results algoritms

Adrian Arimany - 211063

2025-02-10

```r
algorithm_times <- function(data, name) {
  ggplot(data, aes(x = InputSize)) +
    geom_line(aes(y = TimeNano, color = "Measured Time"), size = 1) +
    geom_point(aes(y = TimeNano, color = "Measured Time"), size = 2) +
    geom_line(aes(y = theoretical_time, color = "Theoretical Time"), linetype = "dashed", size = 1) +
    geom_point(aes(y = theoretical_time, color = "Theoretical Time"), size = 2) +
    theme_minimal() +
    labs(title = paste("Measured vs Theoretical Execution Time for", name),
         x = "Input Size (n)",
         y = "Time (nanoseconds)",
         color = "Legend") +
    scale_color_manual(values = c("Measured Time" = "blue", "Theoretical Time" = "red"))
}
```

Merge Sort Method

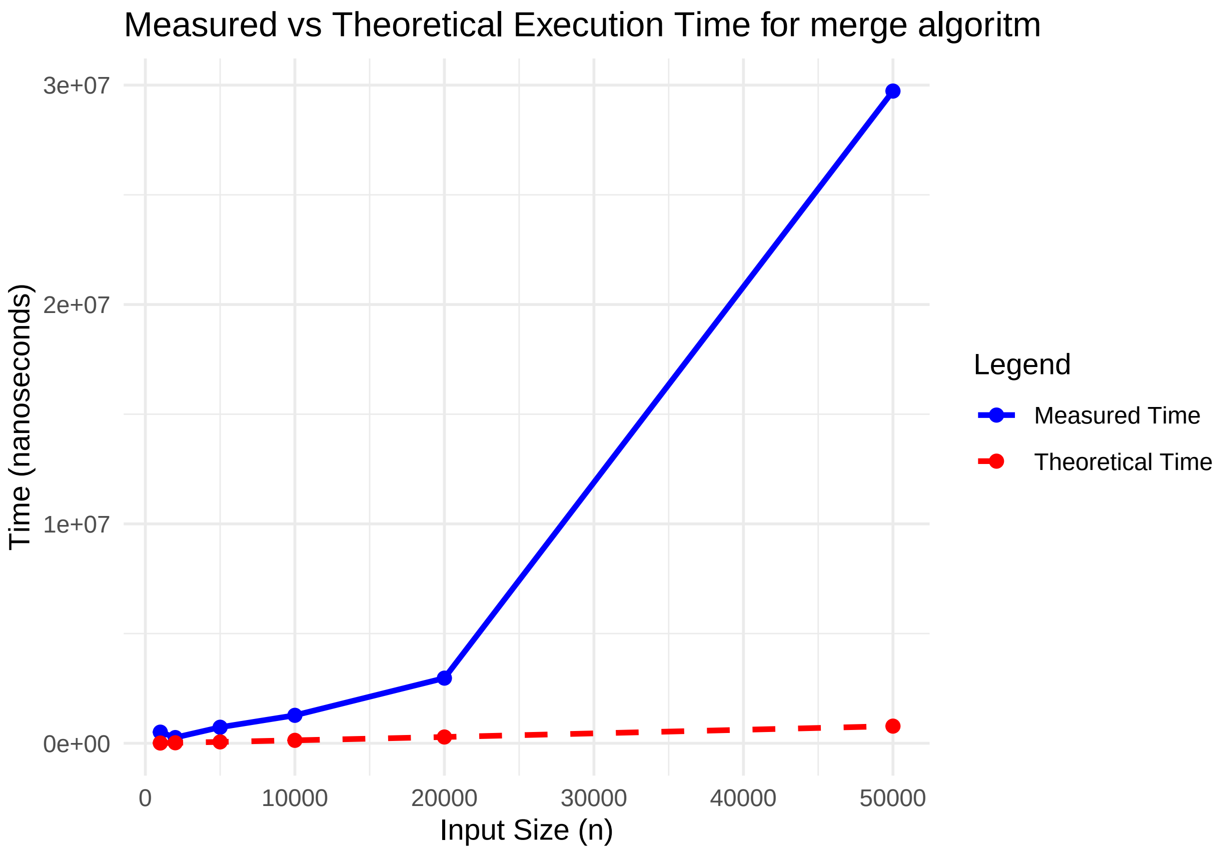
Data:

```r
mergedata <- csvPath("merge_sort_times.csv", removeSpace = FALSE)
# Compute O(n log n)
mergedata <- mergedata %>%
  mutate(
    theoretical_time = input_sizes * log2(input_sizes)
  )
head(mergedata)
```

```
## # A tibble: 6 x 3
##   InputSize TimeNano theoretical_time
##       <dbl>    <dbl>            <dbl>
## 1      1000   506638            9966.
## 2      2000   260374           21932.
## 3      5000   730344           61439.
## 4     10000  1273090          132877.
## 5     20000  2968728          285754.
## 6     50000 29729677          780482.
```

