Characterizing Execution Times on Realistic Programs

Database & Big Data Systems Laboratory,
School of Computer Science and Engineering,
Kyungpook National University,
Young-Kyoon Suh

April 3, 2018

1 Description

This document characterizes execution times measured on several real-world programs with different input sizes. To achieve this characterization, we discuss various histograms of execution times, measured in program time (PT), of the programs throughout this document. In this work we wish to achieve several goals as follows. The first goal is to unravel any structure behind the histograms and present insights into how such structure is formed. Another goal is to build a statistical distribution (or model) fitting in the histograms. From that distribution, we may reach predicting a concrete execution time considering system noise via the model on an arbitrary algorithm with a given input on a real execution environment. As a note, the execution times were measured along with the EMPv5 [1] protocol.

The following section shows histograms for runs on different real-world programs with varying input sizes.

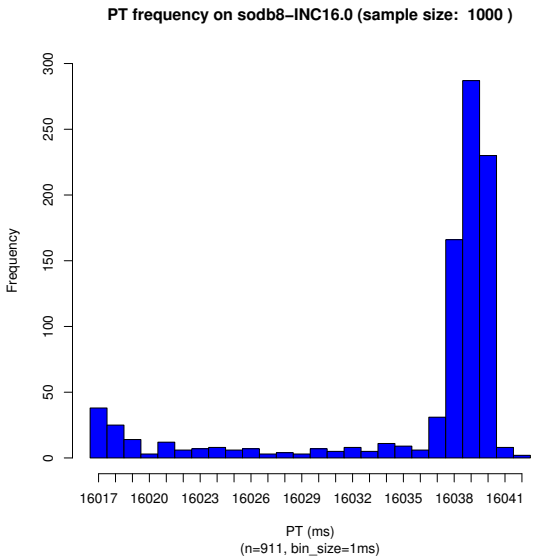
2 Histograms of the Execution Times on Real-World Programs

In this section we run three kinds of programs: *nested for-loop*, *insertion sort* and *matrix multiplication*. For the runs of the latter two programs, we varied their input sizes by $2\times$ while trying some intermediate sizes increased by $\sqrt{2}$ and measured execution times of the programs over each input size. We then exhibit different kinds of histograms in the subsequent sections.

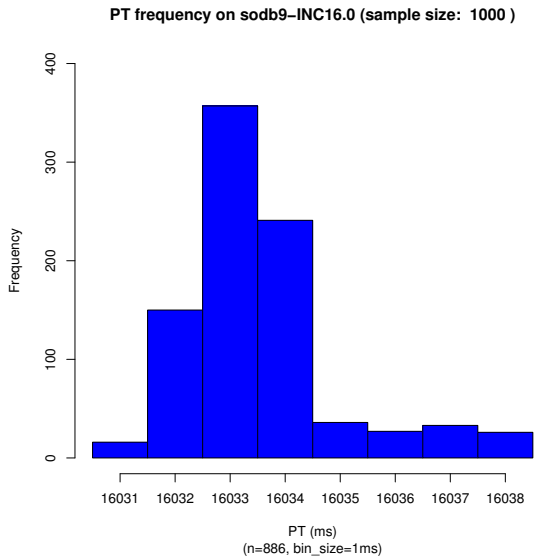
2.1 Inter-machine Repeatability Check

In this section we check if the use of different machines affects the distribution of execution (process) times on the same program. In other words, we examine *inter-machine repeatability*. For this examination, we use the nested for loop program with different task lengths: 16 seconds, 13 seconds, and 17.2 seconds, termed *INC16*, *INC13*, and *INC17.2*, respectively.

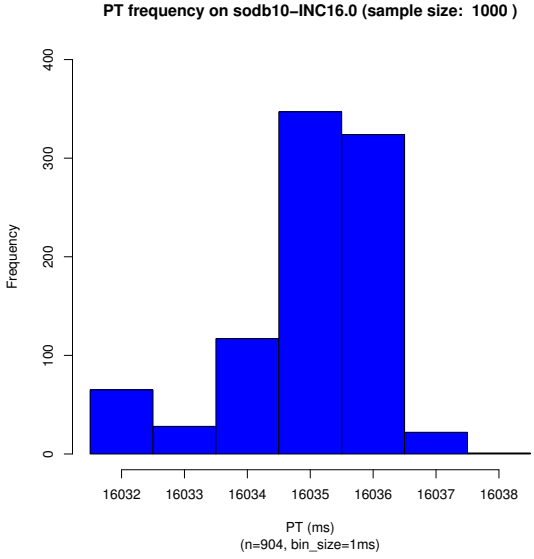
Figures 1, 2, and 3 display the process time (PT) histograms on INC16, INC13, and INC17.2 run on our machines (from `sodb8` to `sodb12`), respectively.



