Guo, Muyang
Wang, Yibo
Yu, Su

Worksheet 1

Programming Assignment 1

Introduction

In this programming assignment, parallel computing is used to sum N random numbers with P processors, assuming N is a multiplier of P. The program is coded in C++ with some basic MPI commands. This document reports the findings on speedup achieved at different N and P conditions.

Theory

Theoretically, the runtime to generate N random variables using P processors is

$$T_{rv,p}(n,p) = \Theta(\frac{n}{p}) \tag{1}$$

, assuming P is a factor of N. The runtime to compute sums is

$$T_{sum,p}(n,p) = \Theta(\frac{n}{p} + log(p)) \tag{2}$$

Thus, the total runtime for a parallel system is expected to be

$$T_p(n,p) = T_{rv,p}(n,p) + T_{sum,p}(n,p) = \Theta(\frac{n}{p}) + \Theta(\frac{n}{p} + log(p)) = \Theta(\frac{n}{p} + log(p)) \tag{3}$$

Comparing to a serial system, of which the total runtime is computed at $P = 1$,

$$T_s(n,1) = T_{rv,s}(n,1) + T_{sum,s}(n,1) = \Theta(n) + \Theta(n) = \Theta(n) \tag{4}$$

the speed up achieved is

$$S = \frac{T_s(n,1)}{T_p(n,p)} = \frac{\Theta(n)}{\Theta(\frac{n}{p} + log(p))} \tag{5}$$

Results

1. Runtime while changing the number of processors P from 1 up to 16.

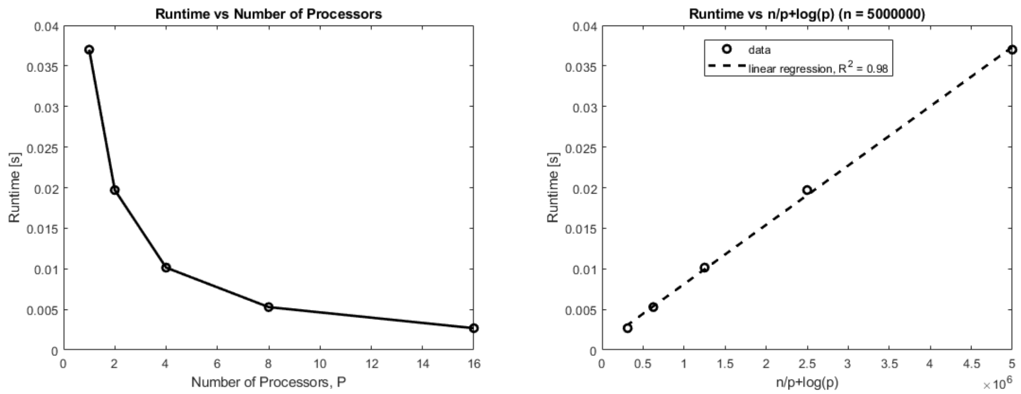

Figure 1: Runtime vs. P and Runtime vs. $\frac{N}{P} + log(P)$ (N $= 5 \times 10^6$)

The run time decreases with P. The right figure shows that this decrease follows $T_n(n,p) \propto \frac{n}{p} + log(p)$ for different P, with $R^2 = 0.98$. It matches the theory in Equation (3).

2. Runtime while changing N from 5×10^6 to 1.6×10^8

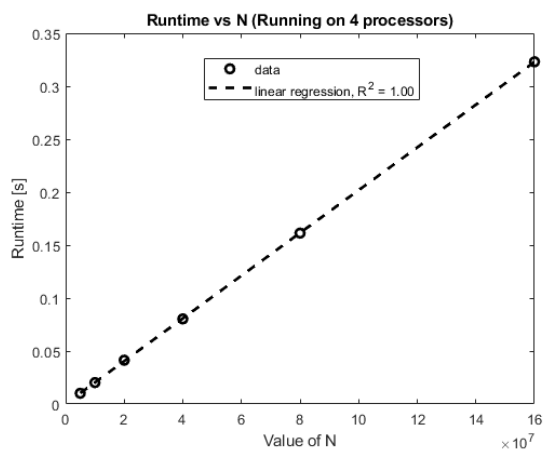

Figure 2: Runtime vs. N (P $= 4$)

The run time increases with N. Assuming no overflow, the run time $T_n(n,p) \propto n$ with $R^2 = 1.00$. This is consistent with Equation (3) because as $N \gg p$, $N/P \gg log(p)$, which approximates Equation (3) to $T_p(n,p) \approx \Theta(\frac{n}{p} + log(p))$.

Discussion

In summary, the runtime using parallel computing can be described by $T_p(n,p) = \Theta(\frac{n}{p} + log(p))$ in this assignment. Note that at $N \gg P$, the speed up is also approximated as $S \approx \frac{\Theta(n)}{\Theta(\frac{n}{p})} = \Theta(p)$. This indicates parallel computing is highly effective when $N \gg P$.