Plots:

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



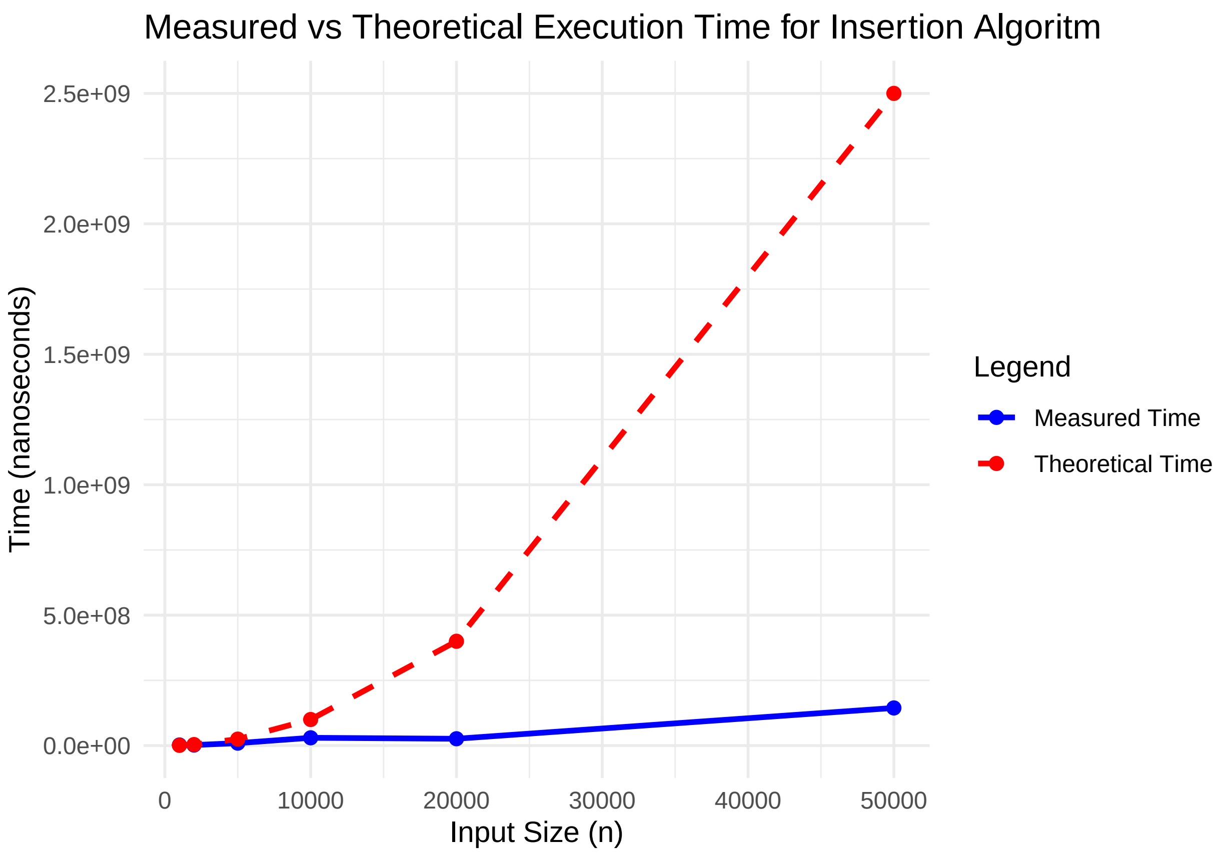
Insertion Sort Method

Data:

```
Insertiondata <- csvPath("insertion_sort_times.csv", removeSpace = FALSE)

Insertiondata <- Insertiondata %>%
  mutate(
    theoretical_time = (input_sizes)^2  # O(n^2)
  )
```

