Towards Faster Execution of Ensemble ML Bootstrap Based Techniques

Vinay Gavirangaswamy, Ajay Gupta, Hisham Saleh, Dept. Computer Science, Western Michigan University; Vasilije Perovic, Dept. of Mathematics, University of Rhode Island
Decision Making, A Fundamental Aspect of life...

- Studied using tools for a specific competency e.g., surveys, computer tasks / games, etc.
- Finds application in life science (Stanford University HAI COVID19 and AI), Marketing, and Finance
- Analysis broadly classified for group / individual, aggregate / differential behaviors
- We applied Ensemble Clustering to tasks data from BART, CUPS, and IGT
Parallel Implementation on Distributed Memory System

Run Times, Speedup and Efficiency of Ensemble Clustering after Parallelization 1000 Participants’ Data

<table>
<thead>
<tr>
<th>#MPI threads</th>
<th>1</th>
<th>2</th>
<th>4</th>
<th>8</th>
<th>12</th>
<th>24</th>
<th>48</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exec Time (in sec)</td>
<td>14237</td>
<td>7198</td>
<td>3603</td>
<td>1808</td>
<td>1251</td>
<td>645</td>
<td>317</td>
</tr>
<tr>
<td>Speedup</td>
<td>1</td>
<td>1.97</td>
<td>3.95</td>
<td>7.87</td>
<td>11.37</td>
<td>22.04</td>
<td>44.79</td>
</tr>
<tr>
<td>Efficiency</td>
<td>1</td>
<td>0.98</td>
<td>0.98</td>
<td>0.98</td>
<td>0.94</td>
<td>0.91</td>
<td>0.93</td>
</tr>
</tbody>
</table>

Execution Times And Speedup for Parallelized RDM Reinforcement Learning Algorithm
Parallel Implementation on Shared Memory System

Speedup achieved for running Ensemble Clustering on CPU vs. CPU-GPU

Run times for unordered vs. batch reconfigured model sequence execution
Parallel Implementation on Shared Memory System (Contd.)

Speedup achieved for running Ensemble Clustering on **CUBLAS-ARPACK**

<table>
<thead>
<tr>
<th>#Participants</th>
<th>Preprocessing</th>
<th>Ensemble Creation</th>
<th>Ensemble Extraction</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>512</td>
<td>0.000002</td>
<td>0.027031</td>
<td>0.200994</td>
<td>0.228027</td>
</tr>
<tr>
<td>1024</td>
<td>0.000004</td>
<td>0.065028</td>
<td>0.829694</td>
<td>0.894726</td>
</tr>
<tr>
<td>2048</td>
<td>0.000008</td>
<td>0.179218</td>
<td>1.74629185</td>
<td>1.9255178</td>
</tr>
<tr>
<td>4096</td>
<td>0.000025</td>
<td>0.353409</td>
<td>4.5596992</td>
<td>4.9134039</td>
</tr>
<tr>
<td>8192</td>
<td>0.000097</td>
<td>0.918421</td>
<td>16.4327205</td>
<td>17.351238</td>
</tr>
<tr>
<td>10240</td>
<td>0.000197</td>
<td>1.4468</td>
<td>25.6589273</td>
<td>27.105924</td>
</tr>
<tr>
<td>11264</td>
<td>0.000228</td>
<td>1.762412</td>
<td>31.1415162</td>
<td>32.904156</td>
</tr>
<tr>
<td>12288</td>
<td>0.000216</td>
<td>1.773678</td>
<td>36.1456167</td>
<td>37.919510</td>
</tr>
<tr>
<td>13312</td>
<td>0.000254</td>
<td>8.05293</td>
<td>42.3747132</td>
<td>50.427897</td>
</tr>
<tr>
<td>14336</td>
<td>0.000309</td>
<td>2.353111</td>
<td>48.4853123</td>
<td>50.838732</td>
</tr>
</tbody>
</table>

