Project ADS: Random Deletions and Insertions on BST

Author:
Alex Herrero

Professors:
Conrado Martínez
Amalia Duch
Salvador Roura

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat d'Informàtica de Barcelona

Acknowledgment

Special thanks to Conrado Martínez. His $\Theta(A(2^{n!}, m \log m))$[1] wisdom and advice have been helpful throughout this project.

[1] Where A is the Ackermann function. Not the inverse!!

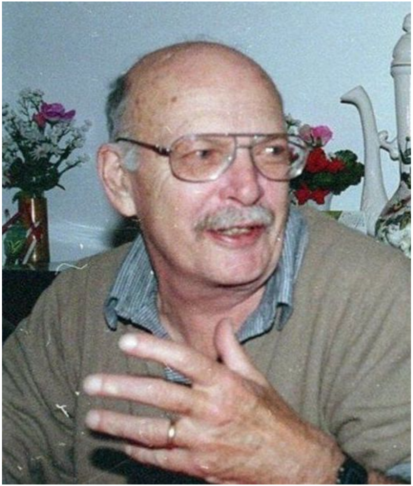
Table of Contents

1. 1962: Hibbard
2. 1975: Knott
3. 1983: Eppinger

1. 1962: Hibbard
2. 1975: Knott
3. 1983: Eppinger

Introduction to BST

- He introduced the concept of BST on his paper: Thomas N. Hibbard. "Some Combinatorial Properties of Certain Trees With Applications to Searching and Sorting". In: *J. ACM* 9.1 (Jan. 1962), pp. 13–28. ISSN: 0004-5411. DOI: 10.1145/321105.321108. URL: https://doi.org/10.1145/321105.321108.



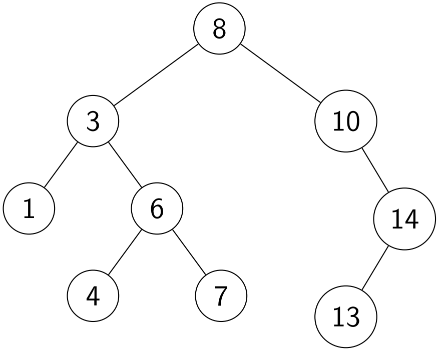
Thomas Hibbard (1929-2016)

- He introduced not only the concept of binary search trees (BSTs), but also the idea of randomness in BSTs.

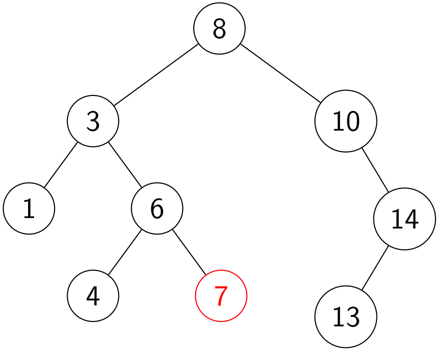
- He introduced not only the concept of binary search trees (BSTs), but also the idea of randomness in BSTs.
- Well-known concepts and algorithms for every computer scientist.

- He introduced not only the concept of binary search trees (BSTs), but also the idea of randomness in BSTs.
- Well-known concepts and algorithms for every computer scientist.
- We will take a look to Hibbard's deletion algorithm.

