Characterizing Execution Times on Realistic Programs

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

May 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.

2   Histograms of the Execution Times on Real-World Programs

This section shows histograms for runs on different real-world programs with varying input sizes. In addition to the insertion sort program, we use another kind of program—*matrix multiplication*. For these 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   Insertion Sort on `sodb9`

This section shows a series of histograms of an insertion sorting program that sorts the elements of a given array in non-decreasing order. The program repeatedly runs 300 times for a given input size. Note that each sort program over a specific input size is termed SORTx: for instance, SORT100 indicates the insertion sort program over 100K elements. (This experiment was performed on `sodb9`.)

PT frequency on SORT100 on sodb9 (sample size: 300)

(a) PT frequency on SORT100 on `sodb9`

PT frequency on SORT200 on sodb9 (sample size: 300)

(b) PT frequency on SORT200 on `sodb9`

PT frequency on SORT400 on sodb9 (sample size: 300)

(c) PT frequency on SORT400 on `sodb9`

PT frequency on SORT800 on sodb9 (sample size: 300)
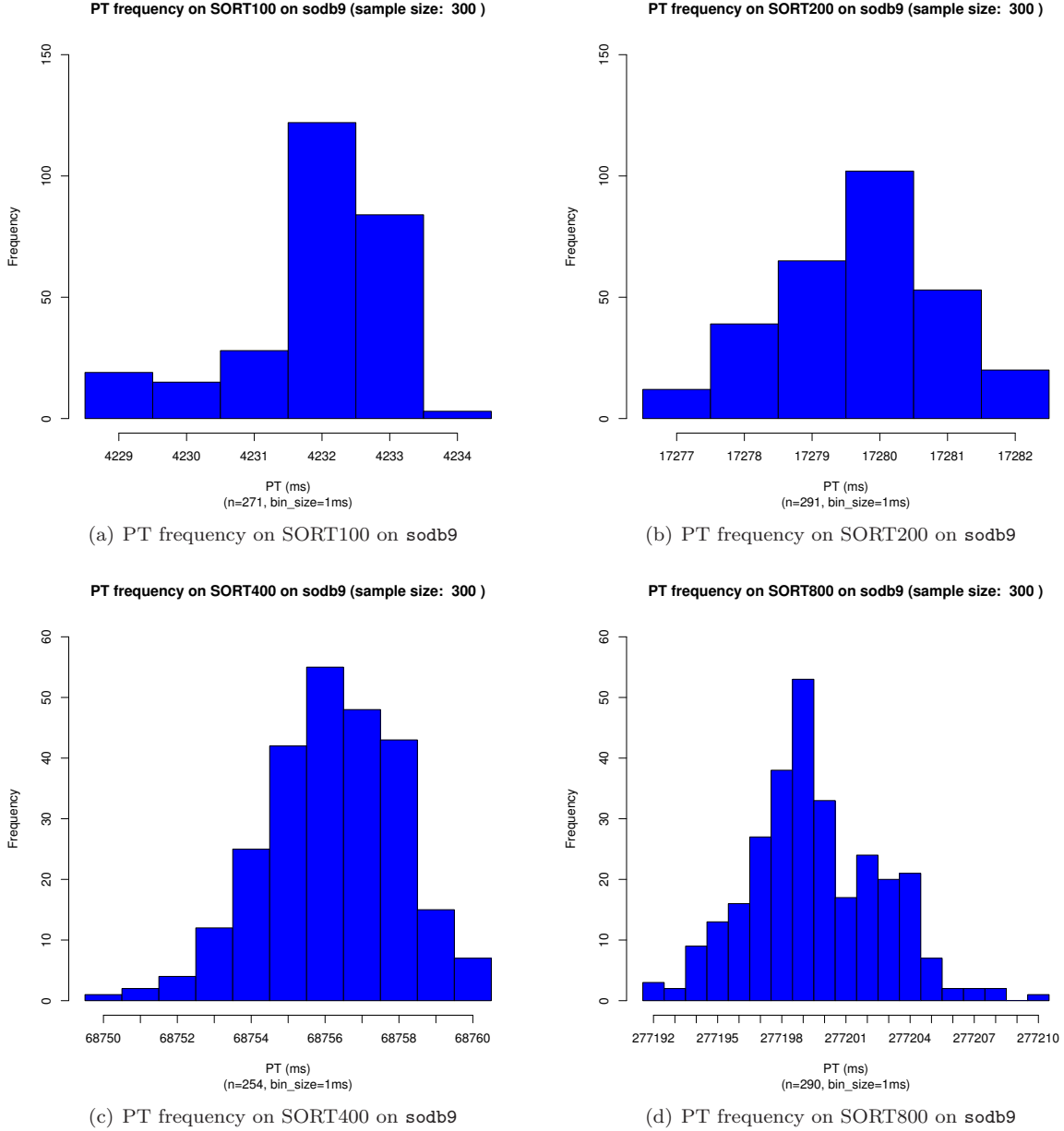
(d) PT frequency on SORT800 on `sodb9`

