



UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

Facultat d'Informàtica de Barcelona



ADVANCED DATA STRUCTURES

COMPUTER SCIENCE DEPARTMENT

Random Binary Search Trees: An empirical analysis

Author:

Alex Herrero

Professor:

Amalia Duch

March 9, 2025



Contents

1	Introduction	3
2	Analysis of the Average Cost of Insertions	3
2.1	Theoretical Study	3
2.2	Experimental Study	4
3	Analysis of Internal Path Length	7
3.1	Theoretical Study	7
3.2	Experimental Study	8
4	Analysis of Interleaver Insertions and Deletions	11
A	More accurate solutions to the previous recurrences	13

List of Figures

1	Plot average number of nodes visited respect theoretical bound	5
2	Plot average IPL respect the theoretical bound	9
3	Plot experimental result divided by $n \log n$	9
4	Plot average IPL with and without alternating insertions and deletions	11

List of Tables

1	Experimental vs Theoretical results on the expected cost of insertion	6
2	Experimental vs Theoretical results on the expected Internal Path Length	10
3	Difference of IPL after doing deletions	12

1 Introduction

Let T be a binary tree with subtrees T_l and T_r . We say that T is a *binary search tree* (BST) if it is either an empty binary tree or it contains at least one element x as its root such that

- T_l and T_r are also BSTs.
- $\forall y \in T_l, y < x$ and $\forall z \in T_r, z > x$.

Although it is well known that, in the worst case, a BST behaves like a linked list (with the height of the tree being $\Theta(n)$), in this report, we focus on *random BSTs* of size n .

By *random BSTs*, we mean the following: Given a universe of keys U with $|U| = n$, we construct the BST by inserting each element of U exactly once, choosing the insertion order uniformly at random.

2 Analysis of the Average Cost of Insertions

2.1 Theoretical Study

Let us first analyze the expected cost of inserting an element $u \in U$ into our BST T . For that, we will consider this cost as the cost of searching for u in our BST, which is valid since we can assume that, if u does not exist in T , our search terminates in any empty subtree with identical probability.

Let I_n be the expected cost of the insertion of a key x in a random BST of size n . Let, also, be $I_{n,q}$ be the expected cost of the insertion of a key x in a random BST which root is the q -th smallest element. Then, the expected cost of I_n is

$$\begin{aligned}
 I_n &= \frac{1}{n} \sum_{q=1}^n I_{n,q} \\
 &= 1 + \frac{1}{n} \sum_{k=1}^n \left(\frac{1}{n} 0 + \frac{k-1}{n} I_{k-1} + \frac{n-k}{n} I_{n-k} \right) \\
 &= 1 + \frac{1}{n} \sum_{k=0}^{n-1} \left(\frac{k}{n} I_k + \frac{n-k-1}{n} I_{n-k-1} \right) \\
 &= 1 + \frac{1}{n^2} \sum_{k=0}^{n-1} (k I_k + (n-k-1) I_{n-k-1}) \\
 &= 1 + \frac{2}{n^2} \sum_{k=0}^{n-1} k I_k
 \end{aligned}$$

We can solve this recurrence using the continuous master theorem. The continuous master theorem solves recurrences of the form

$$F_n = t_n + \sum_{0 \leq j < n} w_{n,j} F_j$$

with $t_n = \Theta(n^a (\log n)^b)$. We proceed as follows:

- Determine the values of a and b : Since $t_n = \Theta(1)$, it is straightforward to see that $a = b = 0$.
- Provide a shape function for the weights $w_{n,j}$: We use the following trick to determine the shape function:

$$w(z) = \lim_{n \rightarrow \infty} n \cdot w_{n,z \cdot n} = n \cdot \frac{2zn}{n^2} = 2z.$$

- Determine the value of

$$\mathcal{H} = 1 - \int_0^1 w(z) z^a dz.$$

Substituting the values, we obtain:

$$\mathcal{H} = 1 - \int_0^1 2z dz = 1 - (1 - 0) = 0.$$

- Since $\mathcal{H} = 0$, we need to compute

$$\mathcal{H}' = -(b+1) \int_0^1 w(z) z^a \ln z dz.$$

Substituting the known values,

$$\mathcal{H}' = -1 \int_0^1 2z \ln z dz.$$

This integral can be solved using integration by parts. For the purpose of applying the theorem, we skip the detailed calculation, giving the result:

$$\mathcal{H}' = -\left(x^2 \ln x - \frac{x^2}{2}\right) \Big|_0^1 = \frac{1}{2}.$$

Since $\mathcal{H} = 0$ and $\mathcal{H}' \neq 0$, we use the result

$$F_n = \frac{t_n}{\mathcal{H}'} \ln n + o(t_n \log n).$$

Substituting the values, we obtain

$$I_n = 2 \ln n + o(\log n).$$

Thus, the expected cost of an insertion into a random binary search tree is bounded by $O(\log n)$.