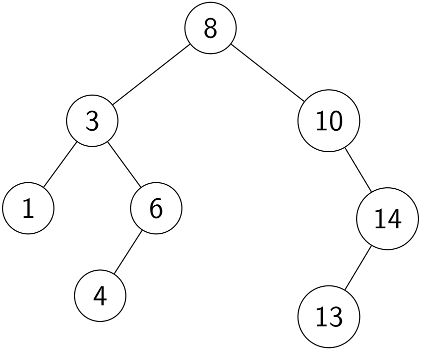
Leaf case



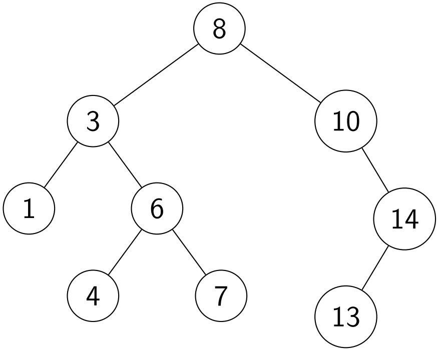
Leaf case



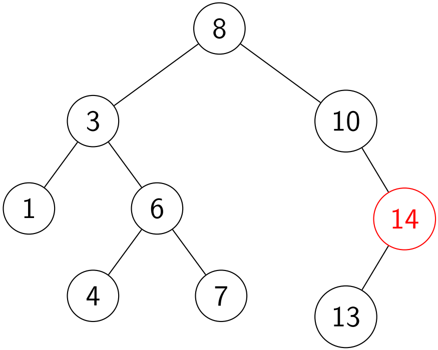
Leaf case



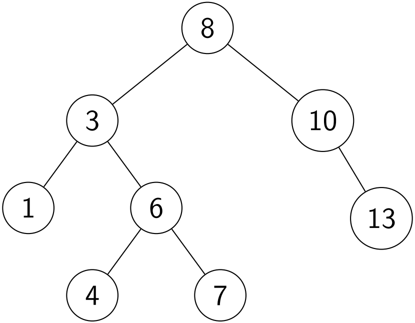
Only one subtree case



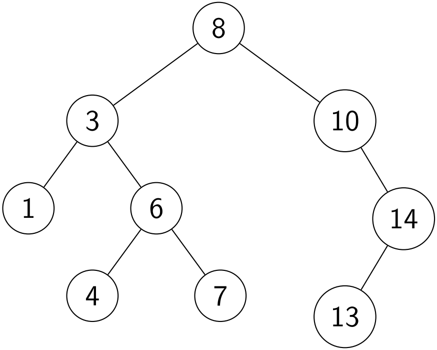
Only one subtree case



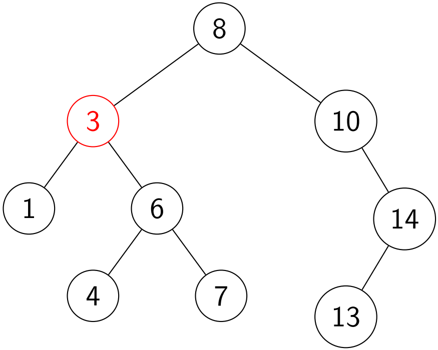
Only one subtree case



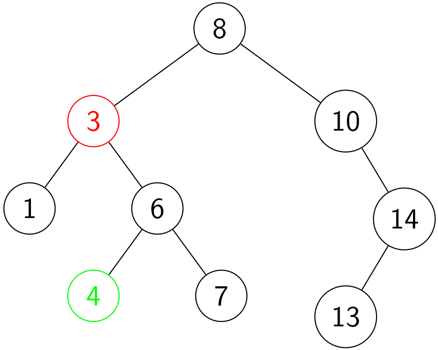
Two subtree case



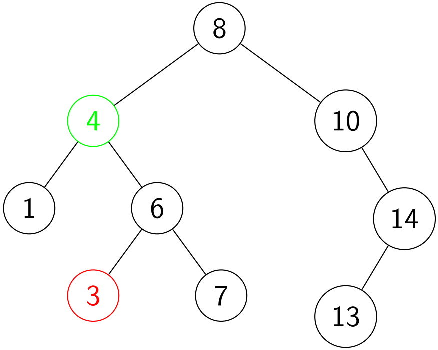
Two subtree case



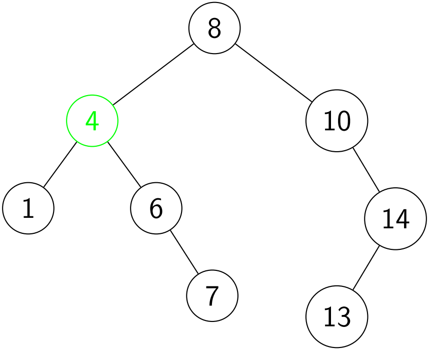
Two subtree case



Two subtree case



Two subtree case



```
function DELETE(T, x)
    if T.val < x then
        T.right ← DELETE(T.right, x)
    else if T.val > x then
        T.left ← DELETE(T.left, x)
    else
        if T.right = null then
            return T.left
        else
            T.val ← MINVALUE(T.right)
            T.right ← DELETE(T.right, T.val)
        end if
    end if
    return T
end function
```