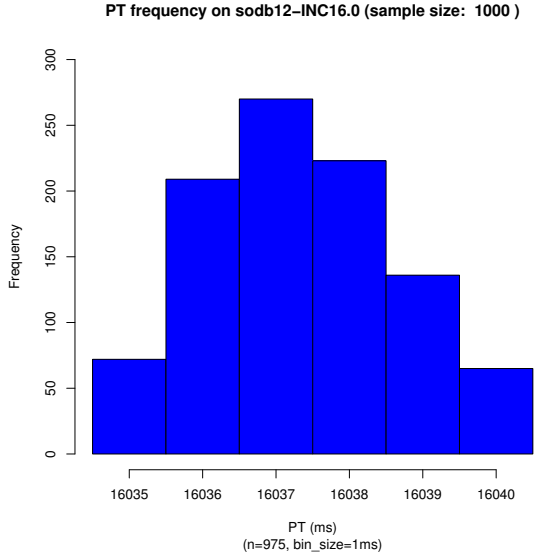
(a) PT frequency on INC16



(b) PT frequency on INC16

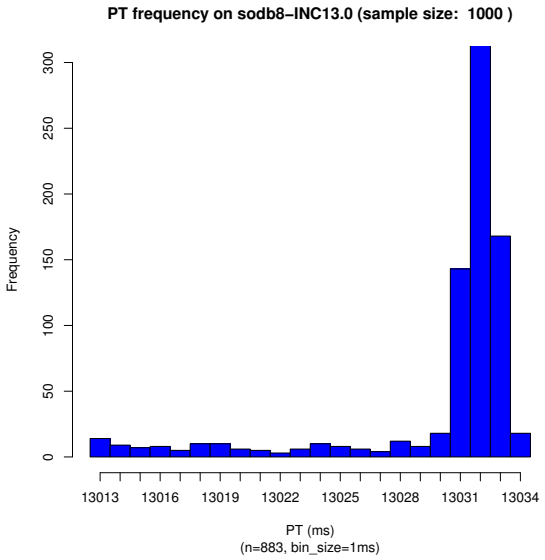


(c) PT frequency on INC16

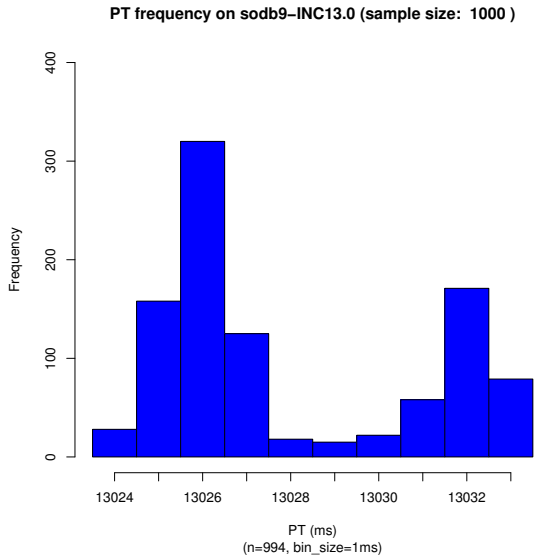


(d) PT frequency on INC16

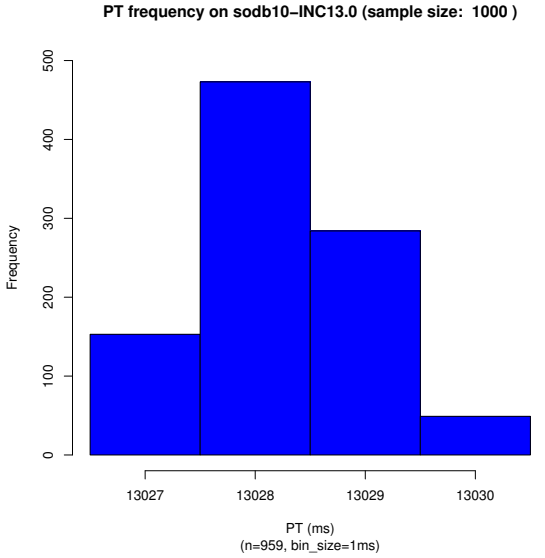
Figure 1: PT Histograms on INC16



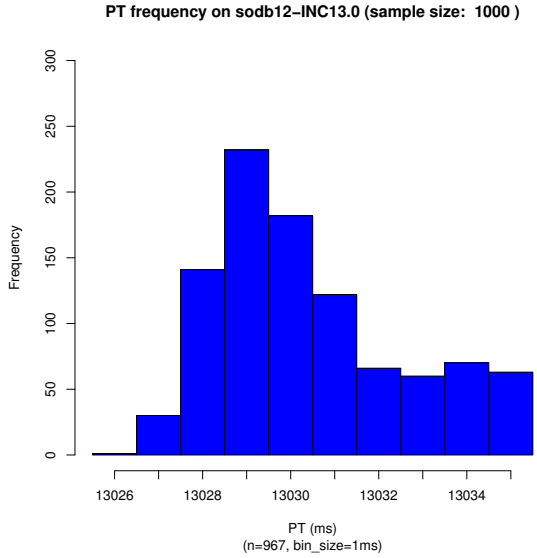
(a) PT frequency on INC13



(b) PT frequency on INC13

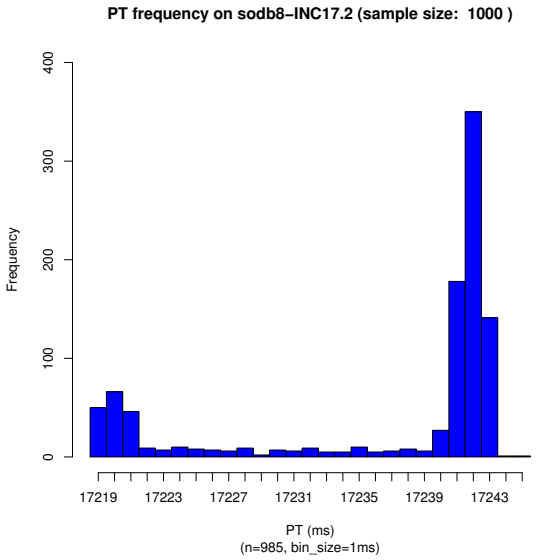


(c) PT frequency on INC13

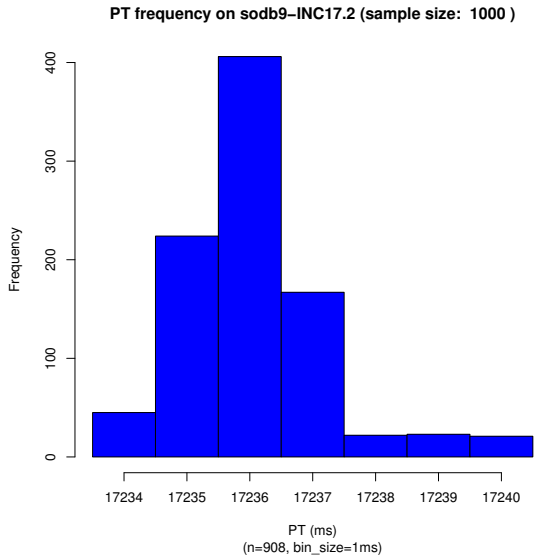


(d) PT frequency on INC13

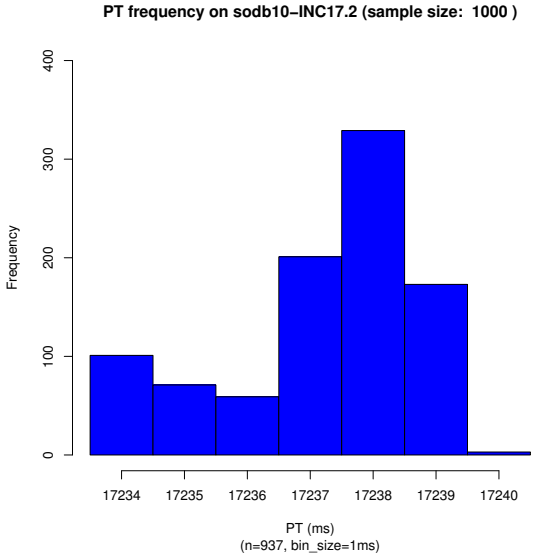
Figure 2: PT Histograms on INC13



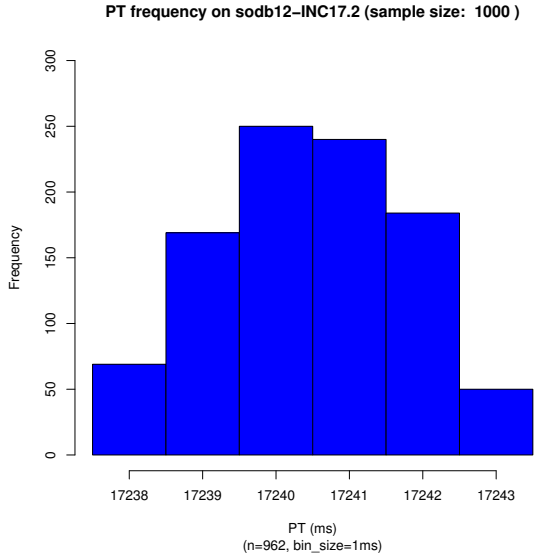
(a) PT frequency on INC17.2



(b) PT frequency on INC17.2



(c) PT frequency on INC17.2

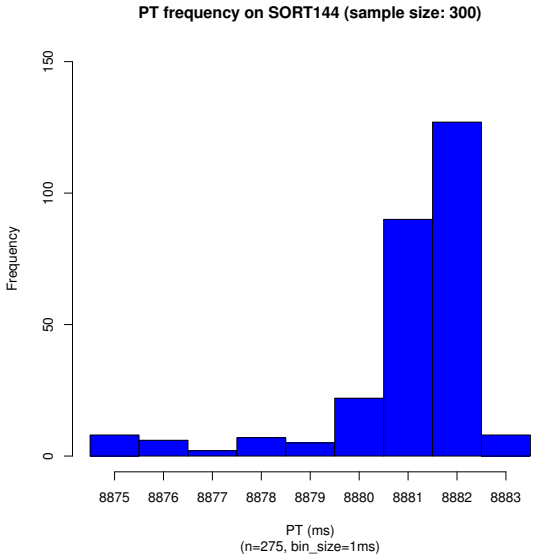


(d) PT frequency on INC17.2

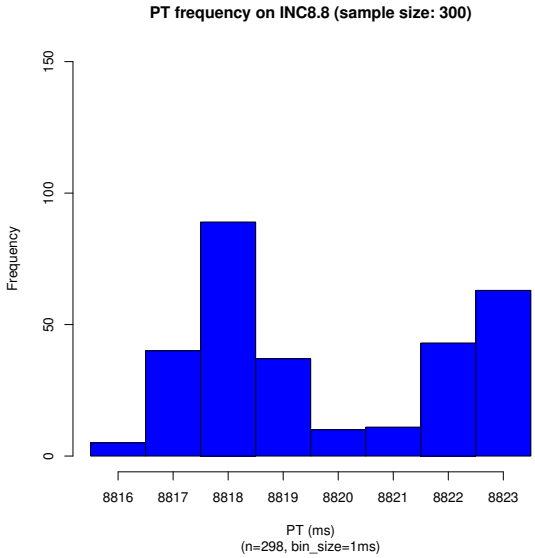
