JJ McCauley – AVL & BST + DSW Timing Report

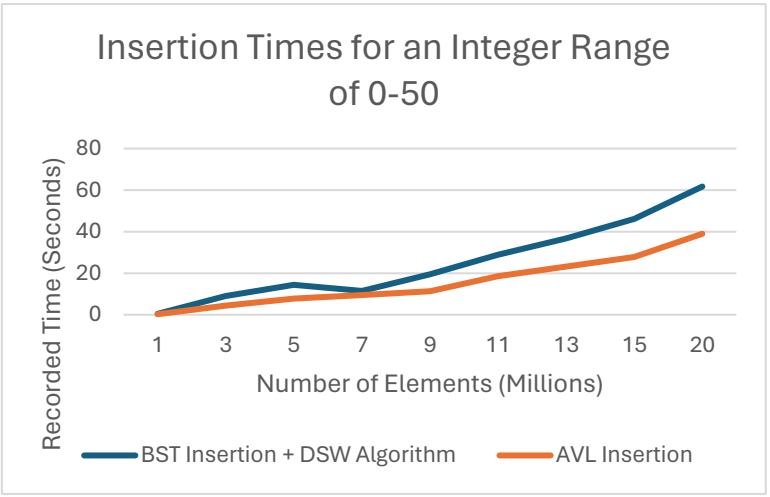
This report aims to compare the timing efficiency of two algorithms used for inserting elements into a balanced tree: AVL Insertion and BST Insertion + the DSW Algorithm.

Methods

For data collection, arrays of random integers were tested at consistent sizes for three different integer ranges. Times were recorded using the Chronos library and arrays were assigned randomly. Each dataset was passed into Desmos to find the best curve function. The most representative equation (determined by the r^2 statistic) is outlined in green.

Element Range: 0-50

Size	BST Insertion + DSW Algorithm	AVL Insertion
1	0.441255	0.246517
3	8.99175	4.41296
5	14.3483	7.75063
7	11.4223	9.45838
9	19.4805	11.2682
11	28.8773	18.6191
13	36.8074	23.0997
15	46.0891	27.8351
20	61.7129	38.9801



BST + DSW Curves

$y_1 \sim ax_1 \cdot \log(x_1) + bx_1 + c$
STATISTICS $R^2 = 0.978$ RESIDUALS e_1 plot
PARAMETERS $a = 2.33904$, $b = -0.0327657$, $c = 2.52013$

$y_1 \sim ax_1 \cdot \log(x_1) + b$
STATISTICS $R^2 = 0.978$ RESIDUALS e_2 plot
PARAMETERS $a = 2.31561$, $b = 2.44616$

$y_1 \sim a \cdot \log(x_1) + b$
Log Mode
STATISTICS $R^2 = 0.7431$ RESIDUALS e_4 plot
PARAMETERS $a = 42.1424$, $b = -10.2715$

$y_1 \sim ax_1 + b$
STATISTICS $r^2 = 0.966$, $r = 0.9829$ RESIDUALS e_3 plot
PARAMETERS $a = 3.19792$, $b = -4.49498$

