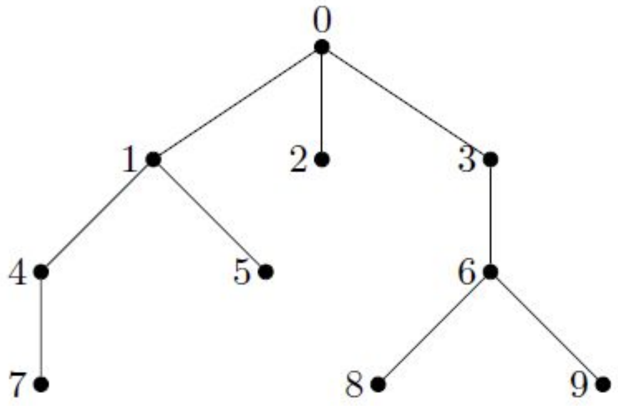
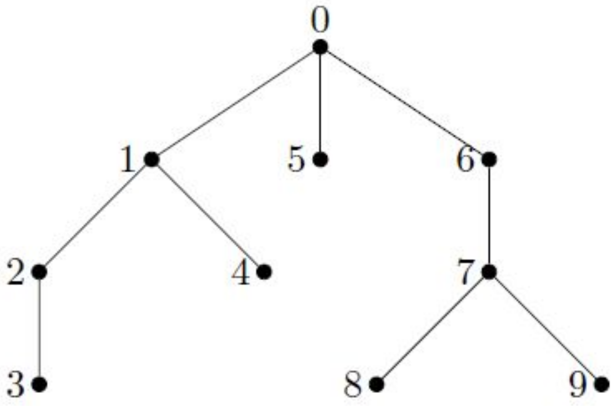
BFS vs DFS on Random Ordered Trees

Introduction





bfsScore(v) = the number of steps before reaching v when using BFS.

totalBfsScore(T,l) = 15 (l=2)
totalB(n,l) = sum over all trees.

dfsScore(v) = the number of steps before reaching v when using DFS.

totalDfsScore(T,l) = 13
totalD(n,l) = sum over all trees.

Question

When searching for a node x at level l, which of the two algorithms has a better expected performance? Formally, is it true that…

$$\mathop{\mathbb{E}}_{\substack{x \in \text{lev}(T,l) \\ T \in \mathcal{T}_n}} (\text{bfsScore}(x)) \geq \mathop{\mathbb{E}}_{\substack{x \in \text{lev}(T,l) \\ T \in \mathcal{T}_n}} (\text{dfsScore}(x))$$

Computing totalB - A recursive generating function

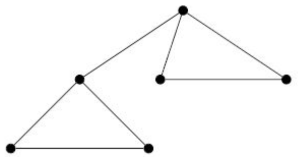
$$F_l(x, y, z) = \sum_{k,m,n} c_{k,m,n} x^k y^m z^n$$

$c_{k,m,n}$ is the number of trees with n nodes that have k nodes at levels smaller than l and m nodes at level l.

Can be generated **recursively** by the following formulae

$$F_0 = y\left(\sum_{n\geq 0} C_n z^n\right) = \frac{2y}{1+\sqrt{1-4z}}$$

$$F_l = \frac{x}{1 - zF_{l-1}}$$



Computing totalB - A formula using k and m

$$F_l(x, y, z) = \sum_{k,m,n} c_{k,m,n} x^k y^m z^n$$

$c_{k,m,n}$ is the number of trees with n nodes that have k nodes at levels smaller than l and m nodes at level l.

The totalBfsScore of one of these trees (at level l) would be:

$$k + (k+1) + \cdots + (k+m) = km + \frac{m(m-1)}{2}$$

Therefore,

$$totalB(n, l) = \sum_{k,m} c_{k,m,n} \cdot \left(km + \frac{m(m-1)}{2}\right)$$

Computing totalB - Using derivatives

One can manipulate $F_l(x,y,z)$ to obtain a generating function for $totalB$

$$\left.\frac{\partial^2 F_l}{\partial x \partial y} + \frac{1}{2}\cdot\frac{\partial^2 F_l}{\partial^2 y}\right|_{x=y=1} = \sum_n totalB(n,l)\cdot z^n$$

