Log-log plots for CS

The order of growth of running time functions is a critical element of computer science. One basic question to ask is "Is the running time of this algorithm linear, or quadratic?" (Many other possibilities exist, cubic, logarithmic, exponential but we'll settle for these two right now).

Plotting `size` vs `time` can be misleading because of noisy data and hard-to-establish trendlines. It can be surprisingly hard sometimes to determine if the running time is $O(n^k)$ and what k is. To help, we will make use of a *log-log* plot.

These plots are valuable because, if you have reason to believe the running time is given by $y = Cn^k$ for some constants C and k, then by taking logs, $\ln y = \ln C + k \ln n$, which shows that $\ln y$ is a linear function of $\ln n$, and the slope k is the exponent in the running time Cn^k. Thus in a log-log plot, degrees become slopes. A best-fit line's slope can tell you the original degree k.

In our case we believe we may have some $k = 1$ and some $k = 2$ running times. Here's how to check (in Google Sheets, but any spreadsheet will be similar.)

Plotting the data

Start with your original data (here's a piece of the sample from my machine)

n	Stack push	Stack pop	Queue push	Queue pop
1000	1	1	1	5
2000	1	1	1	5
4000	1	1	1	2
8000	1	1	1	8
16000	3	2	4	27
32000	9	3	5	131
64000	6	4	6	590
128000	11	7	12	2355

Make 5 new columns where each value is the `log` of the corresponding values above (example below). It's probably easiest to just add these columns to the right of the existing ones. We're going to plot these lines. *BUT FIRST* delete any '0' or 'Error' values from your timing results (those come from times of 0 or 1 ms and so we don't care about them). Instructions if you're new to spreadsheets

log n	log Stack push	log Stack pop	log Queue push	log Queue pop
3.000	0.000	0.000	0.000	0.699
3.301	0.000	0.000	0.000	0.699
3.602	0.000	0.000	0.000	0.301
3.903	0.000	0.000	0.000	0.903
4.204	0.477	0.301	0.602	1.431
4.505	0.954	0.477	0.699	2.117
4.806	0.778	0.602	0.778	2.771
5.107	1.041	0.845	1.079	3.372

Make a chart, using the scatter type (I could only get mine to show single dots, which is fine).

Finding the degree

To get the degree of each running time, we need to make a best-fit linear regression line. Here are instructions courtesy of Claude AI

Prerequisites

- Have a scatter plot or line chart already created with your data
- Your data should show some linear relationship

Steps

1. Add the Trendline - Double-click your chart to enter edit mode - Click on a data point in your series - In the Chart Editor on the right, go to the **Customize** tab - Scroll down and click **Series** - Check the box for **Trendline**

2. Set to Linear Regression - In the Trendline section, set **Type** to **Linear** - This fits a straight line using least-squares regression

3. Show the Equation - Still in the Trendline settings, check **Show equation** - Optionally, also check **Show R^2** to see how well the line fits your data

4. Customize Appearance (optional) - Adjust line color, thickness, and opacity as desired - You can also change where the equation appears on the chart

What You Get

- A best-fit line through your data points
- The equation displayed as: $\mathbf{y = mx + b}$ (where m = slope, b = y-intercept)
- R^2 value showing fit quality (closer to 1.0 = better fit)

The equation updates automatically if your underlying data changes, making it perfect for dynamic analysis and reporting.

Analysis

For the four operations, which are linear and which are quadratic? Do the results surprise you? Please add the chart and your findings to the analysis document you already wrote.

Next Steps

We're going to make everything linear!