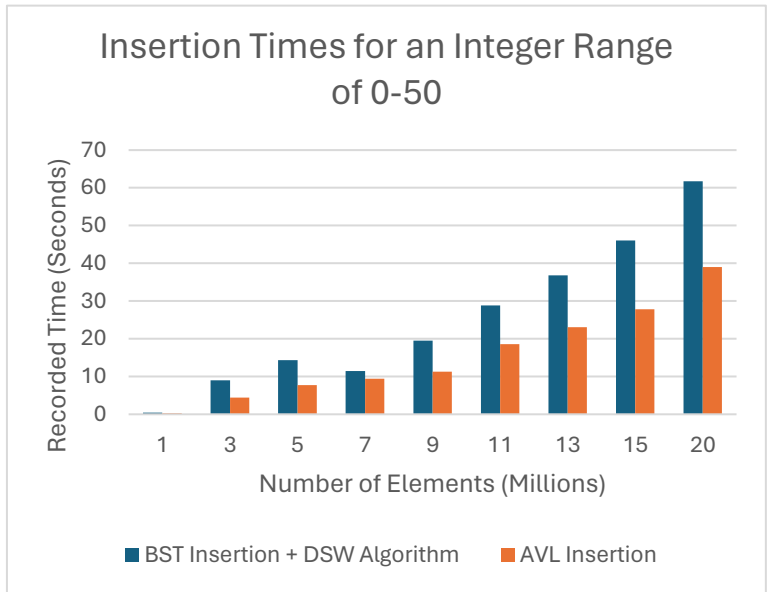
AVL Curves

$y_1 \sim ax_1 \cdot \log(x_1) + bx_1 + c$
STATISTICS $R^2 = 0.9902$ RESIDUALS e_1 plot
PARAMETERS $a = 1.36082$, $b = 0.14691$, $c = 0.908717$

$y_1 \sim ax_1 \cdot \log(x_1) + b$
STATISTICS $R^2 = 0.9901$ RESIDUALS e_2 plot
PARAMETERS $a = 1.4659$, $b = 1.24037$

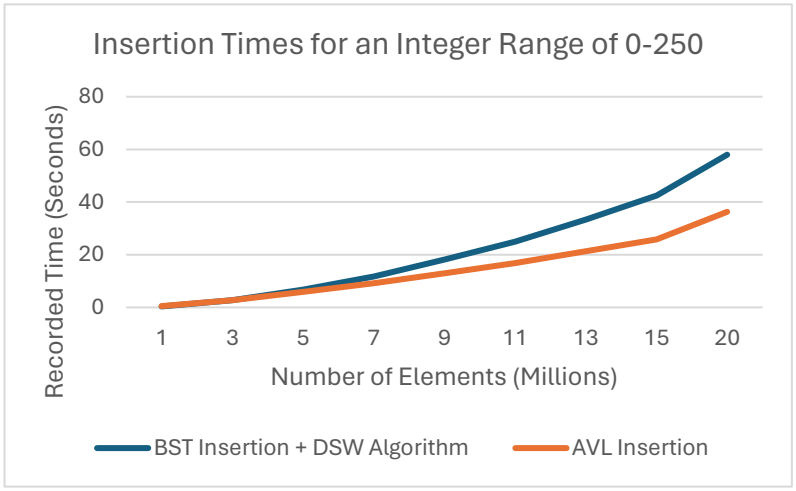
$y_1 \sim a \cdot \log(x_1) + b$
Log Mode
STATISTICS $R^2 = 0.7568$ RESIDUALS e_4 plot
PARAMETERS $a = 26.7592$, $b = -6.87893$

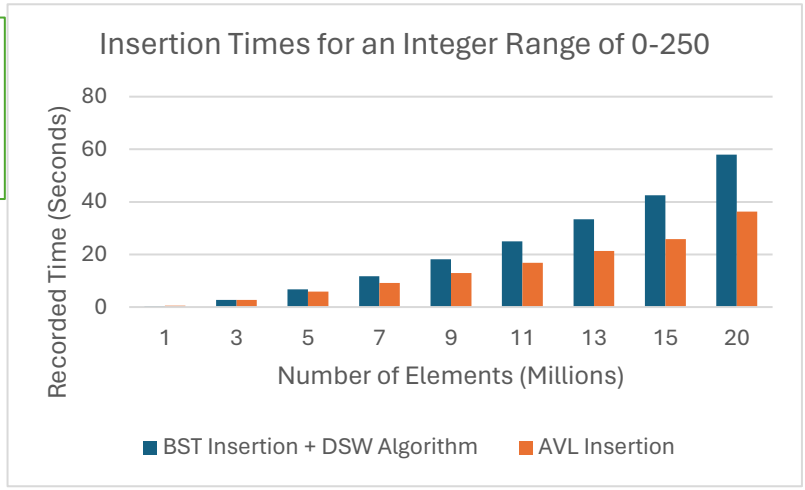
$y_1 \sim ax_1 + b$
STATISTICS $r^2 = 0.9799$, $r = 0.9899$ RESIDUALS e_3 plot
PARAMETERS $a = 2.02647$, $b = -3.17256$