Hibbard's paper was remarkable in that it contained one of the first formal theorems about algorithms:

Hibbard's paper was remarkable in that it contained one of the first formal theorems about algorithms:

Hibbard's Theorem (1962)

If $n + 1$ items are inserted into an initially empty binary tree, in random order, and if one of those items (selected at random) is deleted, the probability that the resulting binary tree has a given shape is the same as the probability that this tree shape would be obtained by inserting n items into an initially empty tree, in random order.

1. 1962: Hibbard
2. 1975: Knott
3. 1983: Eppinger

- It was believed for more than a decade that Hibbard's algorithm preserved randomness on BSTs[a]

- It was believed for more than a decade that Hibbard's algorithm preserved randomness on BSTs[a]
- 1975: Knott, *Deletion in binary storage trees.*

[a] In fact this appeared on the first edition of the well-known book *The Art of Computer Programming: Vol 2* by Donald Knuth (1973)

Knott Paradox

Although Hibbard's theorem establishes that n+1 random insertions followed by a random deletion produce a tree whose shape has the distribution of n random insertions, it does not follow that a subsequent random insertion yields a tree whose shape has the distribution of n+1 random insertions

We will follow Jonassen and Knuth, “A trivial algorithm whose analysis isn't” for a BST of size $n = 3$

We will follow Jonassen and Knuth, “A trivial algorithm whose analysis isn't” for a BST of size $n = 3$ Suppose we have three elements $x < y < z$

All BSTs for $x < y < z$

(x, y, z) (x, z, y) $(y, x, z) = (y, z, x)$ (z, x, y) (z, y, x)

```
x          x                              z         z
 \          \                            /         /
  y          z            y             x         y
   \        /            / \             \       /
    z      y            x   z             y     x
```

RR RL B LR LL

Permutation	Delete x	Delete y	Delete z
(x, y, z)	R	R	R
(x, z, y)	R	R	R
$(y, z, x) = (y, x, z)$	R	L	L
(z, x, y)	L	R	R
(z, y, x)	L	L	L

$$\mathbb{P}[L] = \mathbb{P}[R] = \frac{9}{18} = \frac{1}{2}$$

Now another random insertion w comes to the BST. Then we have four possible cases:

- $w < x < y < z$
- $x < w < y < z$
- $x < y < w < z$
- $x < y < z < w$

18 previous cases and 4 possibilities for w give us a total of 72 cases.

$w < x < y < z$

Permutation	Delete x	Delete y	Delete z
(x, y, z)	B	B	B
(x, z, y)	B	B	B
$(y, z, x) = (y, x, z)$	B	LL	LL
(z, x, y)	LL	LL	B
(z, y, x)	LL	LL	LL

$x < w < y < z$

Permutation	Delete x	Delete y	Delete z
(x, y, z)	B	RL	RL
(x, z, y)	B	RL	RL
$(y, z, x) = (y, x, z)$	B	LR	LR
(z, x, y)	LL	LR	RL
(z, y, x)	LL	LR	LR

$x < y < w < z$

Permutation	Delete x	Delete y	Delete z
(x, y, z)	RL	RL	RR
(x, z, y)	RL	RL	RR
$(y, z, x) = (y, x, z)$	RL	LR	B
(z, x, y)	LR	LR	RR
(z, y, x)	LR	LR	B

$x < y < z < w$

Permutation	Delete x	Delete y	Delete z
(x, y, z)	RR	RR	RR
(x, z, y)	RR	RR	RR
$(y, z, x) = (y, x, z)$	RR	B	B
(z, x, y)	B	B	RR
(z, y, x)	B	B	B