2.2 Experimental Study

Once we have theoretical results on the expected cost of an insertion in a random BST, we can provide experimental results to assess how closely they match the theoretical predictions. For this, we will conduct the following experiment:

1. We create a random BST of size n by generating n random keys in the interval $[0, 1]$.
2. After constructing the BST, we generate $q = 2 \cdot n$ random numbers in the interval $[0, 1]$.
3. For each generated value, we perform a `find` operation in the BST, counting the number of nodes traversed during the operation.
4. We sum up the total number of nodes traversed across all q search operations and compute the average.
5. We repeat all previous steps with 20 different seeds and compute the final average.

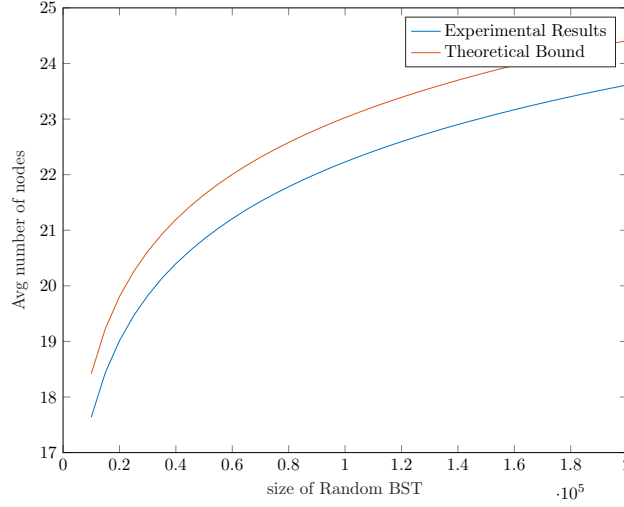


Figure 1: Plot average number of nodes visited respect theoretical bound

6. We repeat the entire experiment for different values of n .

I conducted the experiment previously explained with values of n ranging from 10,000 to 200,000 in increments of 5,000, using seeds to generate random numbers from 1989 (in honor of the year of the first publication of the book *Introduction to Algorithms*, which was a great resource for refreshing my knowledge of BSTs and expected cost) to 2008. Figure 1 provides a plot of the values obtained from this experiment, as well as the theoretical bound derived using the continuous master theorem, indicating that the theoretical bound always bounds the experimental results. In fact, the shapes of both plots appear to be the same, suggesting that the only difference between the two functions is a constant factor. Indeed, by applying the continuous master theorem, we know that this difference is $o(\log n)$, but a more detailed analysis, presented in Appendix CITAR, shows that this difference is actually $O(1)$.

This constant factor is clearly visible in Table 1, where the column *Difference* (computed as the difference between the theoretical value obtained using Roura's theorem and the experimental value) consistently yields basically a constant difference.

n	Experimental	Theoretical	Difference
10000	17.6373	18.4207	0.78335
15000	18.4401	19.2316	0.79148
20000	19.0132	19.807	0.79379
25000	19.4589	20.2533	0.79437
30000	19.8232	20.6179	0.79466
35000	20.1327	20.9262	0.79355
40000	20.3994	21.1933	0.79389
45000	20.6318	21.4288	0.79707
50000	20.8435	21.6396	0.79606
55000	21.0332	21.8302	0.79698
60000	21.2081	22.0042	0.79606
65000	21.3695	22.1643	0.79484
70000	21.5175	22.3125	0.79501
75000	21.6545	22.4505	0.79602
80000	21.7826	22.5796	0.79696
85000	21.9042	22.7008	0.79657
90000	22.0176	22.8151	0.79753
95000	22.1266	22.9233	0.79666
100000	22.2288	23.0259	0.7971

n	Experimental	Theoretical	Difference
105000	22.3268	23.1234	0.79667
110000	22.4196	23.2165	0.79691
115000	22.5088	23.3054	0.79659
120000	22.5934	23.3905	0.79713
125000	22.6754	23.4721	0.7967
130000	22.7532	23.5506	0.79738
135000	22.8291	23.6261	0.79692
140000	22.901	23.6988	0.7978
145000	22.9706	23.769	0.79836
150000	23.0382	23.8368	0.79857
155000	23.1038	23.9024	0.79857
160000	23.1678	23.9659	0.79808
165000	23.2297	24.0274	0.79766
170000	23.2888	24.0871	0.79832
175000	23.3468	24.1451	0.79826
180000	23.4032	24.2014	0.79818
185000	23.4583	24.2562	0.79793
190000	23.5112	24.3096	0.79838
195000	23.5627	24.3615	0.79878
200000	23.6128	24.4121	0.79932

