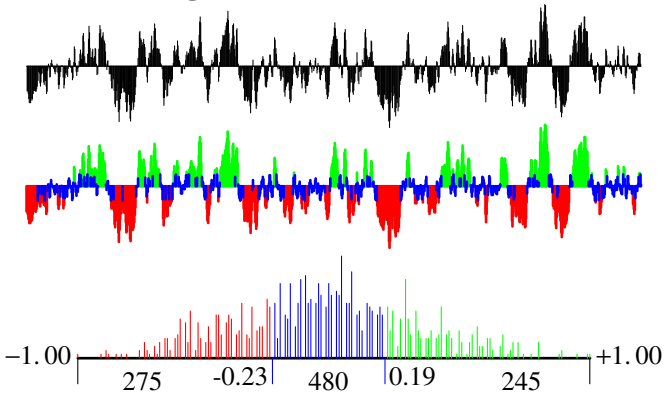
Early Introduction to Parallel Computing via Applications in Data Analytics

Sukhamay Kundu
Computer Science Department,
Louisiana State University, Baton Rouge, LA 70803
kundu@csc.lsu.edu

Why Consider Clustering of 1-Dim. Data

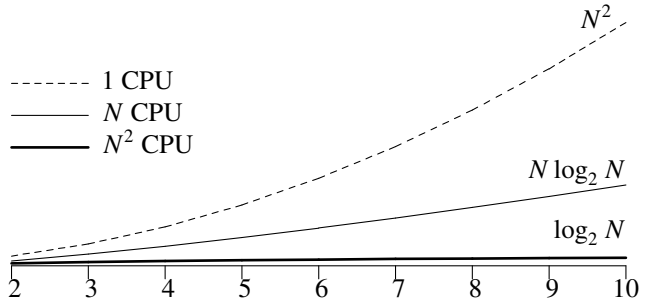
- Many important applications and has an elegant sequential algorithm that can be parallelized in multiple ways.

Low-Mid-High Classification of Machine Noise



The noise-frequencies and their optimum LMH-clustering; mid-part is asymmetric, about ¼ of range, has ½ data-points.

Performance of Parallel Solution



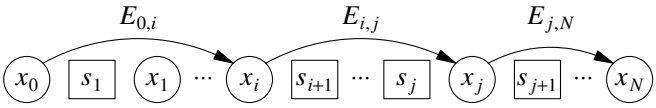
The Clustering Problem

- Scores S: $s_1 < s_2 < \cdots < s_N$ with frequencies $f_i = f(s_i)$.
- n = #(clusters) ≥ 2 to be formed; n much smaller than N.
- Error in cluster $C \subset S$, $E(C) = \sum f_p(s_p - \mu_C)^2 = \sum f_p s_p^2 - \mu_C^2 \sum f_p$, with sums over s_p in C and μ_C = aver(s_p in C).
- Positive weights $w_i > 0$ associated i^{th} cluster.

? Find clusters $C_1, C_2, \cdots, C_n$ which partition S and minimize total error $E = w_1 E(C_1) + w_2 E(C_2) + \cdots + w_n E(C_n)$.

* Each C_m is an interval $I_{i,j} = \{s_{i+1}, s_{i+2}, \cdots, s_j\}$ for $E = E_{\min}$.

A Special Shortest-Path Formulation



A 3-step x_0x_N-path; its cost = $w_1E_{0,i} + w_2E_{i,j} + w_3E_{j,N}$, and the intervals of its 3-clustering: $I_{0,i}$, $I_{i,j}$, and $I_{j,N}$.

- The complete acyclic digraph G with nodes $\{x_0, x_1, \cdots, x_N\}$ and links (x_i, x_j), $i < j$; cost $c(x_i, x_j) = E_{i,j}$.
- Optimal n-clusterings ↔ shortest n-step x_0x_N-paths.

Many Things to Compute

- $d_n(x_N)$ = length of n-step shortest x_0x_N-path.
- $d_{k,j} = d_k(x_j)$ = length of k-step shortest x_0x_j-path, $1 \le k < n$ and $k \le j < N$.
- $E_{i,j}$ = Cost of link (x_i, x_j), $0 \le i < j \le N$.
  - $SS_{i,j} = \sum f_p s_p^2$ over $i < p \le j$.
  - $\mu_{i,j}$ = aver(s_p in $I_{i,j}$) = $S_{i,j}/F_{i,j}$
  - $S_{i,j} = \sum f_p s_p$ and $F_{i,j} = \sum f_p$ over $i < p \le j$.

Parallel Computation of all $F_{i,j}$, $S_{i,j}$, etc.

- $F_{i,j+1} = f_{i+1} + f_{i+2} + \cdots + f_{j+1} = F_{i,j} + f_{j+1}$ for $i < j$.
- For fixed i: $F_{i,j}$'s are prefix-sums of $[f_{i+1}, f_{i+2}, \cdots, f_N]$.
- Time $O(\log N)$ for all $F_{i,j}$ using N. $N/2 = N^2/2$ CPUs.
- Similarly for $S_{i,j}$'s and $SS_{i,j}$'s.

Parallel Computation of $d_{j,j}$'s and $d_{n,N}$

- $d_{1,j} = w_1 E_{0,j}$ for $1 \le j$.
- $d_{k+1,j} = \min\{d_{k,i} + w_{k+1}E_{i,j} : k \le i < j\}$ for $k + 1 < n$.
- $d_{n,N} = \min\{d_{n-1,i} + w_n E_{i,N} : n - 1 \le i < N\}$.
- Time $O(n \log N)$ to compute all min. using $N^2/2$ CPUs.

Conclusion

The 1-dimensional clustering problem provides a good vehicle for an early introduction to parallel computing and, in particular, two important parallelization techniques.