Probabilities

$$\mathbb{P}[LL] = \frac{11}{72} \qquad \mathbb{P}[LR] = \frac{13}{72}$$

$$\mathbb{P}[RL] = \frac{11}{72} \qquad \mathbb{P}[RR] = \frac{12}{72}$$

$$\mathbb{P}[B] = \tfrac{25}{72}$$

Probabilities

$$\mathbb{P}[LL] = \frac{11}{72} \qquad \mathbb{P}[LR] = \frac{13}{72}$$

$$\mathbb{P}[RL] = \frac{11}{72} \qquad \mathbb{P}[RR] = \frac{12}{72}$$

$$\mathbb{P}[B] = \tfrac{25}{72}$$

Probability L Shape

The probability of having an L shape after a random deletion is:
$\mathbb{P}[L] = \mathbb{P}[LL] + \frac{2}{3}\mathbb{P}[LR] + \frac{2}{3}\mathbb{P}[B] = \frac{11}{72} + \frac{2}{3} \cdot \frac{13}{72} + \frac{2}{3} \cdot \frac{25}{72} = \frac{109}{216} > \frac{1}{2}!!$

Knuth, *The Art of Computer Programming: Sorting and Searching, volume 3*

The shape of the tree is random after deletions, but the relative distribution of values in a given tree shape may change, and it turns out that the first random insertion. after a deletion actually destroys the randomness property on shapes. This startling fact, first observed by Gary Knott in 1972, must be seen to be believed

Knott was the first to notice that Hibbard's generalization was wrong.
In his thesis also gave some empirical data summarizing the results of simulation experiments, where BSTs randomly constructed by $I^n(ID)^m$. Leading to the following conjecture:

Knott's conjecture

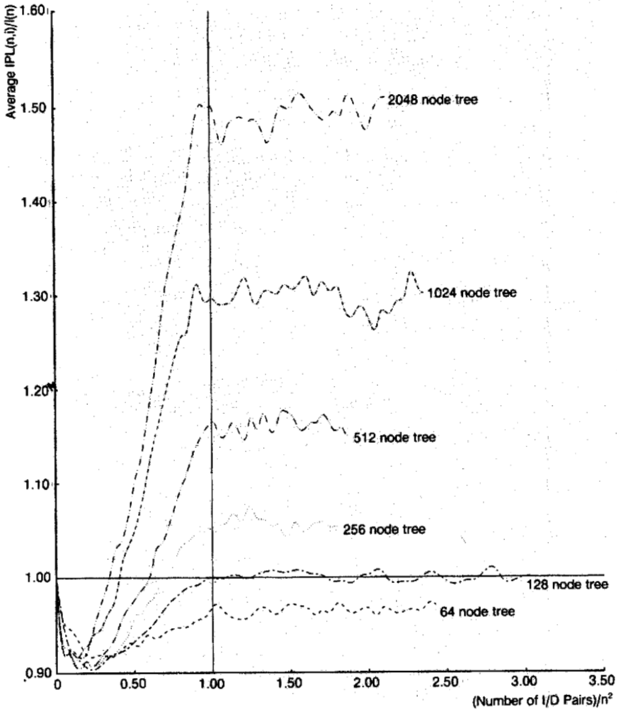
[a] Empirical evidence suggests strongly that the path length tends to decrease after repeated deletions and insertions, so the departure from randomness seems to be in the right direction; a theoretical explanation for this behavior is still lacking.

[a] Knuth, *The Art of Computer Programming: Sorting and Searching, volume 3*.