Figure 3: PT Histograms on INC17.2

2.2 One-to-One Comparison between An Insertion Sort & A Corresponding INC Programs

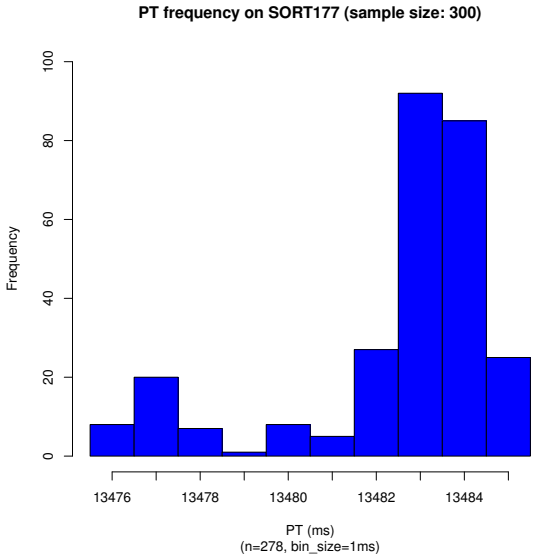
This section compares program time histograms of insertion sort and INC (a nested-for-loop) programs. The insertion sort program sorts the elements of a given array in non-decreasing order. The program repeatedly runs 300 times for a given input size. The input size for the program varies from 144K to 344K integer elements, which are randomly generated. Note that each sort program over a specific input size is termed SORTx: for instance, SORT100 indicates the insertion sort program over 100K elements. An INC program's task length is correspondingly determined by the program time of an SORT program. In Figures 4, 5, and 6 we perform a match on an SORT program and its corresponding INC program.