Plots:



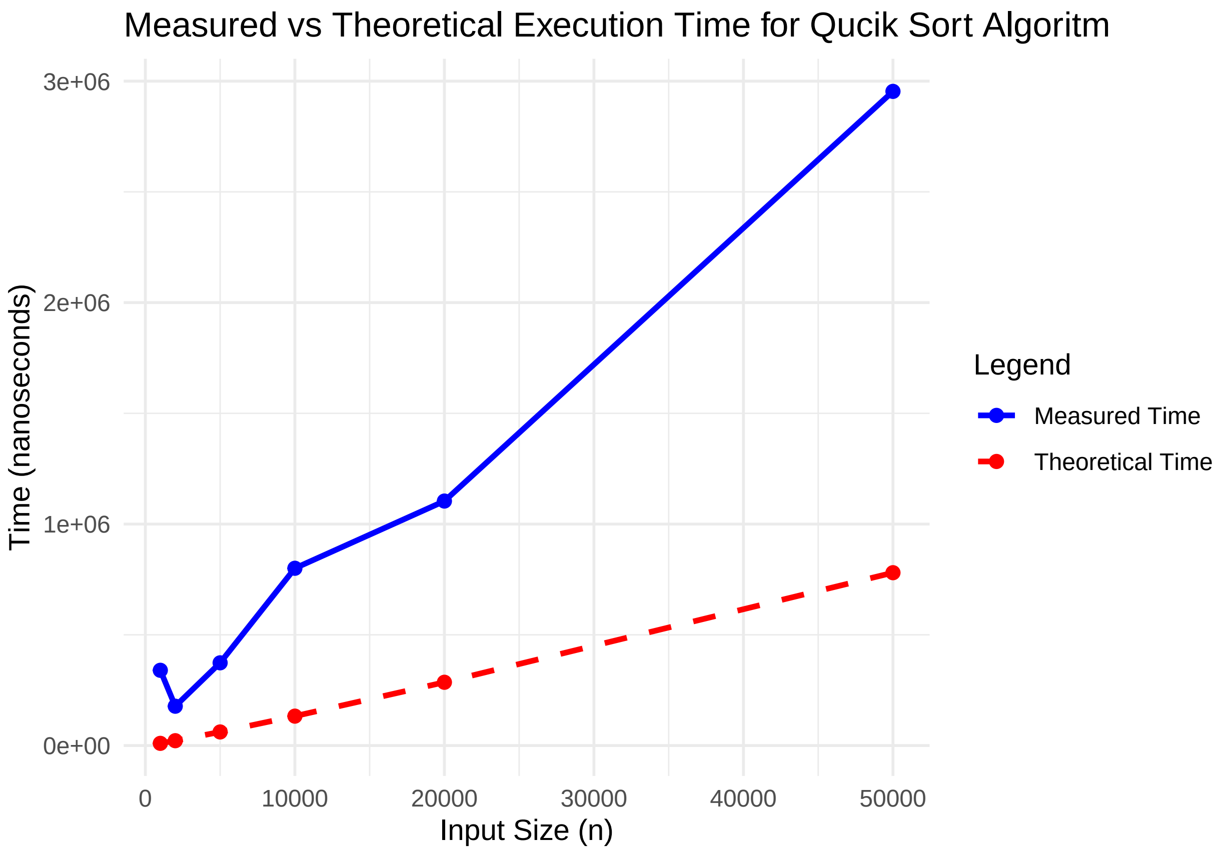
Quick sort

Data:

```
quicksortData <- csvPath("quick_sort_times.csv", removeSpace = FALSE)
quicksortData <- quicksortData %>%
  mutate(
    theoretical_time = input_sizes * log2(input_sizes)  # O(n log n)
  )
head(quicksortData)
```

```
## # A tibble: 6 x 3
##   InputSize TimeNano theoretical_time
##       <dbl>    <dbl>            <dbl>
## 1      1000   339842            9966.
## 2      2000   177604           21932.
## 3      5000   373575           61439.
## 4     10000   800508          132877.
## 5     20000  1103963          285754.
## 6     50000  2953748          780482.
```