1. 1962: Hibbard
2. 1975: Knott
3. 1983: Eppinger

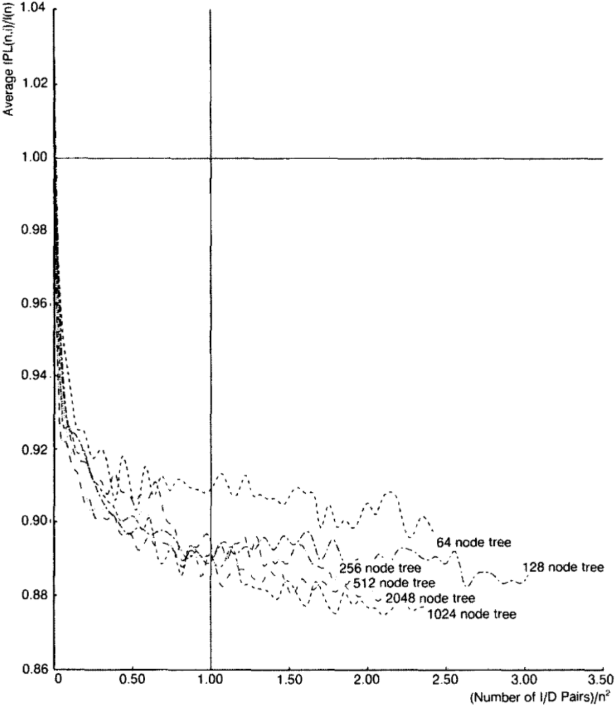
- Jeffrey L. Eppinger. "An empirical study of insertion and deletion in binary search trees". In: *Commun. ACM* 26.9 (Sept. 1983), pp. 663–669. ISSN: 0001-0782. DOI: 10.1145/358172.358183. URL: https://doi.org/10.1145/358172.358183.
- A landmark in experimental algorithmic literature

Jeffrey Eppinger (1960)



Average IPL(n,i)/I(n)
1.60
1.50
1.40
1.30
1.20
1.10
1.00
0.90
0
0.50
1.00
1.50
2.00
2.50
3.00
3.50
(Number of I/D Pairs)/n²
2048 node tree
1024 node tree
512 node tree
256 node tree
128 node tree
64 node tree

```
function SYMMETRIC DELETE(T, x)
    if T.val < x then
        T.right ← SYMMETRIC DELETE(T.right, x)
    else if T.val > x then
        T.left ← SYMMETRIC DELETE(T.left, x)
    else
        if T.right = null then
            return T.left
        else
            if FLIPCOIN() = Head then
                T.val ← MINVALUE(T.right)
                T.right ← SYMMETRIC DELETE(T.right, T.val)
            else
                T.val ← MAXVALUE(T.left)
                T.left ← SYMMETRIC DELETE(T.left, T.val)
            end if
        end if
    end if
    return T
end function
```



Average IPL(n,i)/I(n)
1.04
1.02
1.00
0.98
0.96
0.94
0.92
0.90
0.88
0.86
0
0.50
1.00
1.50
2.00
2.50
3.00
3.50
(Number of I/D Pairs)/n^2
64 node tree
128 node tree
256 node tree
512 node tree
2048 node tree
1024 node tree

- Eppinger, Jeffrey L. "An empirical study of insertion and deletion in binary search trees". In: *Commun. ACM* 26.9 (Sept. 1983), pp. 663–669. ISSN: 0001-0782. DOI: 10.1145/358172.358183. URL: https://doi.org/10.1145/358172.358183.
- Hibbard, Thomas N. "Some Combinatorial Properties of Certain Trees With Applications to Searching and Sorting". In: *J. ACM* 9.1 (Jan. 1962), pp. 13–28. ISSN: 0004-5411. DOI: 10.1145/321105.321108. URL: https://doi.org/10.1145/321105.321108.
- Jonassen, Arne T and Donald E Knuth. "A trivial algorithm whose analysis isn't". In: *Journal of computer and system sciences* 16.3 (1978), pp. 301–322.
- Knott, Gary Don. *Deletion in binary storage trees.* Stanford University, 1975.
- Knuth, Donald E. *The Art of Computer Programming: Sorting and Searching, volume 3.* Addison-Wesley Professional, 1998.

Project ADS: Random Deletions and Insertions on BST

Author:
Alex Herrero

Professors:
Conrado Martínez
Amalia Duch
Salvador Roura

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat d'Informàtica de Barcelona