EECE 5640: High Performance Computing

Assignment 4

Jiayun Xin

NUID: 001563582

College of Engineering

Northeastern University Boston, Massachusetts

Fall, 2021

1.

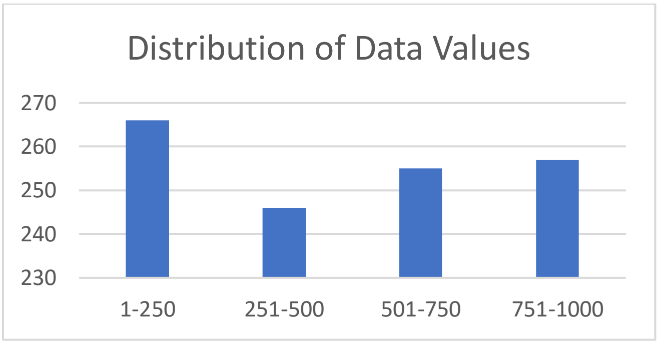
Number of darts	**Number of nodes**	**Execution time (s)**	**accuracy**
10000	4	1	0.990576
10000	8	3	0.990576
10000	16	6	0.990576
100000	4	1	0.999977
100000	8	3	0.999977
100000	16	9	0.999977
1000000	4	3	0.999977
1000000	8	8	0.999977
1000000	16	17	0.999977

Through the table above, 100000 darts with 4 nodes is a relatively ideal implementation to compute pi in case of guaranteeing accuracy. As we can see, same number of darts with different number of nodes will produce same accuracy but execution time; same number of nodes with different number of darts get different computation accuracy. Considering both execution time and accuracy, 100000 darts with 4 nodes is an ideal combination.

2.

a.

N (number of data values input)	Classes	Outputs	Running time (s)
1024	1	1024	1
1024	2	541 483	1
1024	4	277 254 256 236	1
1024 * 1024	1	1048576	1
1024 * 1024	2	522404 525139	1
1024 * 1024	4	261224 261666 262440 262203	3

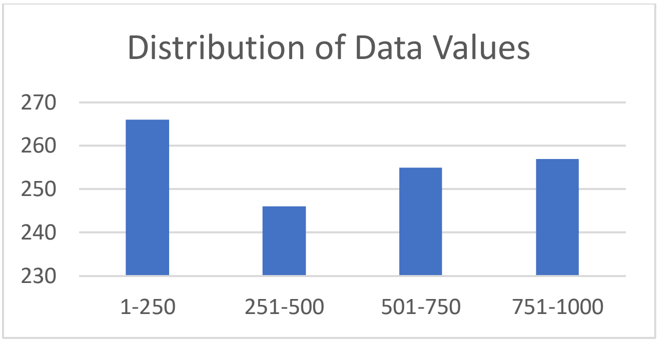


“N = 1024, classes = 4” is taken as an example to plot the results

b.

N (number of data values input)	Classes	Outputs	Running time (s)
1024	1	1024	1
1024	2	534 490	1
1024	4	266 246 255 257	1

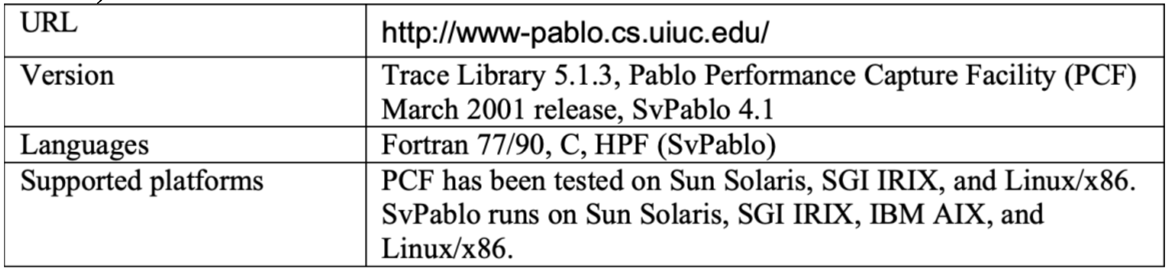
1024 * 1024	1	1048576	1
1024 * 1024	2	523093 524470	1
1024 * 1024	4	261636 261772 262618 261447	4



“N = 1024, classes = 4” is taken as an example to plot the results

c.
Through comparing the results between question a and b, there are just a little differences between them, which are the count of input number with different ranges and its variance distribution. Actually, the decisive factor of result distribution is random number input. The execution process between question a and b is different, which causes the bias of running time. Due to the limitation of time accuracy, question a is a little faster than question b. Question a groups the whole input number into several classes, each node counts its own number and gathers by summing up. Question b sends the whole number to several nodes and each node count numbers with specific range. Question b program does more loops, so that it is slower.