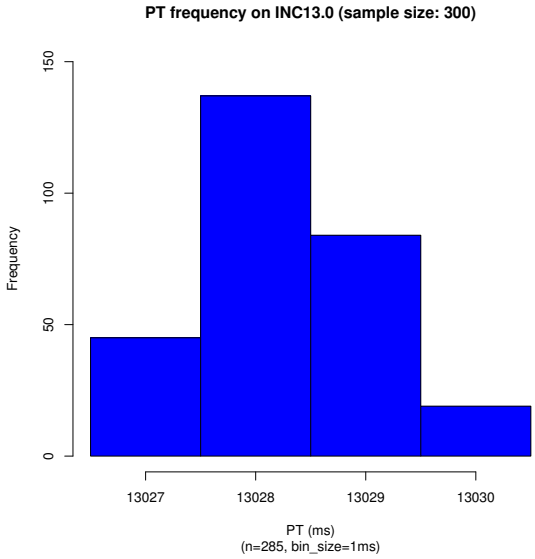
(a) PT frequency on SORT144



(b) PT frequency on INC8.8

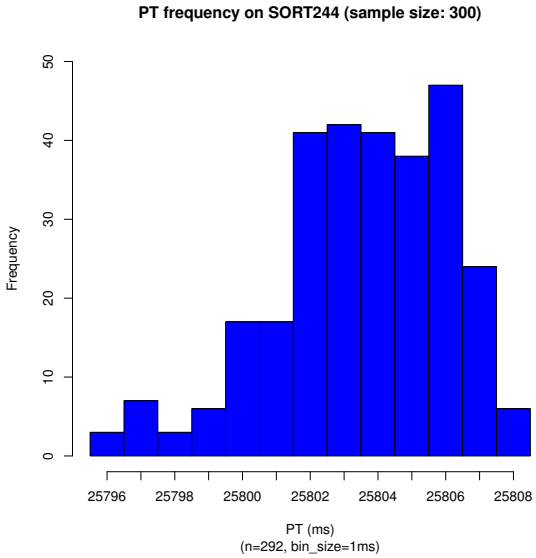


(c) PT frequency on SORT177

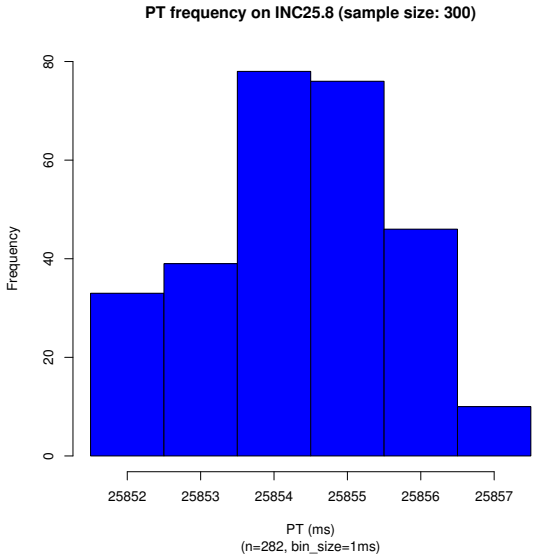


(d) PT frequency on INC13

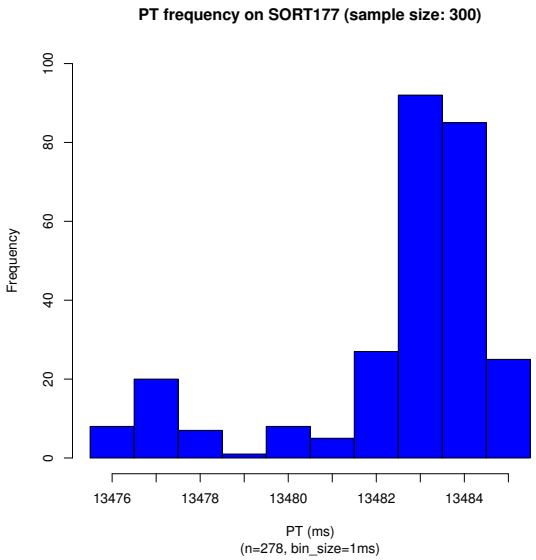
Figure 4: PT Histograms I



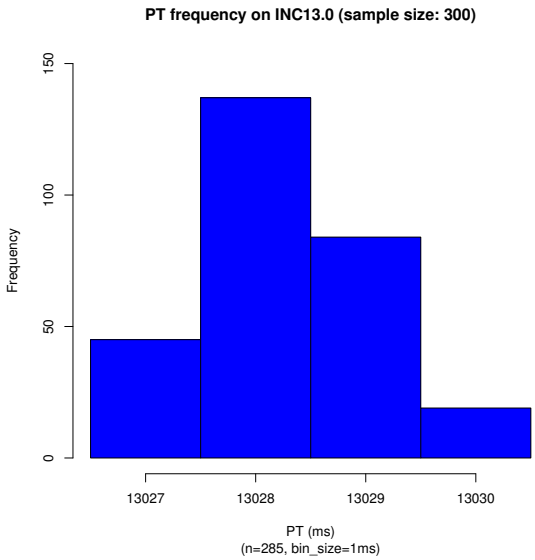
(a) PT frequency on SORT144