Element Range: 0-250

Size	BST Insertion + DSW Algorithm	AVL Insertion
1	0.367914	0.500963
3	2.69211	2.74646
5	6.73876	5.90567
7	11.7372	9.17462
9	18.2088	12.885
11	25.0047	16.8303
13	33.3145	21.2717
15	42.4818	25.8345
20	57.9844	36.2905





<u>BST + DSW Curves</u>

$y_1 \sim ax_1 \cdot \log(x_1) + bx_1 + c$

STATISTICS $R^2 = 0.996$ RESIDUALS e_1 plot

PARAMETERS
$a = 2.57774$
$b = -0.383743$
$c = 0.141297$

$y_1 \sim ax_1 \cdot \log(x_1) + b$

STATISTICS $R^2 = 0.9958$ RESIDUALS e_2 plot

PARAMETERS
$a = 2.30325$
$b = -0.725019$

$y_1 \sim a \cdot \log(x_1) + b$

Log Mode

STATISTICS $R^2 = 0.731$ RESIDUALS e_4 plot

PARAMETERS
$a = 41.2017$
$b = -12.7698$

$y_1 \sim ax_1 + b$

STATISTICS $r^2 = 0.981$ $r = 0.9905$ RESIDUALS e_3 plot

PARAMETERS
$a = 3.17664$
$b = -7.5897$

<u>AVL Curves</u>

$y_2 \sim ax_1 \cdot \log(x_1) + bx_1 + c$

STATISTICS $R^2 = 0.9996$ RESIDUALS e_1 plot

PARAMETERS
$a = 1.08955$
$b = 0.409569$
$c = 0.00388042$

$y_2 \sim ax_1 \cdot \log(x_1) + b$

STATISTICS $R^2 = 0.9991$ RESIDUALS e_2 plot

PARAMETERS
$a = 1.38251$
$b = 0.928499$

$y_2 \sim a \cdot \log(x_1) + b$

Log Mode

STATISTICS $R^2 = 0.7688$ RESIDUALS e_4 plot

PARAMETERS
$a = 25.3219$
$b = -6.8007$

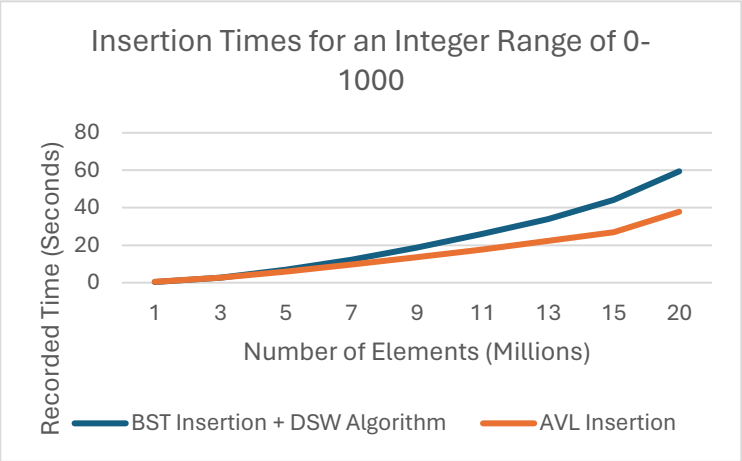
$y_2 \sim ax_1 + b$

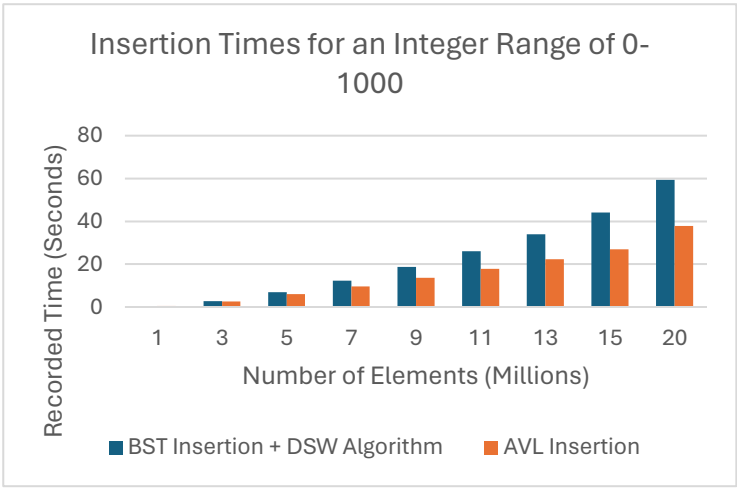
STATISTICS $r^2 = 0.9922$ $r = 0.9961$ RESIDUALS e_3 plot

PARAMETERS
$a = 1.91445$
$b = -3.26382$

Element Range: 0-1000

Size	BST Insertion + DSW Algorithm	AVL Insertion
1	0.342452	0.461769
3	2.68234	2.6298
5	6.91983	5.94892
7	12.3116	9.59533
9	18.7343	13.5857
11	26.089	17.7492
13	34.0026	22.2495
15	44.1409	26.9777
20	59.4248	37.8334