Table 1: Experimental vs Theoretical results on the expected cost of insertion

3 Analysis of Internal Path Length

3.1 Theoretical Study

Let T be a BST of size n with root r . We define the *Internal Path Length* of T as the sum of all distances between every node of the tree and the root. More formally:

$$\text{IPL}(T) = \sum_{v \in V(T)} d(r, v).$$

This metric should not be surprising, as we can estimate the distance between the root and every node by averaging the IPL of such a tree. Before establishing this relationship, let us first derive a recurrence for the IPL of a tree, solve this recurrence, and later analyze the obtained relation.

Under the assumption of having a random BST, given n keys, we consider every permutation among the $n!$ possible ones to be equally likely. Therefore, the sizes of the subtrees T_l and T_r will be completely random, provided that their combined sizes sum to $n - 1$. Consequently, in our recurrence, we must consider every possible size distribution of the subtrees as equally likely.

Moreover, given the IPL of T_l and T_r , since we introduce a new root at the top of the tree, every node will need to traverse one additional edge to reach the new root. This results in adding 1 to the previous distances with respect to their original root. As a consequence, we must add the number of edges in the tree to the sum of the previous IPLs of both subtrees.

Hence, creating the following recurrence:

$$\begin{aligned} \text{IPL}_n &= n - 1 + \frac{1}{n} \sum_{k=0}^{n-1} \text{IPL}_k + \text{IPL}_{n-1-k} \\ &= n - 1 + \frac{1}{n} \sum_{k=0}^{n-1} 2 \cdot \text{IPL}_k \\ &= n - 1 + \frac{2}{n} \sum_{k=0}^{n-1} \text{IPL}_k \end{aligned}$$

We can, again, use the continuous master theorem in order to solve this recurrence. Again we need to determinate the following values:

- Determine the values of a and b : Since $t_n = \Theta(n)$, it is straightforward to see that $a = 1$ and $b = 0$.
- Provide a shape function for the weights $w_{n,j}$: We use the following trick to determine the shape function:

$$w(z) = \lim_{n \rightarrow \infty} n \cdot w_{n,z \cdot n} = n \cdot \frac{2}{n} = 2.$$

- Determine the value of

$$\mathcal{H} = 1 - \int_0^1 w(z) z^a dz.$$

Substituting the values, we obtain:

$$\mathcal{H} = 1 - \int_0^1 2z dz = 1 - (1 - 0) = 0$$

- Since $\mathcal{H} < 0$ we are in the case

$$\mathcal{H}' = -(b+1) \int_0^1 w(z) z^a \ln z dz.$$

Substituting the known values,

$$\mathcal{H}' = -1 \int_0^1 2z \ln z dz.$$

This integral can be solved using integration by parts. For the purpose of applying the theorem, we skip the detailed calculation, giving the result:

$$\mathcal{H}' = -(x^2 \ln x - \frac{x^2}{2}) \Big|_0^1 = \frac{1}{2}.$$

Since $\mathcal{H} = 0$ and $\mathcal{H}' \neq 0$, we use the result

$$F_n = \frac{t_n}{\mathcal{H}'} \ln n + o(t_n \log n).$$

Substituting the values, we obtain

$$I_n = 2(n-1) \ln n + o(n \log n) = 2n \ln n + o(n \log n)$$

Thus, the expected internal path length in a BST is bounded by $O(n \log n)$. Such recurrence can be used to solve the expected cost of doing an insertion in a BST as we can express that cost as: $I_n = 1 + \frac{IPL_n}{n}$ (as we can have an estimation of I_n by averaging the internal path length over the size of the tree in order to get an estimation of the average distance between the root and a node) which would give a result of $O(\log n)$ which was the expected bound we obtained in the previous section.

