CSE531
Homework-4 Solution

1. The MPI program sums n numbers, and thus $T_S = n$.

$$\mathrm{T_p} = (\text{sum n/p numbers in each process}) + (\text{sum} + \text{sendrecv}) * (\text{stages of reduction})$$

$$\mathrm{T_p} = \frac{n}{P} + 2\log P$$

$$T_o = PT_P - T_S = (n + 2P\,log\,P) - n = 2P\,log\,P$$

Isoefficiency function: $W = K(2P\,log\,P)$

2.

a)

First, each processor performs a local prefix sum of its n/p numbers. In the second step, the p processors compute prefix sums of p numbers by using the last prefix sum resulting from the local computation on each processor. This step takes $(1 + \mathrm{t_s} + \mathrm{t_w}) \log p$ time. Finally, the result of the parallel prefix sums operation of the second step is added to all the n/p prefix sums of the first step at each processor. Therefore, $\mathrm{T_P} = 2(\mathrm{n/p}) + (1 + \mathrm{t_s} + \mathrm{t_w}) \log p$.

b)

```
double pdf[N], cdf[N];
MPI_Init(&argc, &argv);
int rank, procs;
MPI_Comm_rank(MPI_COMM_WORLD, &rank);
MPI_Comm_size(MPI_COMM_WORLD, &procs);
int L=rank*N/procs, R=(rank+1)*N/procs;
for (int i=L; i<R; i++) {
    pdf[i] = rand();
}
cdf[L] = pdf[L];
for (int i=L+1; i<R; i++) {
    cdf[i] = cdf [i-1] + pdf[i];
}
// MPI_Barrier(MPI_COMM_WORLD); // optional here
double base;
MPI_Scan(&cdf[R-1], &base, 1, MPI_DOUBLE, MPI_SUM, MPI_COMM_WORLD);
base -= cdf[R-1];
for (int i=L; i<R; i++) {
    cdf[i] += base;
}
for (int i=L; i<R; i++) {
      printf("process %d: cumulative_sum[%d] = %lf\n", rank, i, cdf[i]);
}
MPI_Finalize();
```

3.

b) AllReduce
c) AlltoAll (Personalized)
d) Bcast

4. The loop is performing a series of reductions on the same data, with the result being placed onto processor 0, 1, …, N, respectively. An `MPI_AllReduce` would have the same effect and execute in $log\ P$ time.