Speedup achieved for running Ensemble Clustering on **BLAS-ARPACK**

<table>
<thead>
<tr>
<th>#Participants</th>
<th>Preprocessing</th>
<th>Ensemble Creation</th>
<th>Ensemble Extraction</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>512</td>
<td>0.008813</td>
<td>0.920119</td>
<td>0.596379</td>
<td>1.52531</td>
</tr>
<tr>
<td>1024</td>
<td>0.034078</td>
<td>3.890402</td>
<td>1.2575</td>
<td>5.18198</td>
</tr>
<tr>
<td>2048</td>
<td>0.133376</td>
<td>24.678988</td>
<td>3.91411</td>
<td>28.7264</td>
</tr>
<tr>
<td>4096</td>
<td>0.5259915</td>
<td>178.714399</td>
<td>10.220642</td>
<td>189.460</td>
</tr>
<tr>
<td>8192</td>
<td>2.0748</td>
<td>1325.326938</td>
<td>36.832031</td>
<td>1364.23</td>
</tr>
<tr>
<td>10240</td>
<td>3.240139</td>
<td>2550.207621</td>
<td>57.5115</td>
<td>2610.95</td>
</tr>
<tr>
<td>11264</td>
<td>3.89515</td>
<td>3374.181781</td>
<td>69.8000851</td>
<td>3447.87</td>
</tr>
<tr>
<td>12288</td>
<td>4.637481</td>
<td>4361.379918</td>
<td>81.016194</td>
<td>4447.03</td>
</tr>
<tr>
<td>13312</td>
<td>5.440463</td>
<td>5515.57373</td>
<td>94.977989</td>
<td>5615.99</td>
</tr>
<tr>
<td>14336</td>
<td>6.315224</td>
<td>6858.803793</td>
<td>108.674186</td>
<td>6973.79</td>
</tr>
</tbody>
</table>
Ensemble Clustering, optimizing for redundant computations (RC)


Highlighted boxes denote an example of steps in ensemble clustering that lead to redundant computations.
Example: Dist-Program

<table>
<thead>
<tr>
<th>Original Machine Assembly Code with Color Coding for Redundancies</th>
<th>An Example Optimized Assembly Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>mov DWORD PTR [rbp-60], eax</td>
<td>mov DWORD PTR [rip-60], eax</td>
</tr>
<tr>
<td>mov DWORD PTR [rip-60], 0</td>
<td>mov DWORD PTR [rip-60], 0</td>
</tr>
<tr>
<td>mov edx, edx</td>
<td>mov edx, edx</td>
</tr>
<tr>
<td>call subtract(int, int)</td>
<td>call subtract(int, int)</td>
</tr>
<tr>
<td>mov r12d, r12d</td>
<td>mov r12d, r12d</td>
</tr>
<tr>
<td>mov eax, 0</td>
<td>mov eax, 0</td>
</tr>
<tr>
<td>mov rbp, rbp</td>
<td>mov rbp, rbp</td>
</tr>
<tr>
<td>add rsp, 48</td>
<td>add r12d, r12d</td>
</tr>
<tr>
<td>pop rbx</td>
<td>pop rbx</td>
</tr>
<tr>
<td>pop r12</td>
<td>pop r12</td>
</tr>
<tr>
<td>pop rbp</td>
<td>pop rbp</td>
</tr>
<tr>
<td>ret</td>
<td>ret</td>
</tr>
</tbody>
</table>

Redundancies in code due to numerical operations applied over same data
Problem Formulation

Distinct clustering algorithms

Bootstrapped data sets from the main data set $D$ $(t=1,2,...,k)$

Computational unit (cu) – an atomic unit consisting of both data and mathematical computations operating on the data set.
Example: Dist-Program

1. **Input:** $a, b, c \in \mathbb{R}^2$.
2. **Output:**
   - $\text{dac}$ - squared Euclidean dist. b/w $a$ and $c$,
   - $\text{davgc}$ - squared Euclidean dist. b/w $(a+b)$ and $c$.
3. $\text{dac} = (a_1-c_1) \cdot (a_1-c_1) + (a_2-c_2) \cdot (a_2-c_2)$
4. $\text{davgc} = (a_1-c_1+b_1) \cdot (a_1-c_1+b_1) + (a_2-c_2+b_2) \cdot (a_2-c_2+b_2)$

<table>
<thead>
<tr>
<th># elt's</th>
<th>subsets of $D^{(t)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>$D^{(t)}_1 = \emptyset$</td>
</tr>
<tr>
<td>1</td>
<td>$D^{(t)}_2 = {a}$, $D^{(t)}_3 = {b}$, $D^{(t)}_4 = {c}$,</td>
</tr>
<tr>
<td>2</td>
<td>$D^{(t)}_5 = {a, b}$, $D^{(t)}_6 = {a, c}$, $D^{(t)}_7 = {b, c}$,</td>
</tr>
<tr>
<td>3</td>
<td>$D^{(t)}_8 = {a, b, c}$</td>
</tr>
</tbody>
</table>