(b) PT frequency on INC25.8

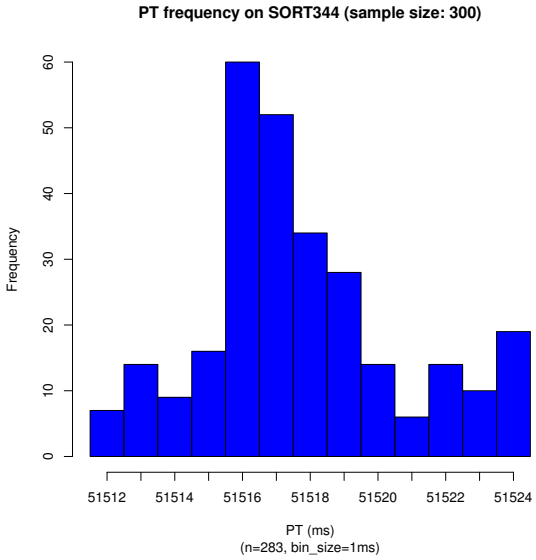


(c) PT frequency on SORT288

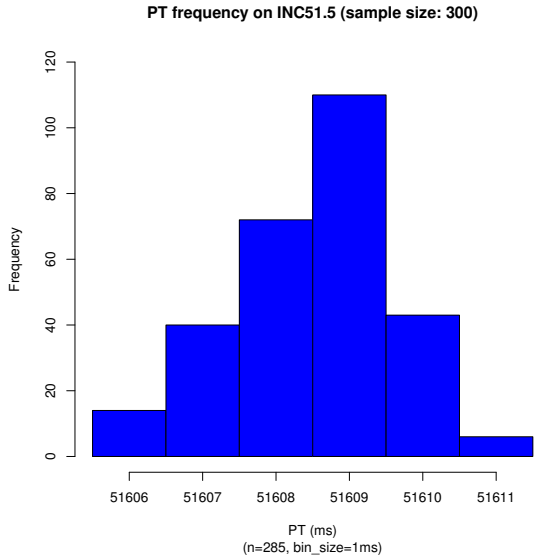


(d) PT frequency on INC35

Figure 5: PT Histogram Comparison II



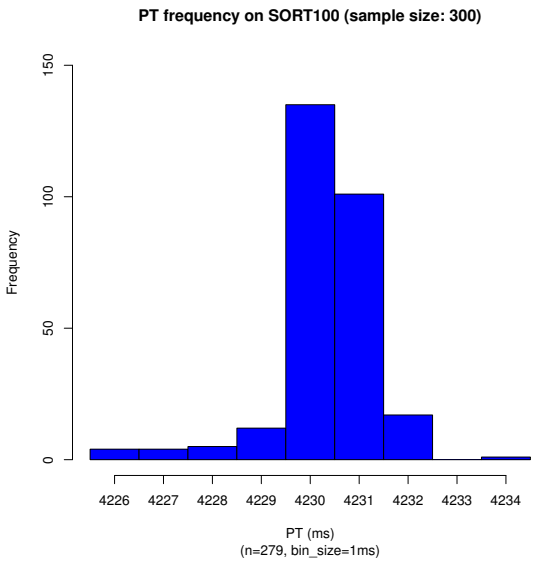
(a) PT frequency on SORT344



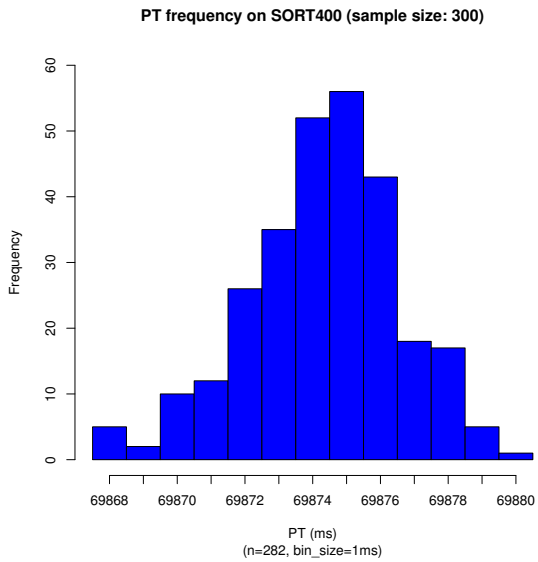
(b) PT frequency on INC51.5