Figure 1: PT Histograms of SORT100 ... SORT800 on `sodb9`

2.2 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).

2.2.1 Column Major

Figure 2 shows a series of histograms of the execution times measured on the same matrix multiplication program in column major as the input sizes grows. The program was run on `sodb10`.
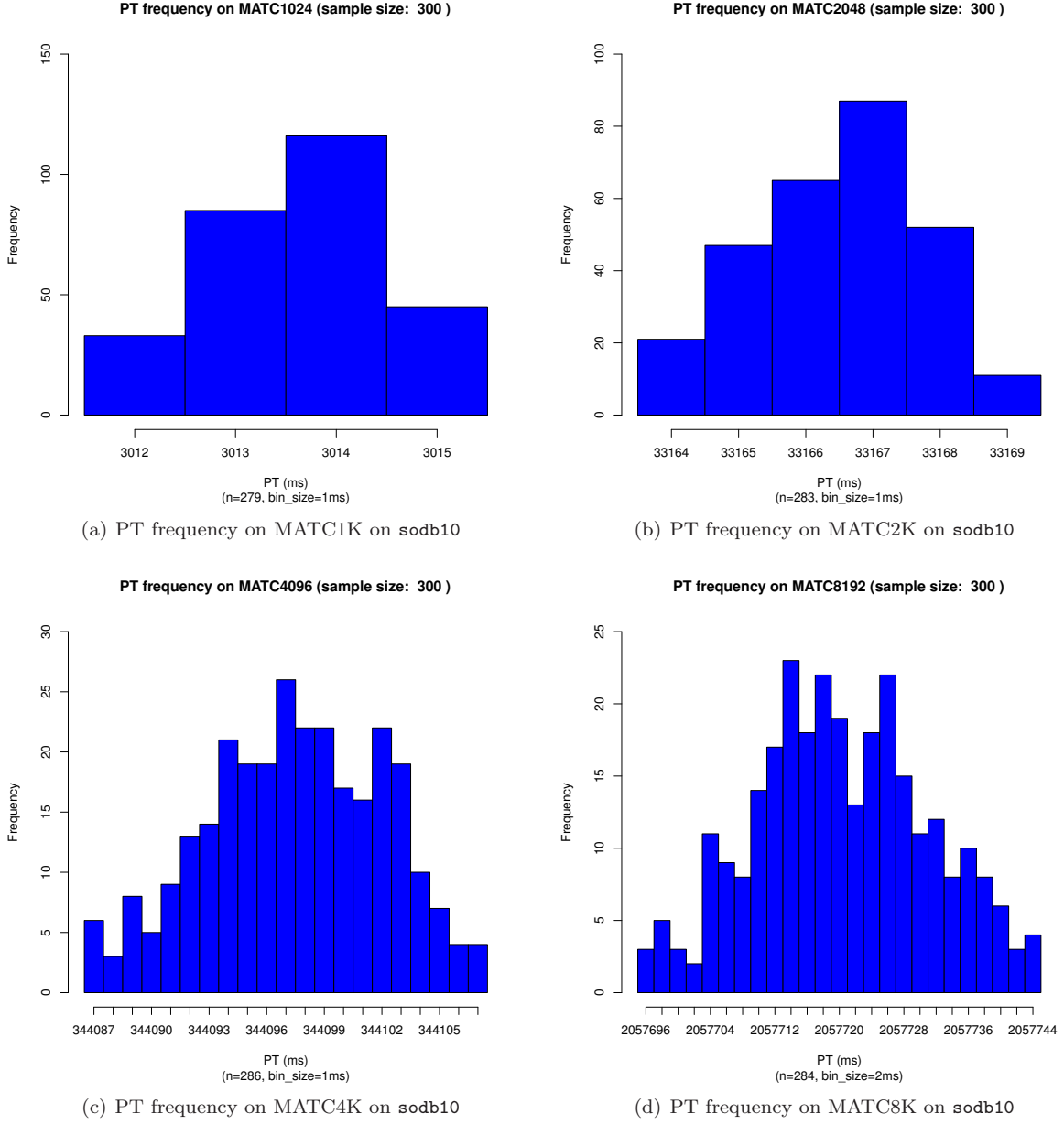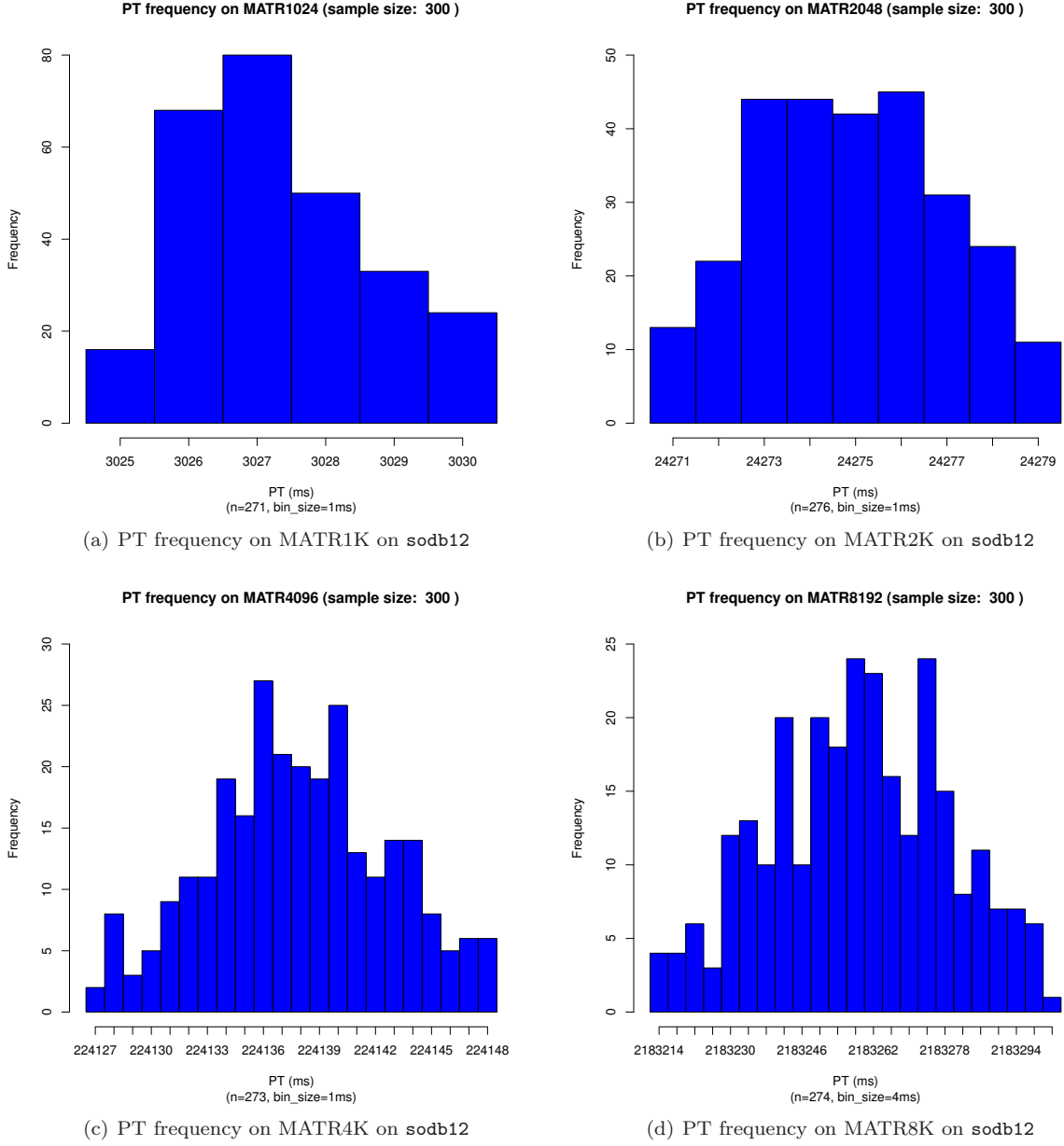