Out[•]= $z + 4z^2 + 14z^3 + 48z^4 + 165z^5 + 572z^6 + 2\,002z^7 + 7\,072z^8 + O[z]^9$ (l=1)

Out[•]= $2z^2 + 13z^3 + 62z^4 + 264z^5 + 1\,066z^6 + 4\,186z^7 + 16\,184z^8 + O[z]^9$ (l=2)

Out[•]= $3z^3 + 26z^4 + 153z^5 + 766z^6 + 3\,520z^7 + 15\,368z^8 + O[z]^9$ (l=3)

Out[•]= $4z^4 + 43z^5 + 298z^6 + 1\,699z^7 + 8\,686z^8 + O[z]^9$ (l=4)

Computing totalB - A match in the OEIS

The sequence obtained in l=1, corresponds to sequence A002057 of the OEIS.

A002057 Fourth convolution of Catalan numbers: 4*binomial(2n+3,n)/(n+4). +40
(Formerly M3483 N1415) 66

1, 4, 14, 48, 165, 572, 2002, 7072, 25194, 90440, 326876, 1188640, 4345965, 15967980, 58929450, 218349120, 811985790, 3029594040, 11338026180, 42550029600, 160094486370, 603784920024, 2282138106804, 8643460269248, 32798844771700, 124680849918352 (list; graph; refs; listen; history; text; internal format)

OFFSET 0,2

COMMENTS a(n) is sum of the (flattened) list obtained by the iteration of: replace each integer k with the list 0,...,k+1 on the starting value 0. Length of this list is Catalan(n) or A000108. - Wouter Meeussen, Nov 11 2001

Furthermore, we can describe a relationship between the interpretation by Wouter Meeussen and the planar trees, which gives us insights on why this is the resulting sequence for the level 1.

Computing totalB - Analogy to nested list

a(n) is sum of the (flattened) list obtained by the iteration of: replace each integer k with the list 0,...,k+1 on the starting value 0. Length of this list is Catalan(n) or A000108. - Wouter Meeussen, Nov 11 2001

n=0 → [0]

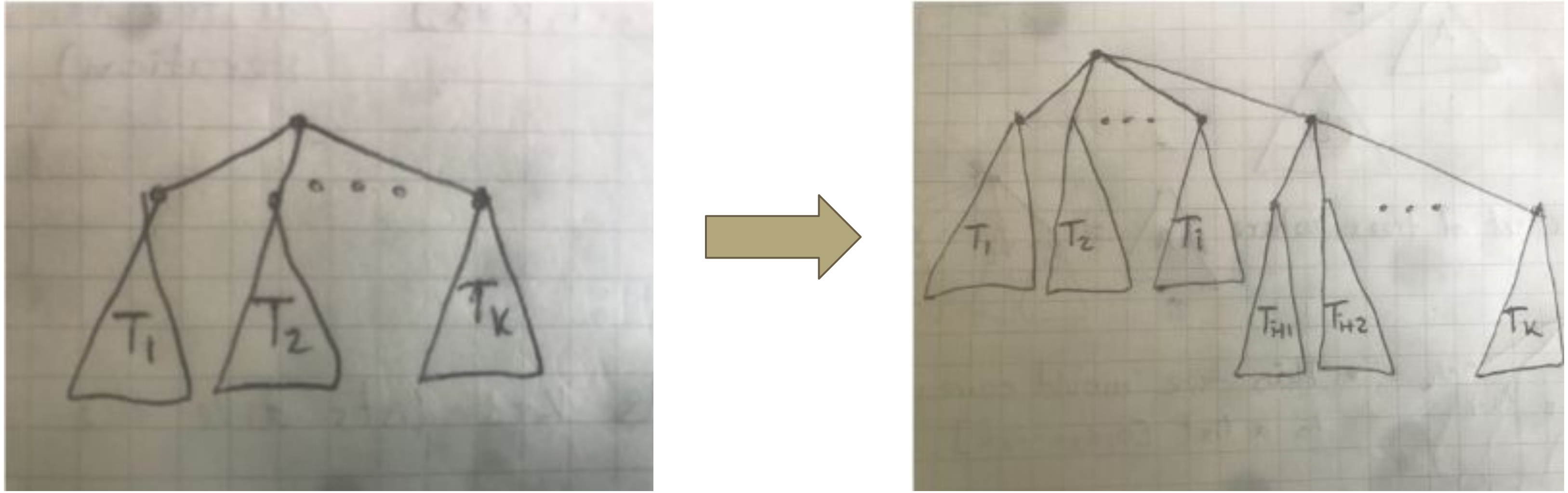
n=1 → [[0,1]]

n=2 → [[[0,1],[0,1,2]]]

n=3 → [[[[0,1],[0,1,2]],[[0,1],[0,1,2],[0,1,2,3]]]]