Figure 6: PT Histogram Comparison III

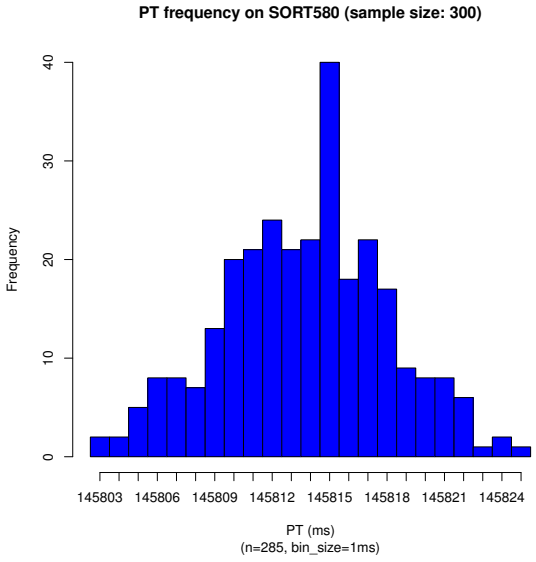
3 Additional Insertion Sort



(a) PT frequency on SORT100

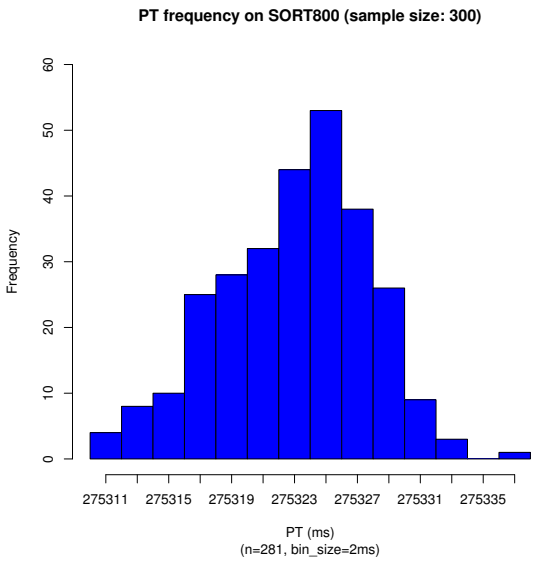


(b) PT frequency on SORT400

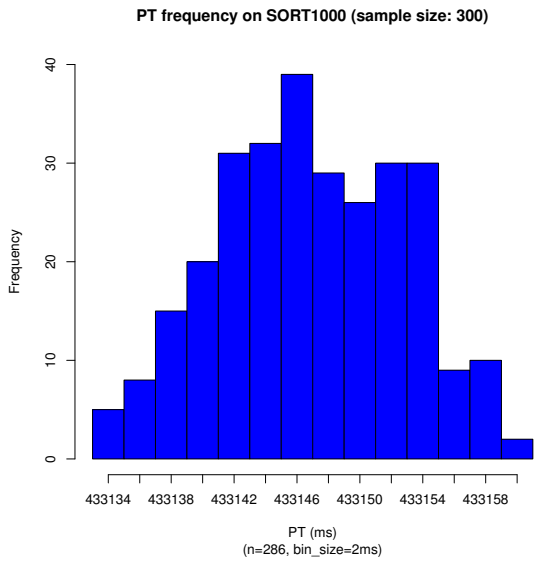


(c) PT frequency on SORT580

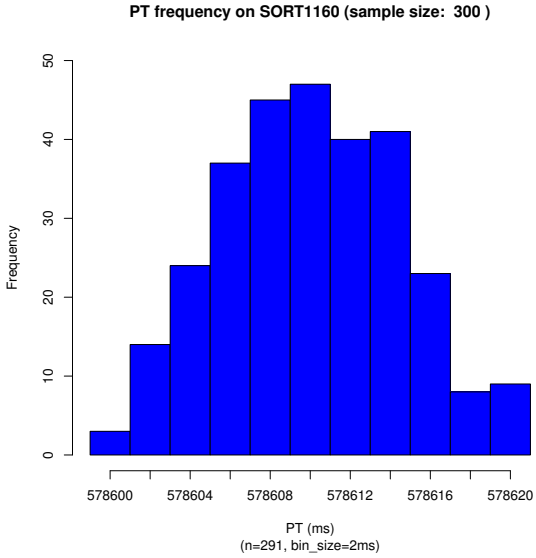
Figure 7: PT Histograms of SORT100 ... SORT580