(a) PT frequency on MATC1K on `sodb10`

(b) PT frequency on MATC2K on `sodb10`

(c) PT frequency on MATC4K on `sodb10`

(d) PT frequency on MATC8K on `sodb10`

Figure 2: PT Histograms of MATC1024 ... MATC8192

2.2.2 Row Major

Figure 2 shows a series of histograms of the execution times measured on the same matrix multiplication program in row major as the input sizes grows. The program was run on `sodb12`.

(a) PT frequency on MATR1K on `sodb12`

(b) PT frequency on MATR2K on `sodb12`

(c) PT frequency on MATR4K on `sodb12`

(d) PT frequency on MATR8K on `sodb12`

Figure 3: PT Histograms of MATR1K ... MATR8K

3 Appendix

3.1 Insertion Sort on `sodb8`

This section shows a series of histograms of an insertion sorting program that 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 100,000 to 1,160,000 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. This experiment was performed on `sodb8`.
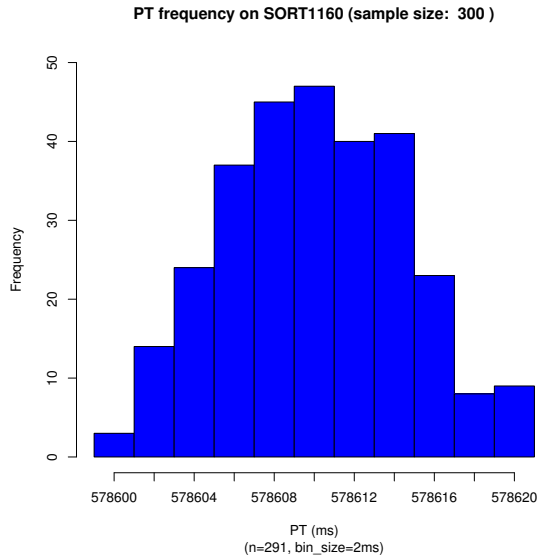