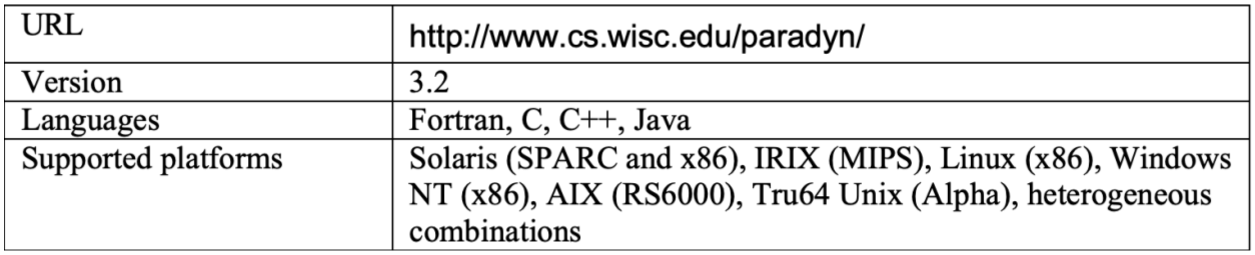
3.
Pablo performance analysis **tool is** used to record timestamped events and performance information about MPI calls, MPI I/O calls, and I/O requests[1]. This tool is specifically invoked by the Pablo Trace Library; all recorded events and information are SDDF (Self Defining Data Format).



URL	http://www-pablo.cs.uiuc.edu/
Version	Trace Library 5.1.3, Pablo Performance Capture Facility (PCF) March 2001 release, SvPablo 4.1
Languages	Fortran 77/90, C, HPF (SvPablo)
Supported platforms	PCF has been tested on Sun Solaris, SGI IRIX, and Linux/x86. SvPablo runs on Sun Solaris, SGI IRIX, IBM AIX, and Linux/x86.

Cited from: https://www.icl.utk.edu/files/publications/2001/icl-utk-90-2001.pdf

Paradyn is a tool which is used to measure the performance of parallel and distributed programs and includes two main sections (the Paradyn front-end and user interface, and the Paradyn daemons) [1]. It dynamically inserts instrumentation into a running application and displays performance in real-time.



URL	http://www.cs.wisc.edu/paradyn/
Version	3.2
Languages	Fortran, C, C++, Java
Supported platforms	Solaris (SPARC and x86), IRIX (MIPS), Linux (x86), Windows NT (x86), AIX (RS6000), Tru64 Unix (Alpha), heterogeneous combinations

Cited from: https://www.icl.utk.edu/files/publications/2001/icl-utk-90-2001.pdf

4.
In the paper "MPI on Millions of Cores", it mainly talked about three aspects of the scalability issue, which are MPI scalability issue, how to improve scalability and the parallel algorithms used to scale to millions of cores. In 2009, MPI has some issues at large scale, especially in terms of memory consumption. At that time, memory consumption and performance of all collective functions are two concentrations to improve scalability. To enable application scalability, the basic algorithms used by the application are the only solution.
Depending on the paper "A survey of MPI usage in the US exascale computing project" first published in 2018, application development, software technology and hardware technology are the three main areas that exascale computing focused at that time [2].
Depending on another paper, a new parallel programming model using coarse-grained and fine-grained parallelism over inter-node and intra-node is proposed to support massively parallel performance [3].

References:

[1] Moore, S., Cronk, D., London, K., & Dongarra, J. (2001, September). Review of performance analysis tools for MPI parallel programs. In European Parallel Virtual Machine/Message Passing Interface Users' Group Meeting (pp. 241-248). Springer, Berlin, Heidelberg.
[2] Bernholdt, D. E., Boehm, S., Bosilca, G., Gorentla Venkata, M., Grant, R. E., Naughton, T., ... & Vallee, G. R. (2020). A survey of MPI usage in the US exascale computing project. Concurrency and Computation: Practice and Experience, 32(3), e4851.
[3] Ashraf, M. U., Eassa, F. A., Albeshri, A. A., & Algarni, A. (2018). Toward exascale computing systems: An energy efficient massive parallel computational model. International Journal of Advanced Computer Science and Applications, 9(2).