O(u)
- SUCCESSOR(***S***,key) O(u)

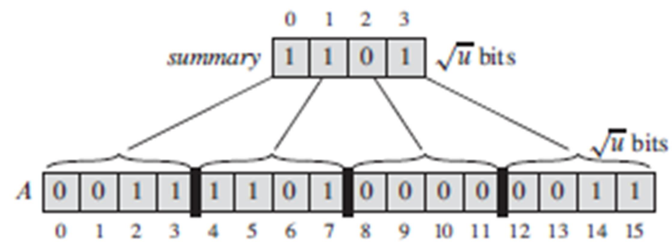
Direccionamiento directo + superposición árbol binario



- CREATE-SET(***S***) O(u)
- MEMBER(***S***,key) O(1)
- INSERT(***S***,key) O(log u)
- DELETE(***S***,key) O(log u)
- MINIMUM(***S***) O(log u)
- MAXIMUM(***S***) O(log u)
- PREDECESSOR(***S***,key) O(log u)
- SUCCESSOR(***S***,key) O(log u)

Superposición árbol de altura constante





Si $u = 2^{2k}$ para algún k entero entonces $\sqrt{u} = 2^k$, $high(x) = \left\lfloor \frac{x}{\sqrt{u}} \right\rfloor$ y $low(x) = x \bmod \sqrt{u}$

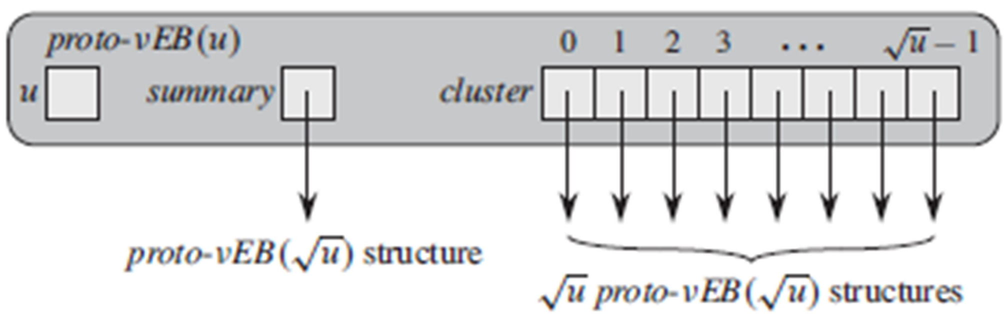
- CREATE-SET(***S***) O(u)
- MEMBER(***S***,key) O(1)
- INSERT(***S***,key) O(1)
- DELETE(***S***,key) O($\sqrt{u}$)
- MINIMUM(***S***) O($\sqrt{u}$)
- MAXIMUM(***S***) O($\sqrt{u}$)
- PREDECESSOR(***S***,key) O($\sqrt{u}$)
- SUCCESSOR(***S***,key) O($\sqrt{u}$)

¿Qué ocurre si en lugar de A, sean $\sqrt{u}$ arreglos de $\sqrt{u}$ elementos cada uno (se crean cdo se usan)?