3.2 Experimental Study

Once we have theoretical results on the expected internal path length in a random BST, we can provide experimental results to assess how closely they match the theoretical predictions. For this, we will conduct the following experiment:

1. We create a random BST of size n by generating n random keys in the interval $[0, 1]$.
2. We calculate the internal path length of the random BST by doing a Breadth-first search algorithm
3. We repeat all previous steps with 20 different seeds and compute the final average.
4. We repeat the entire experiment for different values of n .

Again, I conducted the experiment with the same value of n and the same *seeds* as before. Figure 2 presents, as before, a plot comparing the expected values obtained from the continuous master theorem result with the experimental results. This time, however, the two curves do not exhibit similar behavior as observed previously. It appears that the theoretical bound increases faster than the experimental

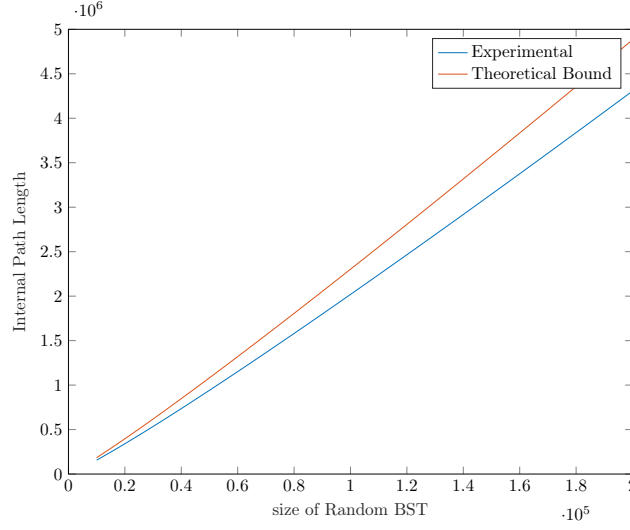


Figure 2: Plot average IPL respect the theoretical bound

result, unlike the previous case. A more numerical perspective is provided in Table 2, where we observe that the difference is no longer constant. In fact, the discrepancy increases as the input size grows.

We know that there is a factor $o(n \log n)$ (in fact, with a more detailed solution of the recurrence, see Appendix CITAR, we know that this $o(n \log n)$ is $O(n)$) that must be taken into account in our study. As a result, even with a high number of nodes in a random BST, this theoretical bound may still be influenced by the $o(n \log n)$ factor until we reach a significantly larger number of nodes. To analyze this effect, we provide Figure 3, where we divide the experimental data by $n \log n$ to observe when this $o(n \log n)$ factor stabilizes. As seen in the figure, even for large input sizes (over 200,000 nodes), this factor still plays a significant role in our analysis.

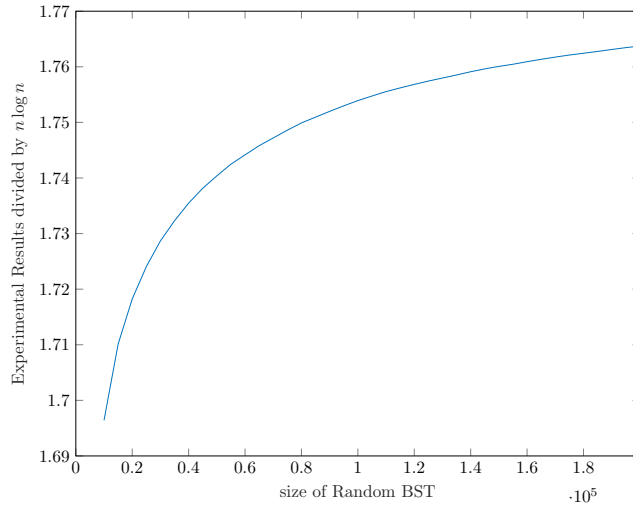


Figure 3: Plot experimental result divided by $n \log n$

So, that $o(n \log n)$ factor must be the reason why it is difficult to predict a more accurate theoretical solution to the IPL: because this factor is very influential even in instances with more than 100,000 nodes, and we need to reach an even larger size of random BSTs to make this factor smaller than the actual bound.