BST + DSW Curves

$y_2 \sim ax_1 \cdot \log(x_1) + bx_1 + c$

STATISTICS
$R^2 = 0.9953$

RESIDUALS
e_1 plot

PARAMETERS
$a = 2.5427$
$b = -0.243888$
$c = -0.137627$

$y_2 \sim ax_1 \cdot \log(x_1) + b$

STATISTICS
$R^2 = 0.9953$

RESIDUALS
e_2 plot

PARAMETERS
$a = 2.36825$
$b = -0.688213$

$y_2 \sim a \cdot \log(x_1) + b$

Log Mode

STATISTICS
$R^2 = 0.7341$

RESIDUALS
e_4 plot

PARAMETERS
$a = 42.4666$
$b = -13.1592$

$y_2 \sim ax_1 + b$

STATISTICS
$r^2 = 0.9815$
$r = 0.9907$

RESIDUALS
e_3 plot

PARAMETERS
$a = 3.26809$
$b = -7.76352$

AVL Curves

$y_2 \sim ax_1 \cdot \log(x_1) + bx_1 + c$

STATISTICS
$R^2 = 0.9996$

RESIDUALS
e_1 plot

PARAMETERS
$a = 1.0748$
$b = 0.523576$
$c = -0.292978$

$y_2 \sim ax_1 \cdot \log(x_1) + b$

STATISTICS
$R^2 = 0.9988$

RESIDUALS
e_2 plot

PARAMETERS
$a = 1.44931$
$b = 0.889017$

$y_2 \sim a \cdot \log(x_1) + b$

Log Mode

STATISTICS
$R^2 = 0.7715$

RESIDUALS
e_4 plot

PARAMETERS
$a = 26.5957$
$b = -7.25616$

$y_2 \sim ax_1 + b$

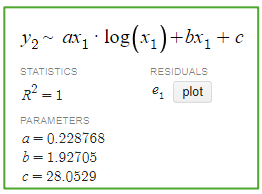
STATISTICS
$r^2 = 0.993$
$r = 0.9965$

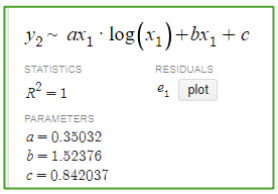
RESIDUALS
e_3 plot

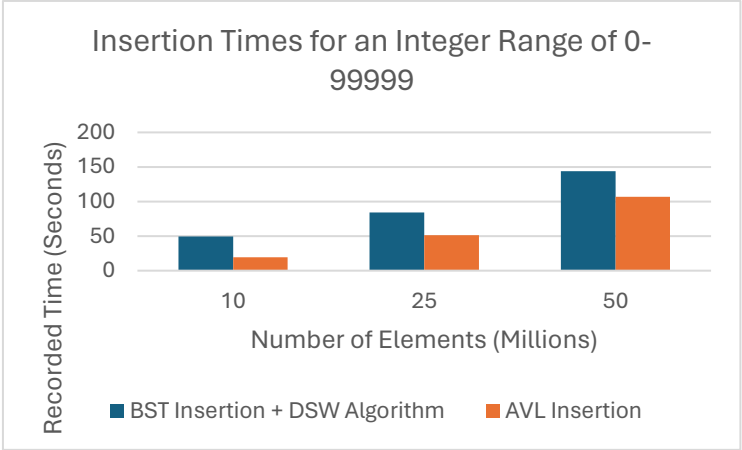
PARAMETERS
$a = 2.00809$
$b = -3.51644$

Additional Testing: **Element Range: 0-99999**

Size	BST Insertion + DSW Algorithm	AVL Insertion
10	49.6111	19.5828
25	84.2243	51.1791
50	143.839	106.789







Observations

When observing these two methods, it is clear that AVL Insertion is significantly faster than BST Insertion + the DSW Algorithm. When looking at the data, the AVL tree consistently outperforms the Binary Search Tree, being at least 20 seconds faster during the largest array size for each of the samples. Although it consistently outperforms the BST + DSW Algorithm for large datasets, it outperforms the DSW algorithm for the smaller datasets in two out of the three samples, leading to the idea that the BST + DSW Algorithm may be desirable when working with datasets of size 1-3 million or smaller. However, more data may need to be collected to come to this conclusion.