Estructura recursiva – Proto Van Emde Boas Trees

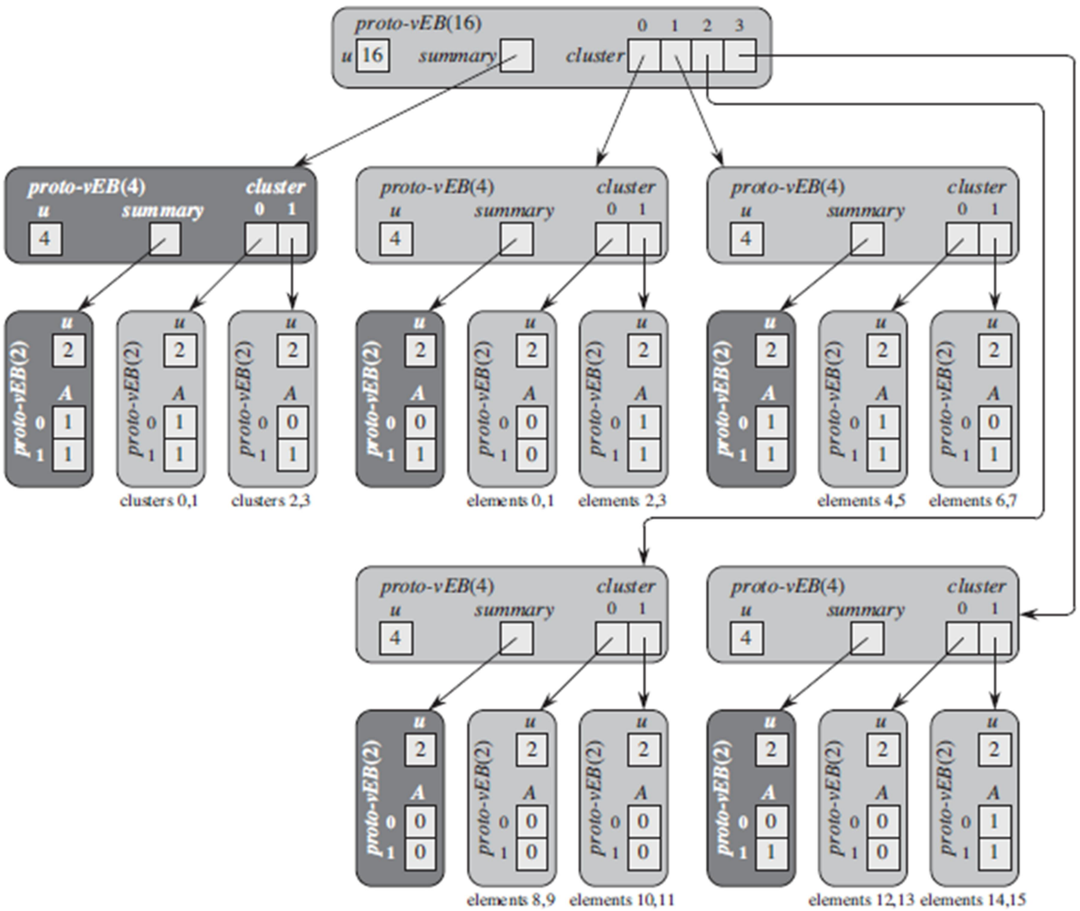
Suponer que $u = 2^{2^k}$ por ahora, los posibles valores de u son 2, 4, 16, 256, 65536,…

$$\begin{aligned} \text{high}(x) &= \lfloor x/\sqrt{u} \rfloor , \\ \text{low}(x) &= x \bmod \sqrt{u} , \\ \text{index}(x, y) &= x\sqrt{u} + y . \end{aligned} \qquad x = \text{index}(\text{high}(x), \text{low}(x))$$



Después de log log u = k niveles llega a proto-vEB(2)

Proto Van Emde Boas Trees



{2, 3, 4, 5, 7, 14, 15}.

Proto Van Emde Boas Trees

```
PROTO-vEB-MEMBER(V, x)
if V.u == 2
    return V.A[x]
else return PROTO-vEB-MEMBER(V.cluster[high(x)], low(x))
```

$T(u) = T(\sqrt{u}) + O(1)$

Let $m = \lg u$, so that $u = 2^m$ $T(2^m) = T(2^{m/2}) + O(1)$

we rename $S(m) = T(2^m)$, $S(m) = S(m/2) + O(1)$

$T(u) = T(2^m) = S(m) = O(\lg m) = O(\lg \lg u)$.

Proto Van Emde Boas Trees

```
PROTO-vEB-MINIMUM(V)
if V.u == 2
    if V.A[0] == 1
        return 0
    elseif V.A[1] == 1
        return 1
    else return NIL
else min-cluster = PROTO-vEB-MINIMUM(V.summary)
    if min-cluster == NIL
        return NIL
    else offset = PROTO-vEB-MINIMUM(V.cluster[min-cluster])
        return index(min-cluster, offset)
```

$T(u) = 2T(\sqrt{u}) + O(1)$ $m = \lg u$ $T(2^m) = 2T(2^{m/2}) + O(1)$ $S(m) = T(2^m)$ $S(m) = 2S(m/2) + O(1)$

$T(u) = T(2^m) = S(m) = \Theta(m) = \Theta(\lg u)$

Proto Van Emde Boas Trees

```
PROTO-vEB-SUCCESSOR(V, x)
if V.u == 2
    if x == 0 and V.A[1] == 1
        return 1
    else return NIL
else offset = PROTO-vEB-SUCCESSOR(V.cluster[high(x)], low(x))
    if offset ≠ NIL
        return index(high(x), offset)
    else succ-cluster = PROTO-vEB-SUCCESSOR(V.summary, high(x))
        if succ-cluster == NIL
            return NIL
        else offset = PROTO-vEB-MINIMUM(V.cluster[succ-cluster])
            return index(succ-cluster, offset)
```

$$\begin{aligned} T(u) &= 2T(\sqrt{u}) + \Theta(\lg \sqrt{u}) \\ &= 2T(\sqrt{u}) + \Theta(\lg u) \; . \end{aligned}$$