(a) PT frequency on SORT100 on `sodb8`

(b) PT frequency on SORT400 on `sodb8`

(c) PT frequency on SORT580 on `sodb8`
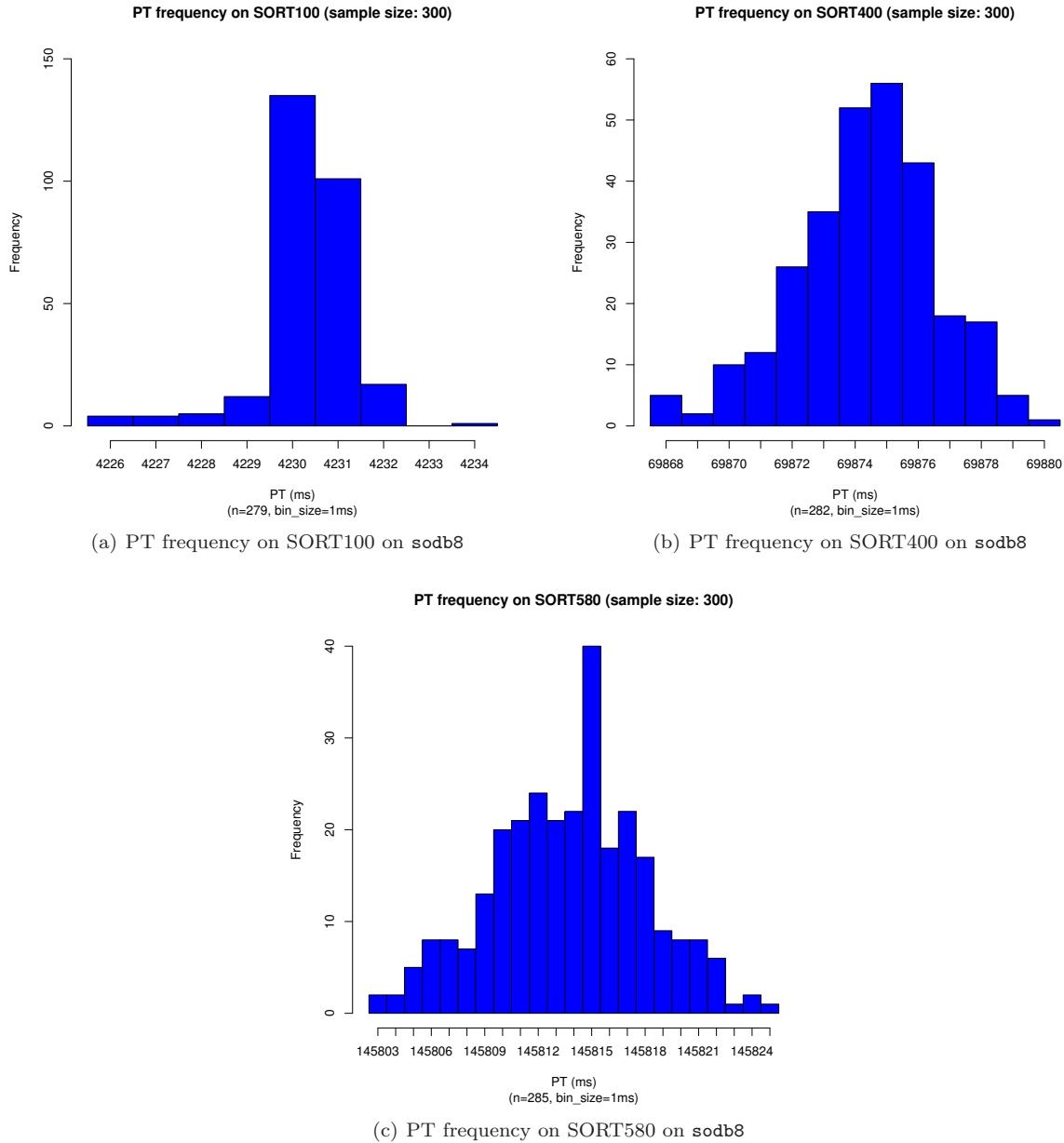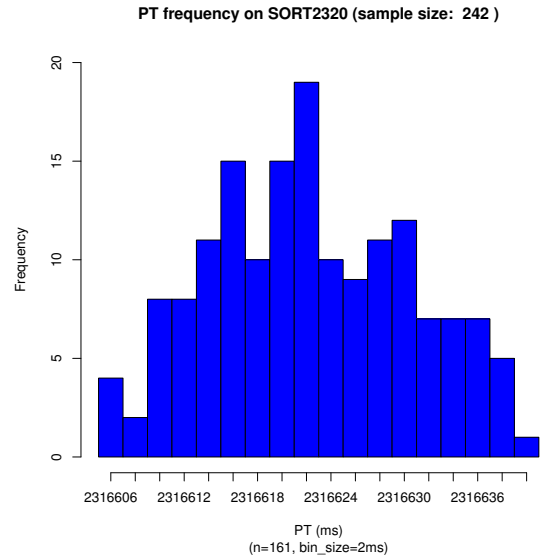
Figure 4: PT Histograms of SORT100 ... SORT580
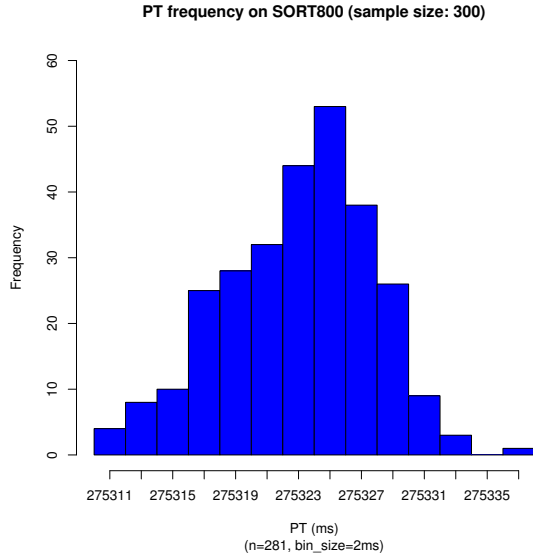
(a) PT frequency on SORT800 on `sodb8`
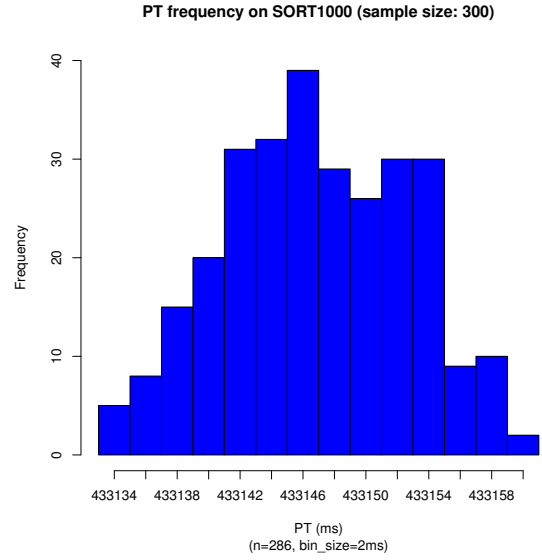
(b) PT frequency on SORT1000 on `sodb8`