Plots:



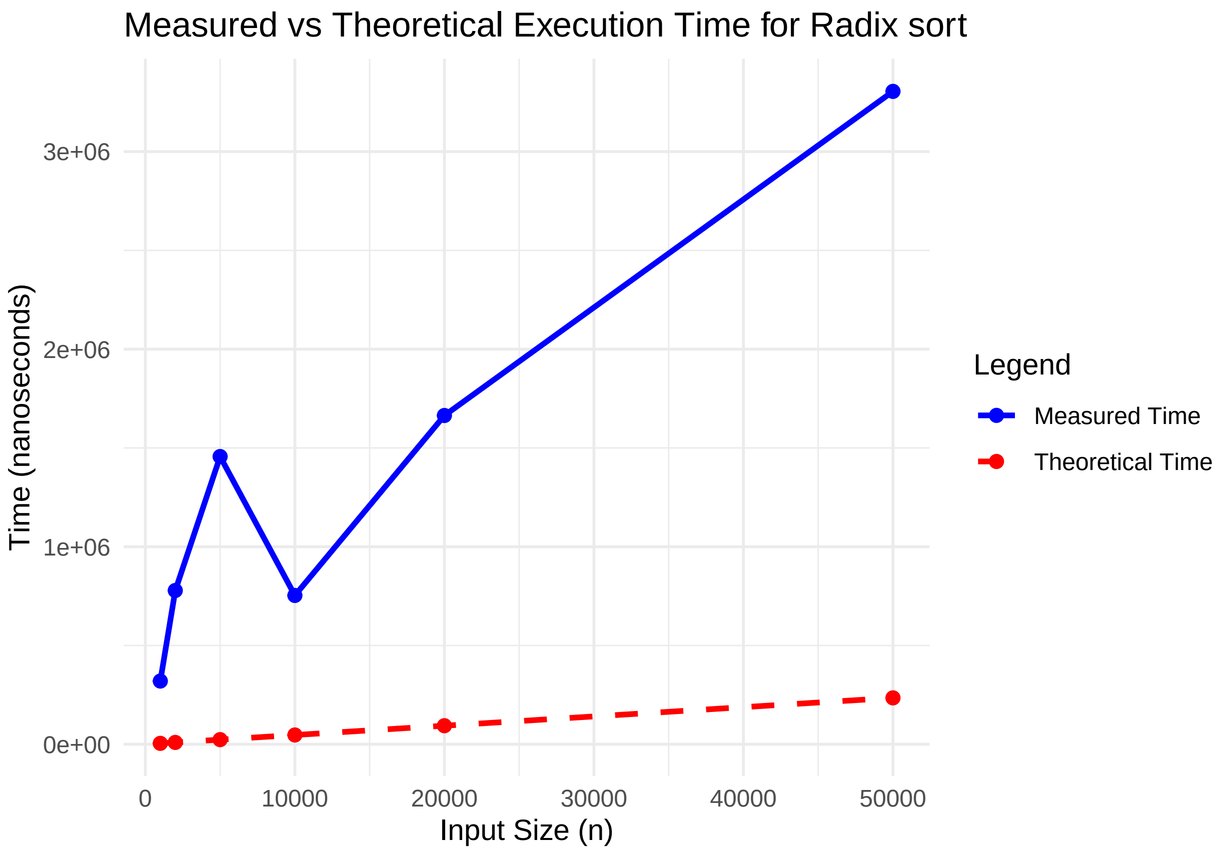
Radix Sort

Data:

```r
radfixdata <- csvPath("radix_sort_times.csv")
radfixdata <- radfixdata %>%
  mutate(
    theoretical_time = input_sizes * log10(max(input_sizes))  # O(nk), assuming k = log(n)
  )
head(radfixdata)
```

```
## # A tibble: 6 x 3
##   InputSize TimeNano theoretical_time
##       <dbl>    <dbl>            <dbl>
## 1      1000   320078            4699.
## 2      2000   778368            9398.
## 3      5000  1456447           23495.
## 4     10000   753296           46990.
## 5     20000  1664222           93979.
## 6     50000  3304767          234949.
```