n	Experimental	Theoretical	Difference
10000	156245.35	184206.8074	27961.4574
15000	246662.85	288474.1644	41811.3144
20000	340332.35	396139.5021	55807.1521
25000	436473.15	506331.5552	69858.4052
30000	534628.65	618537.1596	83908.5096
35000	634383.5	732417.2338	98033.7338
40000	735610.7	847730.7786	112120.0786
45000	838045.5	964297.5992	126252.0992
50000	941527.05	1081977.8284	140450.7784
55000	1046067.1	1200659.7311	154592.6311
60000	1151365.35	1320251.9809	168886.6309
65000	1257575.2	1440678.5314	183103.3314
70000	1364456.25	1561875.0729	197418.8229
75000	1472133.3	1683786.5089	211653.2089
80000	1580482.2	1806365.1062	225882.9062
85000	1689287.05	1929569.111	240282.061
90000	1798726	2053361.6909	254635.6909
95000	1908744.3	2177710.1124	268965.8124
100000	2019274.8	2302585.093	283310.293

n	Experimental	Theoretical	Difference
105000	2130221.85	2427960.2821	297738.4321
110000	2241658.35	2553811.8419	312153.4919
115000	2353417.6	2680118.1037	326700.5037
120000	2465595.3	2806859.2852	341263.9852
125000	2578168	2934017.2541	355849.2541
130000	2691101.95	3061575.3297	370473.3797
135000	2804431.8	3189518.1155	385086.3155
140000	2918219.4	3317831.3564	399611.9564
145000	3032270.35	3446501.8162	414231.4662
150000	3146588.05	3575517.1719	428929.1219
155000	3261131.6	3704865.9227	443734.3227
160000	3376159.9	3834537.3101	458377.4101
165000	3491454.05	3964521.2485	473067.1985
170000	3607009.65	4094808.2635	487798.6135
175000	3722815.9	4225389.4385	502573.5385
180000	3838806.45	4356256.3668	517449.9168
185000	3955127.75	4487401.1085	532273.3585
190000	4071748.65	4618816.1534	547067.5034
195000	4188621.75	4750494.3866	561872.6366
200000	4305674.95	4882429.0582	576754.1082

Table 2: Experimental vs Theoretical results on the expected Internal Path Length

4 Analysis of Interleaver Insertions and Deletions

Knuth, in Volume 3 of his well-known book *The Art of Computer Programming* [Knu98], mentioned that random insertions and deletions indeed *destroy* the randomness of a BST. This phenomenon was first observed by Gary Knott in 1972, and later, Eppinger [Epp83] demonstrated experimentally that the path length, although initially decreasing slightly, eventually increases and stabilizes after performing a quadratic number of deletions and insertions.

The following experiment aims to support the observations made by Eppinger and Knuth:

1. We create a random BST of size n by generating n random keys in the interval $[0, 1]$.
2. We perform a quadratic number of insertions and deletions as follows: we alternately insert a random element from the interval $[0, 1]$ and delete a random element from the BST.
3. We compute the internal path length of the BST using a Breadth-First Search algorithm.
4. We repeat all the previous steps with 20 different seeds and compute the final average.
5. We repeat the entire experiment for different values of n .

This time, as we are doing a quadratic number of steps, because of memory and time performance I have decided to conduct this experiment with Random BSTs of size 1000 to 2000 in increases of 50 elements. Figure 4 provides a plot of the final IPL with and without doing this alternation between deletions and insertion. As expected, we see that, indeed, the IPL of a tree increases if we perform this operations, indicating that we are unbalancing our random BST (hence, destroying our randomness).

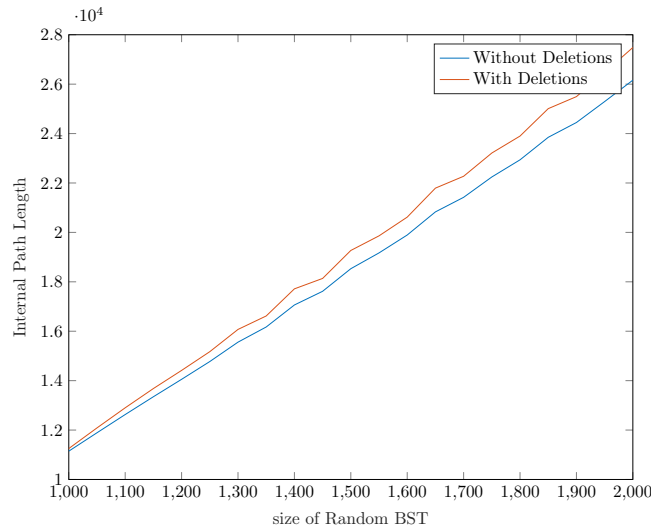


Figure 4: Plot average IPL with and without alternating insertions and deletions

Table 3 provides a more numerical view of such plot. As we see, the difference in IPL gets bigger as we increase the size of n , definitely destroying the initial randomness that we created.