$\mathrm{m} = \log u, \quad T(2^m) = 2T\left(2^{\frac{m}{2}}\right) + m, \;\; sea\; S(m) = T(2^m), S(m) = 2s\left(\frac{m}{2}\right) + m = O(m \log m) = O(\log u \log \log u) = T(u)$

Proto Van Emde Boas Trees

```
PROTO-vEB-INSERT(V, x)
if V.u == 2
    V.A[x] = 1
else PROTO-vEB-INSERT(V.cluster[high(x)], low(x))
    PROTO-vEB-INSERT(V.summary, high(x))
```

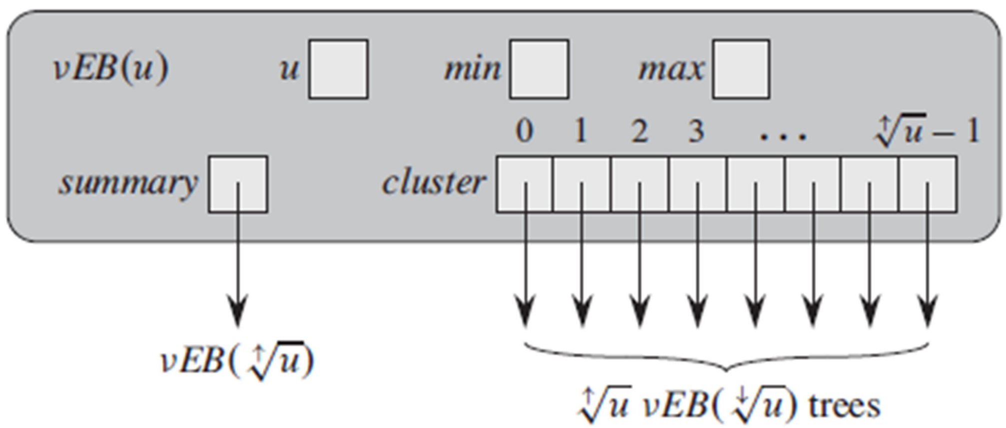
Similar a PROTO-VEB-MINIMUM ($O(\log u)$)

¿Cómo se implementan PROTO-VEB-MAXIMUM, PROTO-VEB-PREDECESSOR y PROTO-VEB-DELETE?