Plots:



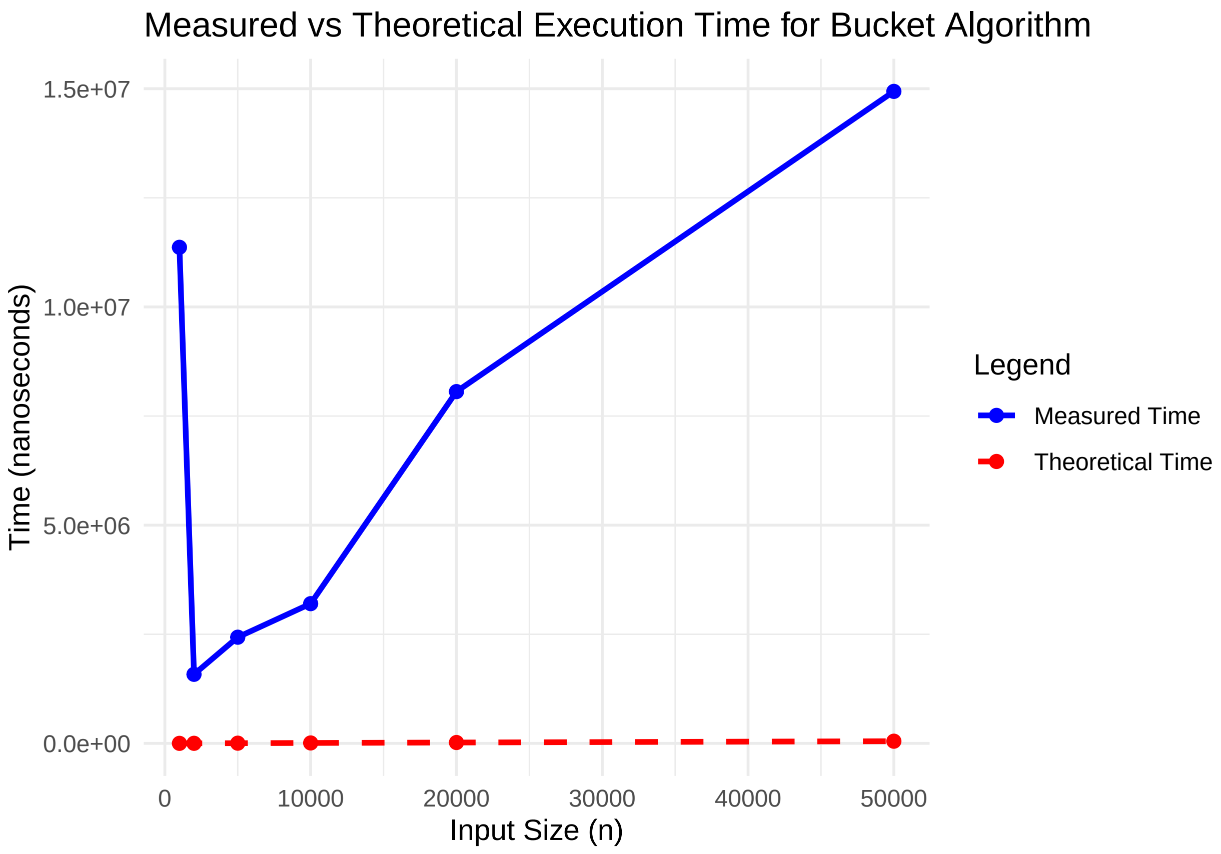
Bucket Sort

Data:

```
bucketdata <- csvPath("bucket_sort_times.csv")
bucketdata <- bucketdata %>%
  mutate(
    theoretical_time = input_sizes + sqrt(input_sizes)  # O(n + k), where k = sqrt(n)
  )
head(bucketdata)
```

```
## # A tibble: 6 x 3
##   InputSize TimeNano theoretical_time
##       <dbl>    <dbl>            <dbl>
## 1      1000 11366354            1032.
## 2      2000  1582510            2045.
## 3      5000  2433023            5071.
## 4     10000  3201824           10100
## 5     20000  8060259           20141.
## 6     50000 14938681           50224.
```