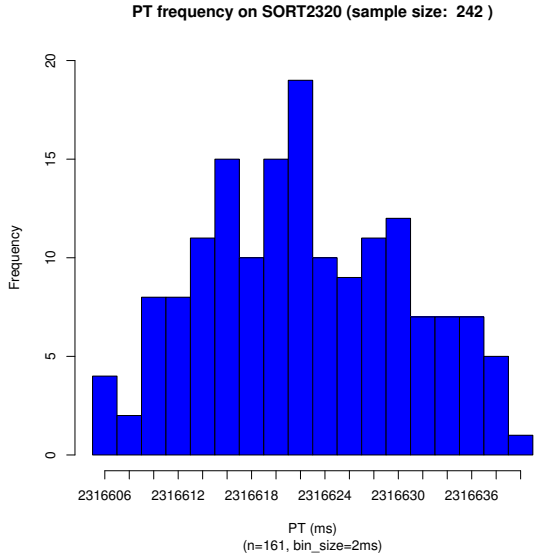
(a) PT frequency on SORT800



(b) PT frequency on SORT1000



(c) PT frequency on SORT1160



(d) PT frequency on SORT2320 (not complete)

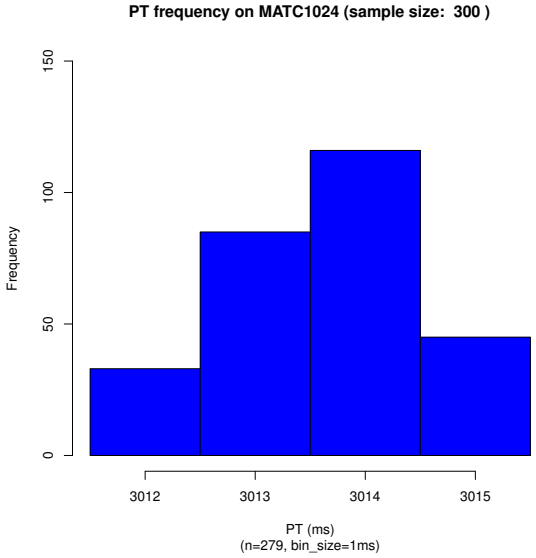
Figure 8: PT Histograms of SORT800 ... SORT2320

3.1 Matrix Multiplication

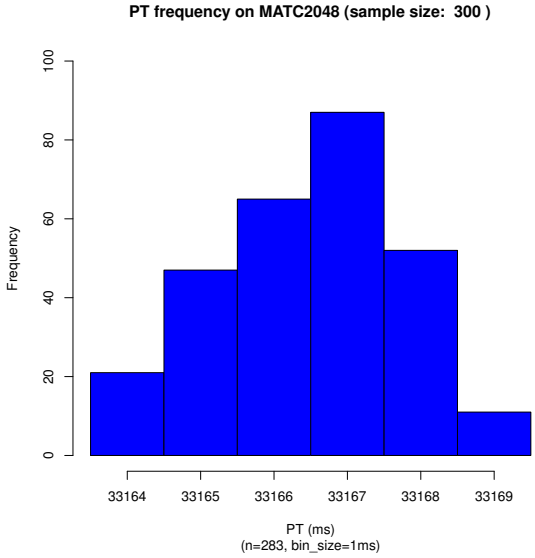
This section shows a series of histograms of the execution times of an matrix multiplication program. We used the same sample size for each input size of this program: i.e., 40 iterations. For simplicity, we used two square matrices for performing their multiplication in the program. We also varied the input sizes of each of the two matrices: from 1K×1K to 8K×8K integer elements that are also randomly generated. Note that each matrix multiplication program for a specific size is called MAT*xyyyy*, where *x* indicates which major, specifically *column* vs. *row*, is used, and *yyyy*, how large a given matrix is. For instance, MATC1000 represents a matrix multiplication program in column major over two square matrices having 1,000 integer (random) elements in a row (and a column).

3.1.1 Column Major

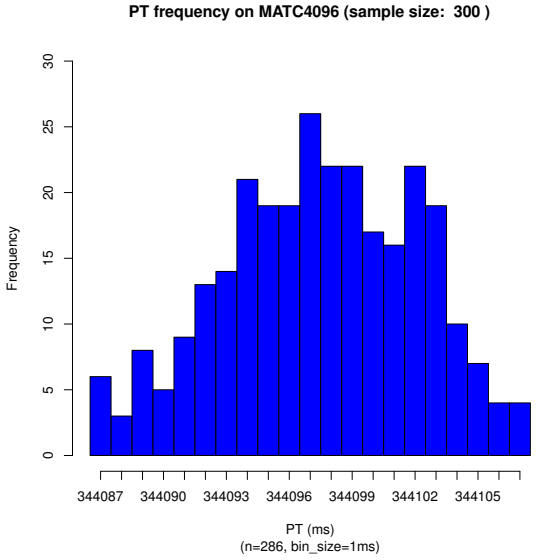
Figure 9 shows a series of histograms of the execution times measured on the same matrix multiplication program in column major as the input sizes grows.