Van Emde Boas Trees

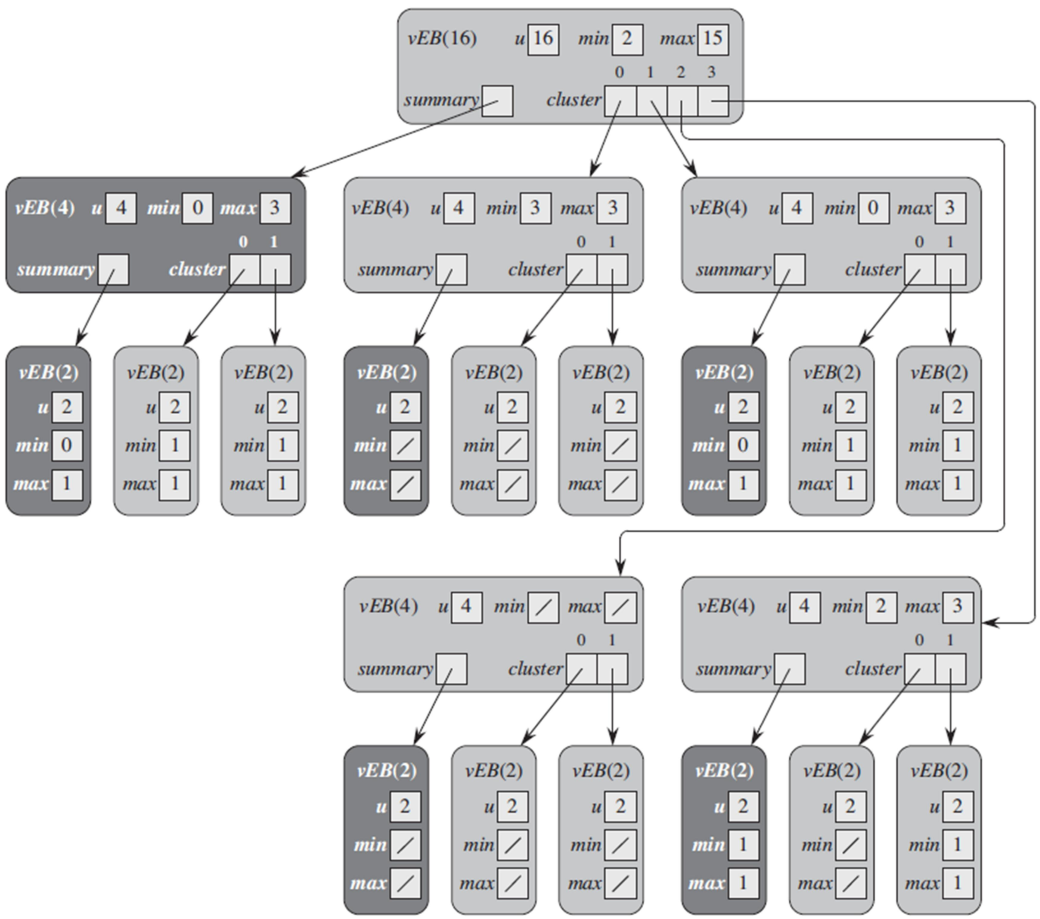
Ahora vamos a relajar la condición sobre $u = 2\hat{}k, k \in Z_{\geq 0}$, $\sqrt{u}$ ya puede no ser potencial 2 (ni siquiera sea entero). Por lo tanto se separan los k bits de u, los primeros $\left\lceil \frac{\log u}{2} \right\rceil$ bits más significativos y $\left\lfloor \frac{\log u}{2} \right\rfloor$ bits menos significativos. Sean $\sqrt[\uparrow]{u} = 2^{\left\lceil \frac{\log}{2} \right\rceil}$ y $\sqrt[\downarrow]{u} = 2^{\left\lfloor \frac{\log u}{2} \right\rfloor}$, claramente $u = \sqrt[\uparrow]{u} \times \sqrt[\downarrow]{u}$.

$$\begin{aligned} \text{high}(x) &= \left\lfloor x / \sqrt[\downarrow]{u} \right\rfloor , \\ \text{low}(x) &= x \bmod \sqrt[\downarrow]{u} , \\ \text{index}(x, y) &= x \sqrt[\downarrow]{u} + y . \end{aligned}$$



min guarda el mínimo y max guarda el máximo del árbol vEB. Es más el mínimo ya no se guarda en el subárbol correspondiente con el tamaño del universo $\sqrt[\downarrow]{u}$ ni es tenido en cuenta en summary.

Van Emde Boas Trees



Van Emde Boas Trees

OPERACIÓN	Proto-vEB	vEB
MEMBER(***S***,key)	$O(\log \log u)$	$O(\log \log u)$
INSERT(***S***,key)	$O(\log u \log \log u)$	$O(\log \log u)$
DELETE(***S***,key)	$O(\log u \log \log u)$	$O(\log \log u)$
MINIMUM(***S***)	$O(\log u)$	$O(1)$
MAXIMUM(***S***)	$O(\log u)$	$O(1)$
PREDECESSOR(***S***,key)	$O(\log u)$	$O(\log \log u)$
SUCCESSOR(***S***,key)	$O(\log u)$	$O(\log \log u)$

```
vEB-TREE-MINIMUM(V)
return V.min
```

```
vEB-TREE-MAXIMUM(V)
return V.max
```

```
vEB-TREE-MEMBER(V, x)
if x == V.min or x == V.max
    return TRUE
elseif V.u == 2
    return FALSE
else return vEB-TREE-MEMBER(V.cluster[high(x)], low(x))
```

Van Emde Boas Trees

```
vEB-TREE-SUCCESSOR(V, x)
if V.u == 2
    if x == 0 and V.max == 1
        return 1
    else return NIL
elseif V.min ≠ NIL and x < V.min
    return V.min
else max-low = vEB-TREE-MAXIMUM(V.cluster[high(x)])
    if max-low ≠ NIL and low(x) < max-low
        offset = vEB-TREE-SUCCESSOR(V.cluster[high(x)], low(x))
        return index(high(x), offset)
    else succ-cluster = vEB-TREE-SUCCESSOR(V.summary, high(x))
        if succ-cluster == NIL
            return NIL
        else offset = vEB-TREE-MINIMUM(V.cluster[succ-cluster])
            return index(succ-cluster, offset)
```