Plots:



#

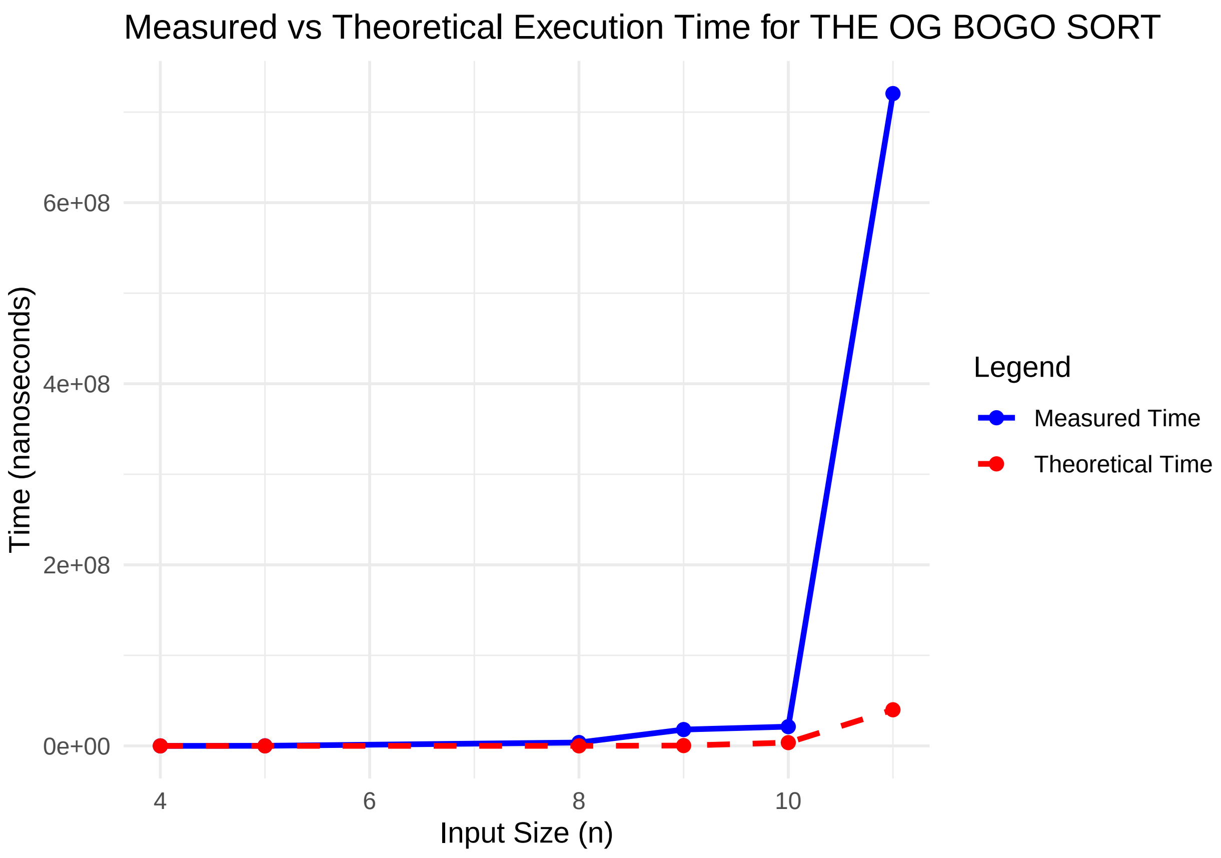
Bogo Sort

data

```r
bogodata <- csvPath("bogo_sort_times.csv")
bogodata <- bogodata %>%
  mutate(theoretical_time = factorial(InputSize))  # O(n!) complexity
```

plot

```r
algorithm_times(bogodata, name ="THE OG BOGO SORT")
```



References:

Programiz. (2025). Sorting Algorithm. Programiz: Learn to Code for Free. https://www.programiz.com/dsa/sorting-algorithm

Neto, A. (2023, May 5). Bogosort: The Stupid Sorting Algorithm. DEV Community. http://dev.to/adolfont/bogosort-the-stupid-sorting-algorithm-168f