n	No Deletions	Deletions	Difference
1000	11145.85	11252.15	106.3
1050	11890.2	12086.95	196.75
1100	12627.65	12900.5	272.85
1150	13348.65	13680.2	331.55
1200	14053.475	14414.25	360.775
1250	14773.025	15172.7	399.675
1300	15557.95	16072.8	514.85
1350	16171.5	16617.6	446.1
1400	17062.5	17715.25	652.75
1450	17615.975	18136.7	520.725
1500	18530.425	19268.7	738.275
1550	19169.875	19860.2	690.325
1600	19894.3	20618.35	724.05
1650	20830.225	21792	961.775
1700	21419.875	22272.1	852.225
1750	22243.3	23213	969.7
1800	22937.55	23899.25	961.7
1850	23848.45	25009.05	1160.6
1900	24447.525	25495.45	1047.925
1950	25295.475	26473.65	1178.175
2000	26161.125	27483.5	1322.375

Table 3: Difference of IPL after doing deletions

A More accurate solutions to the previous recurrences

Let us consider the previous recurrence for the IPL of a Random BST:

$$I_n = n - 1 + \frac{2}{n} \sum_{k=0}^{n-1} I_k$$

We have to try to put this recurrence in terms of just one call instead of multiple calls of it. For that we need to firstly see that:

$$I_{n+1} = n + \frac{2}{n+1} \sum_{k=0}^n I_k$$

Now we need to subtract carefully both recurrences. For that, let us first multiply each recurrence by n and $n+1$:

$$\begin{aligned} (n+1)I_{n+1} &= (n+1)n + 2 \sum_{k=0}^n I_k \\ nI_n &= n^2 - n + 2 \sum_{k=0}^{n-1} I_k \end{aligned}$$

Now we are ready to subtract both recurrences:

$$\begin{aligned} (n+1)IPL_{n+1} - nIPL_n &= (n+1)n + 2IPL_n - n^2 + n \\ (n+1)IPL_{n+1} &= n^2 + n + 2IPL_n - n^2 + n + nIPL_n \\ (n+1)IPL_{n+1} &= 2n + 2IPL_n + nIPL_n = 2n + (2+n)IPL_n \\ IPL_{n+1} &= \frac{2n}{n+1} + \frac{2+n}{n+1} IPL_n \\ &= \frac{2n}{n+1} + \frac{2(2+n)(n-1)}{(n+1)n} + \frac{(2+n)(n+1)}{(n+1)n} IPL_{n-1} \\ &= \frac{2n}{n+1} + \frac{2(2+n)(n-1)}{(n+1)n} + \frac{2+n}{n} \left(\frac{2(n-2)}{n-1} + \frac{n}{n-1} IPL_{n-2} \right) \\ &= \frac{2n}{n+1} + 2(n+2) \sum_{i=1}^n \frac{n-i}{(n-i+1)(n-i+2)} \\ &= \frac{2n}{n+1} + 2(n+2) \sum_{i=1}^n \frac{i}{(i+1)(i+2)} \\ &= \frac{2n}{n+1} + 2(n+2) \sum_{i=1}^n \frac{2}{i+2} - \frac{1}{i+1} \\ &= \frac{2n}{n+1} + 2(n+2) \left(\frac{2}{n+2} + \frac{1}{n+1} + H_n - 2 \right) \\ &= 2nH_n - 4n + 4H_n + O(1) \end{aligned}$$

We know that harmonic numbers grow similarly to the natural logarithm, so $H_n = \ln n + O(1)$. Hence, $I_n = 2n \ln n + O(n)$. Using the IPL, we can estimate the average number of nodes required for an insertion by averaging the IPL over the number of nodes. This gives the following estimation:

$$I_n = 1 + \frac{IPL_n}{n} = 2 \ln n + O(1).$$

Special thanks to Conrado's slides on Data Structures and Algorithms, which, along with the book *Introduction to Algorithms*, were a great source for refreshing my knowledge and assisting me when I struggled with certain calculations!

References

- [Epp83] Jeffrey L Eppinger. An empirical study of insertion and deletion in binary search trees. *Communications of the ACM*, 26(9):663–669, 1983.
- [Knu98] Donald E Knuth. *The Art of Computer Programming: Sorting and Searching, volume 3*. Addison-Wesley Professional, 1998.