We will show that we can match every minimal list [0, ..., i] in the nth iteration to a unique tree with n edges whose root has i children.

Computing totalB - Analogy to nested list



Computing totalB - Other observations

Going back to the generating function for $totalB$...

In[∘]:= `Simplify[TotalB[1]]`

Out[∘]= $$\frac{2\left(1+\sqrt{1-4z}\right)^2 z}{\left(1+\sqrt{1-4z}-2z\right)^3}$$

In[∘]:= `Simplify[TotalB[2]]`

Out[∘]= $$-\frac{4\left(1+\sqrt{1-4z}\right)\left(1+\sqrt{1-4z}-z\right)z^2}{\left(-1-\sqrt{1-4z}+\left(3+\sqrt{1-4z}\right)z\right)^3}$$

In[∘]:= `Simplify[TotalB[3]]`

Out[∘]= $$\frac{4z^3\left(3\left(1+\sqrt{1-4z}\right)-2\left(5+2\sqrt{1-4z}\right)z+4z^2\right)}{\left(1+\sqrt{1-4z}-2\left(2+\sqrt{1-4z}\right)z+2z^2\right)^3}$$

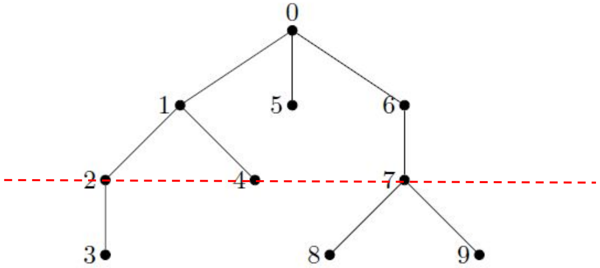
In[∘]:= `Simplify[TotalB[4]]`

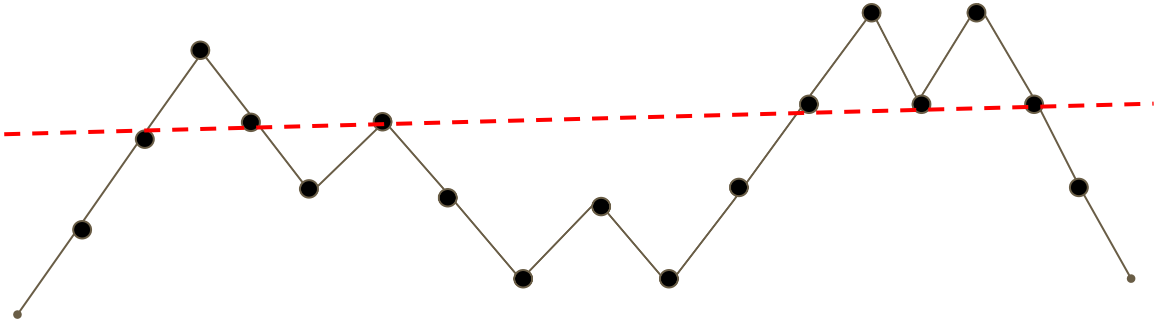
Out[∘]= $$-\frac{4\left(1+\sqrt{1-4z}\right)z^4\left(-2\left(1+\sqrt{1-4z}\right)+3\left(2+\sqrt{1-4z}\right)z-2z^2\right)}{\left(1+\sqrt{1-4z}-\left(5+3\sqrt{1-4z}\right)z+\left(5+\sqrt{1-4z}\right)z^2\right)^3}$$