(c) PT frequency on SORT1160 on `sodb8`

(d) PT frequency on SORT2320 on `sodb8`

Figure 5: PT Histograms of SORT800 ... SORT2320

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

This section compares program time histograms of an insertion sort and a nested-for-loop programs (termed INC) with different input sizes. 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 6, 7, and 8 we perform a match on an SORT program and its corresponding INC program.

(a) PT frequency on SORT144 on `sodb8`

(b) PT frequency on INC8.8 on `sodb10`
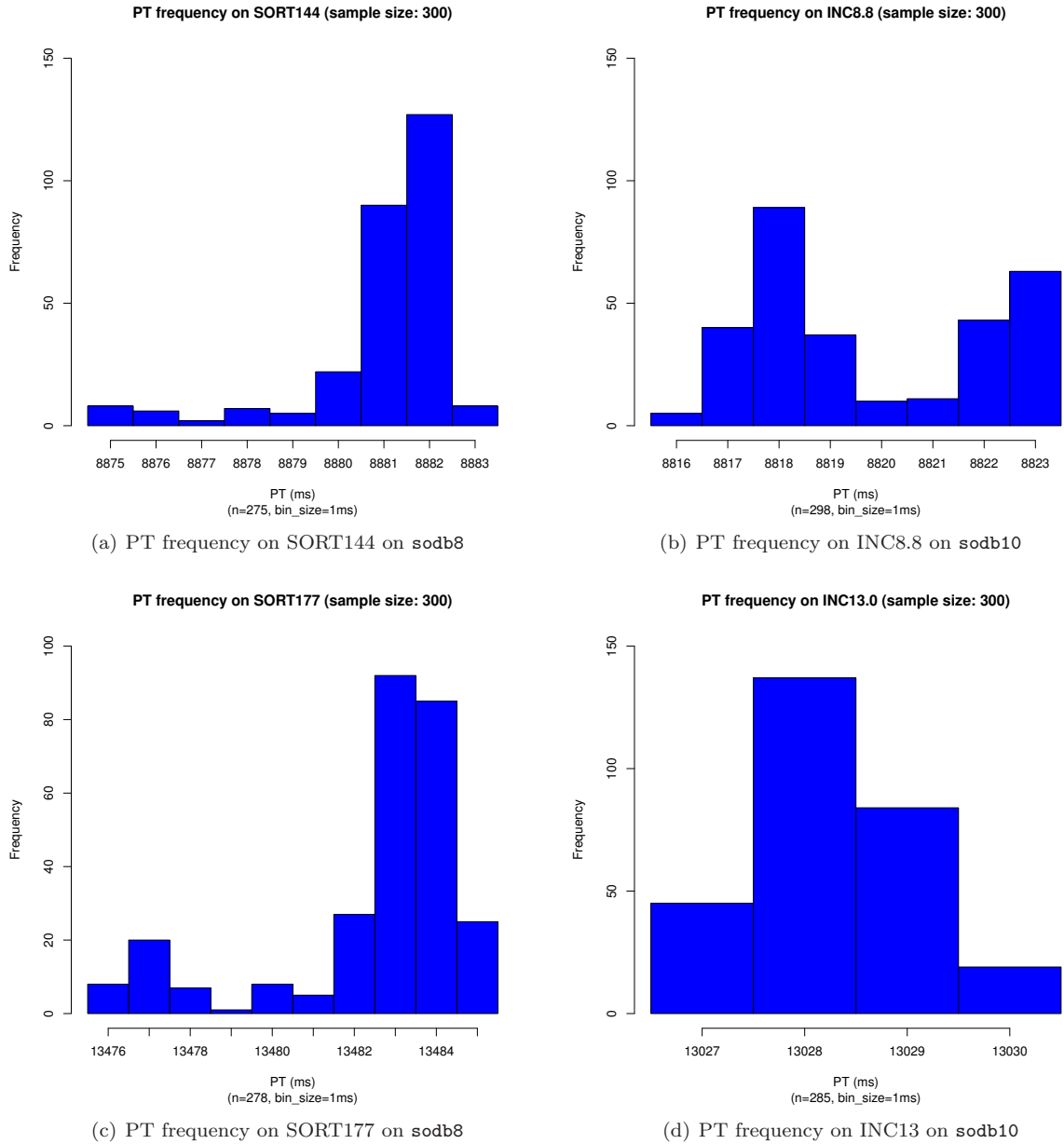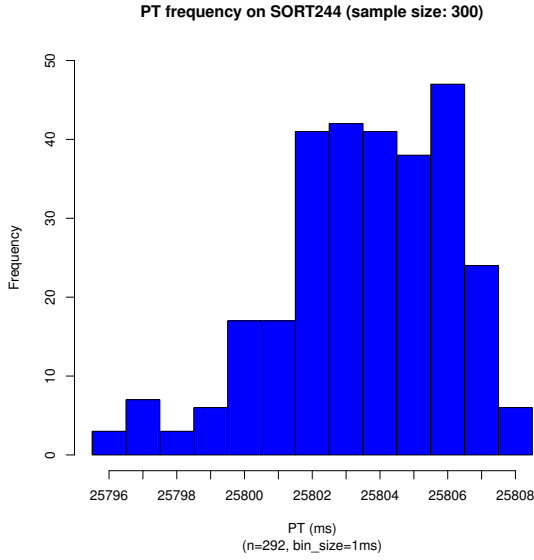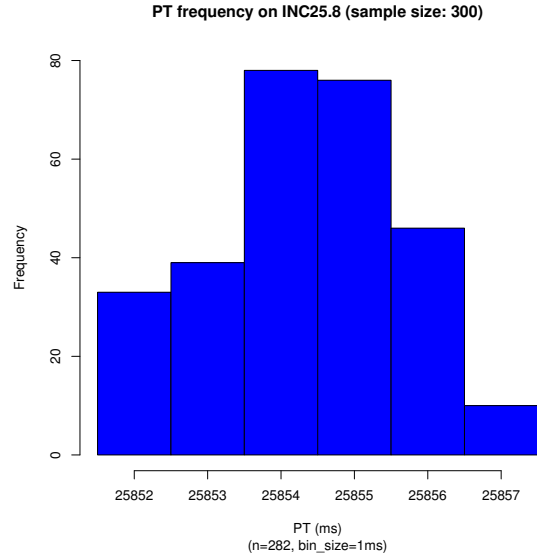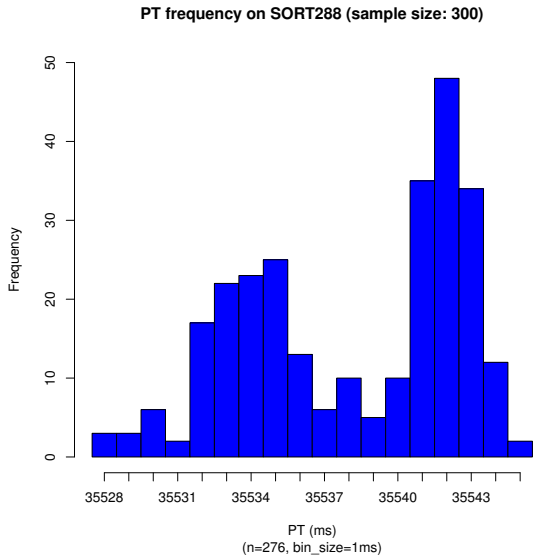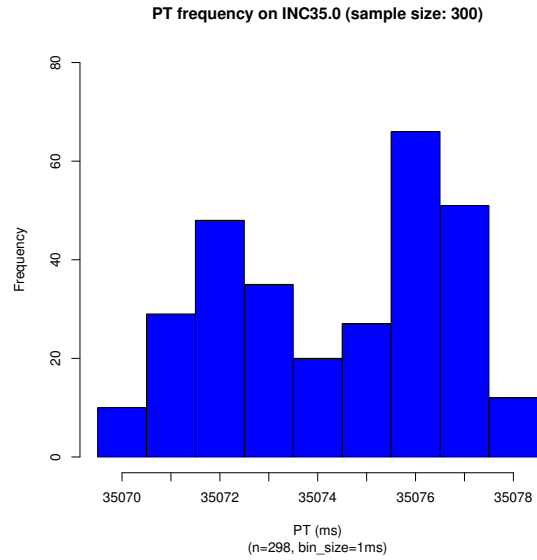
(c) PT frequency on SORT177 on `sodb8`