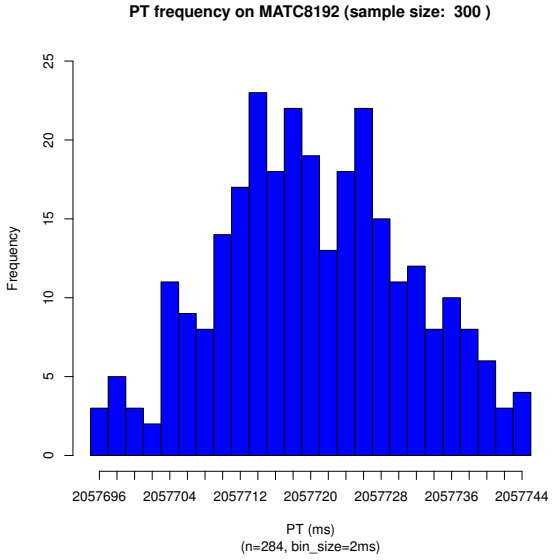
(a) PT frequency on MATC1K



(b) PT frequency on MATC2K



(c) PT frequency on MATC4K

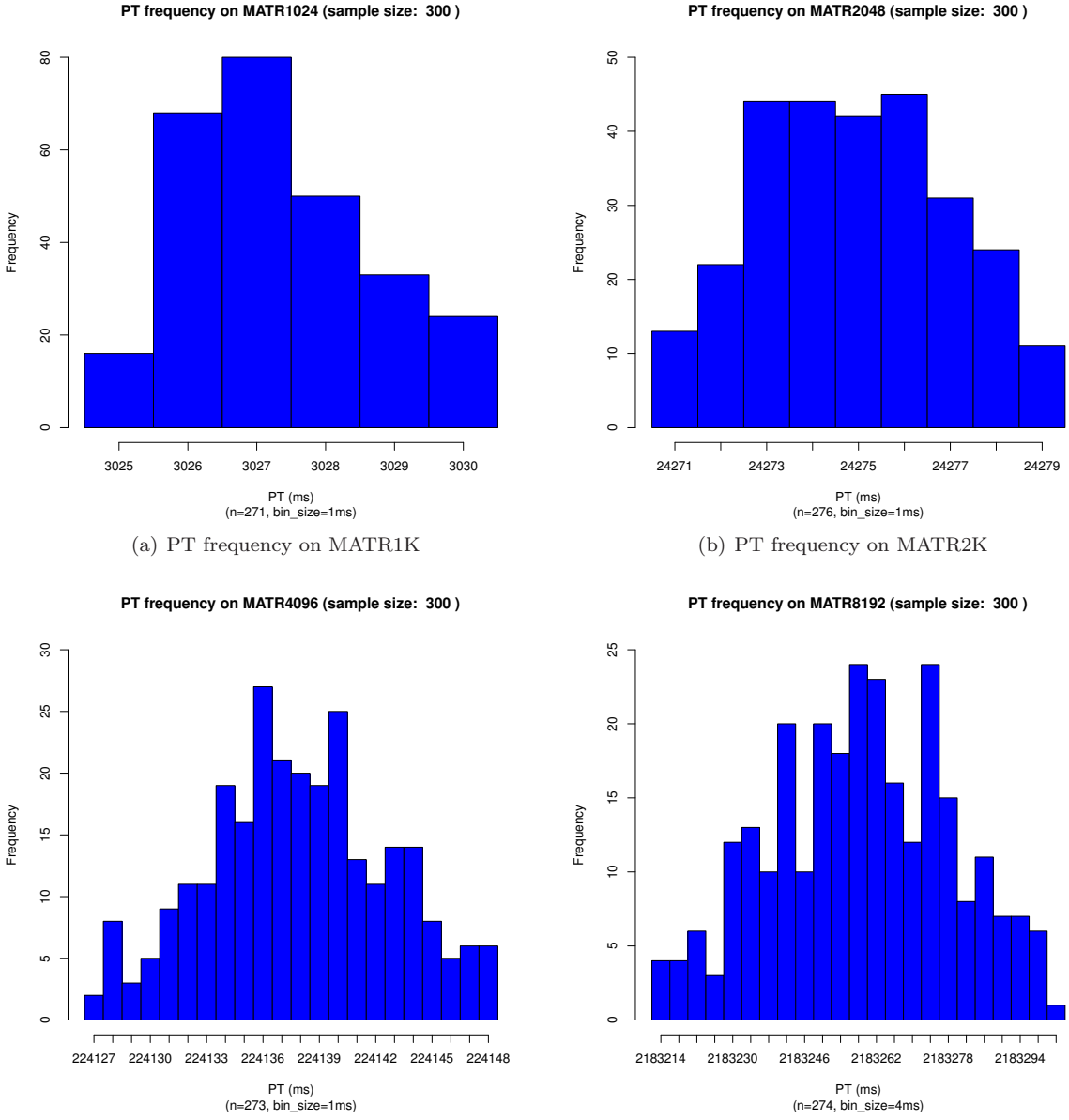


(d) PT frequency on MATC8K

Figure 9: PT Histograms of MATC1024 ... MATC8192

3.1.2 Row Major

Figure 9 shows a series of histograms of the execution times measured on the same matrix multiplication program in row major as the input sizes grows.



(a) PT frequency on MATR1K

(b) PT frequency on MATR2K

(c) PT frequency on MATR4K

(d) PT frequency on MATR8K

Figure 10: PT Histograms of MATR1K ... MATR8K

References

[1] Young-Kyoon Suh, Richard T. Snodgrass, John Kececioglu, Peter J. Downey, Rob S. Maier, and Cheng Yi, "EMP: Execution Time Measurement Protocol for Compute-Bound Programs", in *Software: Practice and Experience*, 47(4):559–597, 2017.

[2] Sabah Currim,Richard T. Snodgrass, Young-Kyoon Suh, and Rui Zhang, "DBMS Metrology: Measuring Query Time", in *ACM Transactions on Database Systems*, 42(1):3:1–42(+8), 2017.