In[∘]:= `Simplify[TotalB[l]]`

Out[∘]= $$\frac{16\left(1-\sqrt{1-4z}\right)^{-1+l}\left(1+\sqrt{1-4z}\right)^{-1-2l}\sqrt{1-4z}\,z\left(\left(\left(1-\sqrt{1-4z}\right)^l-\left(1+\sqrt{1-4z}\right)^l\right)\left(1+\sqrt{1-4z}\right)z-l\left(1+\sqrt{1-4z}\right)^l\left(-1-\sqrt{1-4z}+2\left(2+\sqrt{1-4z}\right)z\right)\right)}{\left(1+\sqrt{1-4z}-4z\right)^3}$$

Computing totalD - Using the bijection between plane trees and Dyck paths





totalDfsScore(T, l) will be exactly the number of steps in all prefixes of the Dyck path corresponding to T , ending by an ‘up’ step at level l.

$$totalD(n,l) = \sum_{j \geq l}^{n} j \cdot [(0,0) \longrightarrow (i, j-1)] \cdot [(i,j) \longrightarrow (n,n)]$$

$$[(0,0) \longrightarrow (i,j)] = \binom{j+i+1}{i} \frac{j-i+1}{j+i+1}$$

Computing totalD - Analysing the formula

Rewriting this expression in terms of j and l gives us

$$\sum_{j \geq l}^{n} j \cdot [(0,0) \longrightarrow (j-l, j-1)] \cdot [(j-l, j) \longrightarrow (n,n)].$$

By symmetry, this is equal to

$$\sum_{j \geq l}^{n} j \cdot [(0,0) \longrightarrow (j-l, j-1)] \cdot [(0,0) \longrightarrow (n-j, n-j+l)]$$

$$= \sum_{j \geq l}^{n} j \cdot \binom{2j-l}{j} \frac{l}{2j-l} \binom{2n-2j+l+1}{n-j} \frac{l+1}{2n-2j+l+1}.$$

Now, we can rewrite this as

$$l \cdot \sum_{j \geq l}^{n} \binom{2j-l-1}{j-1} \binom{2n-2j+l+1}{n-j} \frac{l+1}{2n-2j+l+1}.$$

This expression resembles the following identity from Eq. 26 of Section 1.2.6 of [2]

$$\sum_{k=0}^{m} \binom{r-kt}{k} \frac{r}{r-kt} \binom{s-mt+kt}{m-k} = \binom{r+s-mt}{m}$$

Replacing $k = n-j, m = n-1, r = l+1, s = 1-l, t = -2$ in this identity gives us:

$$l \cdot \sum_{j=1}^{n} \binom{2j-l-1}{j-1} \binom{2n-2j+l+1}{n-j} \frac{l+1}{2n-2j+l+1} = l \binom{2n}{n-1}.$$

Inspired by a similar derivation found in the appendix of [1]

Computing totalD - Our conjecture

From this, we get a closed form formula for l=1: $\binom{2n}{n-1}$

We can also compute closed form formula for bigger l, by subtracting the first few terms. We have done this by hand for l=2, 3, 4, 5 and we have obtained

$$l \cdot \binom{2n}{n-l}$$

We conjecture that this is the formula for any l.

Computing totalD - Attempts at proving the conjecture

need to show that

$$\sum_{j=1}^{l-1} \binom{2j-l-1}{j-1} \binom{2n-2j+l+1}{n-j} \frac{l+1}{2n-2j+l+1} = \binom{2n}{n-1} - \binom{2n}{n-l}.$$

The expression on the right suggests using the reflection principle for counting certain Dyck paths.

- Reflection principle
- Induction
- Telescopic sum

Further questions

II. Further questions

(1) What if I know the value of l and n in advance? Assuming that I can switch between BFS and DFS at every node, but not just to pick one of these algorithms at the beginning, then what is the optimal algorithm giving me the smallest expected number of steps before reaching the target?

(2) Assume I do not know the exact level l, but just that l is in some interval $[u, v]$. What is the optimal algorithm and the expected number of steps to reach the target node, then?

(3) What if I have two randomly selected target nodes at a given level? What will be the expected number of steps to reach one of them?

References

REFERENCES

[1] Dershowitz, N. and Zaks, S., 1980. Enumerations of ordered trees. Discret. Math., 31(1), pp.9-28.
[2] Knuth, D.E., 2005. The Art of Computer Programming, Volume 1, Fascicle 1: MMIX–A RISC Computer for the New Millennium. Addison-Wesley Professional.