Another observation can be made about the data size. Both the raw data and the graphs suggest that both methods increase exponentially relative to the array size, suggesting that they will begin to take proportionally longer to execute as the array size is increased. As for the integer range, there seems to be significant impacts in the timing when the array is small, however impacts are significantly mitigated as the array size increases. This could likely be because array sizes with a smaller range could be more clumped together, leading to more inefficiencies within the algorithms. However, the data seemed somewhat inconsistent, so more samples may need to be taken to solidify these claims.

As previously mentioned, the data increases exponentially, leading to the belief that these algorithms are likely to fall between O(n) and O(n^2) time complexity, likely around nlog(n). To uncover this, each dataset was compared to numerous different equations within this range. Unsurprisingly, every BST and AVL tree from all samples most closely resembled the equation $an \times \log(n) + bn + c$, with the similar equation $an \times \log(n) + b$ showcasing very similar, if not the exact same, representation of the data. The lowest r^2 value for the best fit equation was .978, with the highest r^2 value being 1. Therefore, it can be concluded that these lines are likely the optimal fit for this data.

Reference Data

JJ McCauley – BST Insertion vs. AVL Insertion Timing

Seconds

Element Size (millions)	BST Insertion + DSW Algorithm	AVL Insertion
Element Range: 0-50		
1	0.441255	0.246517
3	8.99175	4.41296
5	14.3483	7.75063
7	11.4223	9.45838
9	19.4805	11.2682
11	28.8773	18.6191
13	36.8074	23.0997
15	46.0891	27.8351
20	61.7129	38.9801
Element Range: 0-250		
1	0.367914	0.500963
3	2.69211	2.74646
5	6.73876	5.90567
7	11.7372	9.17462
9	18.2088	12.885
11	25.0047	16.8303
13	33.3145	21.2717
15	42.4818	25.8345
20	57.9844	36.2905
Element Range: 0-1000		
1	0.342452	0.461769
3	2.68234	2.6298
5	6.91983	5.94892
7	12.3116	9.59533
9	18.7343	13.5857
11	26.089	17.7492
13	34.0026	22.2495
15	44.1409	26.9777
20	59.4248	37.8334

Additional Testing- Element Range: 0-99999		
10	49.6111	19.5828
25	84.2243	51.1791
50	143.839	106.789