(d) PT frequency on INC13 on `sodb10`

Figure 6: PT Histograms I

PT frequency on SORT244 (sample size: 300)

PT frequency on INC25.8 (sample size: 300)

(a) PT frequency on SORT244 on `sodb8`

(b) PT frequency on INC25.8 on `sodb10`

PT frequency on SORT288 (sample size: 300)
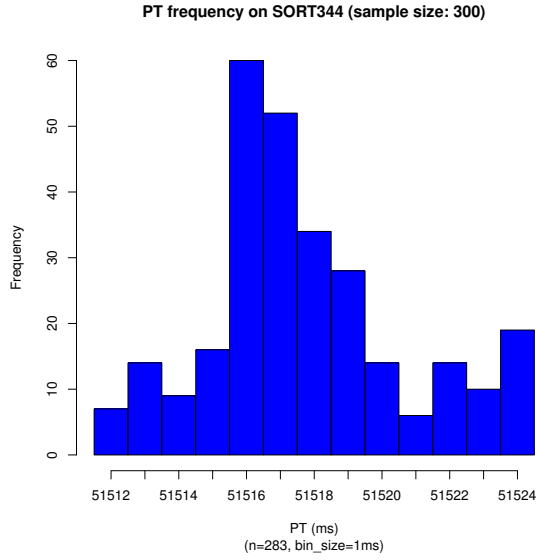
PT frequency on INC35.0 (sample size: 300)

(c) PT frequency on SORT288 on `sodb8`

(d) PT frequency on INC35 on `sodb10`

Figure 7: PT Histogram Comparison II

PT frequency on SORT344 (sample size: 300)

PT frequency on INC51.5 (sample size: 300)

(a) PT frequency on SORT344 on `sodb8`
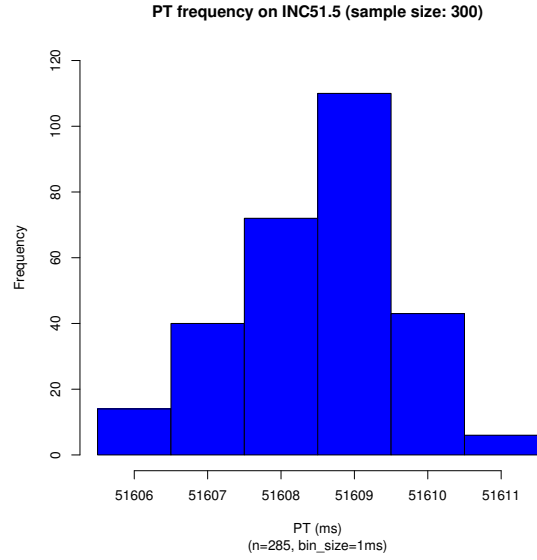
(b) PT frequency on INC51.5 on `sodb10`

Figure 8: PT Histogram Comparison III

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.