Van Emde Boas Trees

```
vEB-TREE-PREDECESSOR(V, x)
if V.u == 2
    if x == 1 and V.min == 0
        return 0
    else return NIL
elseif V.max ≠ NIL and x > V.max
    return V.max
else min-low = vEB-TREE-MINIMUM(V.cluster[high(x)])
    if min-low ≠ NIL and low(x) > min-low
        offset = vEB-TREE-PREDECESSOR(V.cluster[high(x)], low(x))
        return index(high(x), offset)
    else pred-cluster = vEB-TREE-PREDECESSOR(V.summary, high(x))
        if pred-cluster == NIL
            if V.min ≠ NIL and x > V.min
                return V.min
            else return NIL
        else offset = vEB-TREE-MAXIMUM(V.cluster[pred-cluster])
            return index(pred-cluster, offset)
```

Van Emde Boas Trees

```
vEB-EMPTY-TREE-INSERT(V, x)
V.min = x
V.max = x
```

```
vEB-TREE-INSERT(V, x)
if V.min == NIL
    vEB-EMPTY-TREE-INSERT(V, x)
else if x < V.min
        exchange x with V.min
    if V.u > 2
        if vEB-TREE-MINIMUM(V.cluster[high(x)]) == NIL
            vEB-TREE-INSERT(V.summary, high(x))
            vEB-EMPTY-TREE-INSERT(V.cluster[high(x)], low(x))
        else vEB-TREE-INSERT(V.cluster[high(x)], low(x))
    if x > V.max
        V.max = x
```

Van Emde Boas Trees

```
vEB-TREE-DELETE(V, x)
if V.min == V.max
    V.min = NIL
    V.max = NIL
elseif V.u == 2
    if x == 0
        V.min = 1
    else V.min = 0
```

```
else if x == V.min
        first-cluster = vEB-TREE-MINIMUM(V.summary)
        x = index(first-cluster,
            vEB-TREE-MINIMUM(V.cluster[first-cluster]))
        V.min = x
    vEB-TREE-DELETE(V.cluster[high(x)], low(x))
    if vEB-TREE-MINIMUM(V.cluster[high(x)]) == NIL
        vEB-TREE-DELETE(V.summary, high(x))
        if x == V.max
            summary-max = vEB-TREE-MAXIMUM(V.summary)
            if summary-max == NIL
                V.max = V.min
            else V.max = index(summary-max,
                    vEB-TREE-MAXIMUM(V.cluster[summary-max]))
    elseif x == V.max
        V.max = index(high(x),
            vEB-TREE-MAXIMUM(V.cluster[high(x)]))
```

Van Emde Boas Trees

¿Cuánto espacio ocupa un vEB(u)?

$$S(u) = (\sqrt{u} + 1) \times S(\sqrt{u}) + \theta(\sqrt{u}) \leq (\sqrt{u} + 1) \times S(\sqrt{u}) + c1\sqrt{u}$$

$$S(u^2) = (u + 1) \times S(u) + \theta(u) \leq (u + 1) \times S(u) + c1 \times u$$

$$Sea\ c2 = \max\{S(4), c1\}, c1 \leq c2\ y\ S(4) \leq c2$$

$$Definimos\ S'(u) = \frac{S(u)}{c2}, S'(u^2) \leq (u + 1) \times \frac{S(u)}{c2} + \frac{c1}{c2}u \leq (u + 1)S'(u) + u$$

$$S'(4) = \frac{S(4)}{c2} \leq 1$$

Probamos a continuación, $S'(u) \leq u - 2\ para\ u \geq 4$

- Caso base, $u = 4, S'(4) \leq 1 \leq 4 - 2 = 2$
- Supongamos que vale $S'(u) \leq u - 2$ ahora veamos que $S'(u^2) \leq u^2 - 2$

  $S'(u^2) \leq (u + 1)S'(u) + u \leq (u + 1) \times (u - 2\ ) + u = u^2 - 2$ ■

$$\boldsymbol{S(u) = c2 \times S'(u) \leq c2 \times (u - 2) = O(u)}$$