### Nontrivial cu’s with repetitions

$m_1^{6,t} = a_1 - c_1$  $\rightarrow$  $m_3^{6,t} = m_1^{6,t} \cdot m_2^{6,t}$

$m_2^{6,t} = a_1 - c_1$  $\rightarrow$  $m_4^{6,t} = m_2^{6,t} \cdot m_5^{6,t}$

$m_3^{6,t} = a_2 - c_2$  $\rightarrow$  $m_5^{6,t} = m_3^{6,t} + m_6^{6,t}$

$m_4^{6,t} = a_2 - c_2$  $\rightarrow$  $m_6^{6,t} = m_4^{6,t} \cdot m_5^{6,t}$

$m_5^{6,t} = a_1 - c_1$  $\rightarrow$  $m_7^{6,t} = m_3^{6,t} + m_6^{6,t}$

$m_6^{6,t} = a_1 - c_1$  $\rightarrow$  $m_8^{6,t} = m_7^{6,t} + b_1$

$m_7^{6,t} = a_2 - c_2$  $\rightarrow$  $m_9^{6,t} = m_6^{6,t} + b_2$  $\rightarrow$  $m_{10}^{6,t} = m_9^{6,t} + b_2$

Note: $m_{j,t} = j^{th}$ computational unit acting on subset $D_i^{(t)}$
Example: Dist-Program

Nontrivial cu’s with repetition

\[
\begin{align*}
 m_1^{6,t} &= a_1 - c_1 \\
 m_2^{6,t} &= a_1 - c_1 \\
 m_4^{6,t} &= a_2 - c_2 \\
 m_5^{6,t} &= a_2 - c_2 \\
 m_7^{6,t} &= a_1 - c_1 \\
 m_8^{6,t} &= a_1 - c_1 \\
 m_9^{6,t} &= a_2 - c_2 \\
 m_{10}^{6,t} &= a_2 - c_2 \\
\end{align*}
\]

\[
\begin{align*}
 m_3^{6,t} &= m_1^{6,t} \cdot m_2^{6,t} \\
 m_6^{6,t} &= m_4^{6,t} \cdot m_5^{6,t} \\
 m_7^{6,t} &= m_3^{6,t} + m_6^{6,t} \\
 m_8^{6,t} &= m_4^{6,t} + b_1 \\
 m_9^{6,t} &= m_5^{6,t} + b_2 \\
 m_{10}^{6,t} &= m_6^{6,t} + b_2 \\
\end{align*}
\]

\[M = \text{collection of all nontrivial computational units along with their dependencies}\]

Nontrivial cu’s without repetitions

\[
\begin{align*}
 m_1^{6,t} &= a_1 - c_1 \\
 m_3^{6,t} &= m_1^{6,t} \cdot m_2^{6,t} = m_1^{6,t} \cdot m_1^{6,t} \\
 m_4^{6,t} &= a_2 - c_2 \\
 m_6^{6,t} &= m_4^{6,t} \cdot m_5^{6,t} = m_4^{6,t} \cdot m_4^{6,t} \\
 m_7^{6,t} &= m_3^{6,t} + m_6^{6,t} \\
 m_8^{6,t} &= m_4^{6,t} + b_1 \\
 m_9^{6,t} &= m_5^{6,t} + b_2 \\
 m_{10}^{6,t} &= m_6^{6,t} + b_2 \\
\end{align*}
\]

\[
\begin{align*}
 \text{Original cu} & \quad \text{Relabeled cu} & \quad \text{Dependency} \\
 m_1^{6,t} &= a_1 - c_1 & m_1 \quad \text{none} \\
 m_3^{6,t} &= m_1^{6,t} \cdot m_2^{6,t} = m_1^{6,t} \cdot m_1^{6,t} & m_2 \quad m_1 \\
 m_4^{6,t} &= a_2 - c_2 & m_3 \quad \text{none} \\
 m_6^{6,t} &= m_4^{6,t} \cdot m_5^{6,t} = m_4^{6,t} \cdot m_4^{6,t} & m_4 \quad m_3 \\
 m_7^{6,t} &= m_3^{6,t} + m_6^{6,t} & m_5 \quad m_2, m_4 \\
 m_8^{6,t} &= m_4^{6,t} + b_1 = m_1^{6,t} + b_1 & m_6 \quad m_1 \\
 m_9^{6,t} &= m_5^{6,t} + b_2 = m_4^{6,t} + b_2 & m_7 \quad m_6 \\
 m_{10}^{6,t} &= m_6^{6,t} + b_2 = m_4^{6,t} + b_2 & m_8 \quad m_3 \\
\end{align*}
\]
### Nontrivial cu’s with repetition (Dist–Program)

<table>
<thead>
<tr>
<th>Original cu</th>
<th>Relabeled cu</th>
<th>Dependency</th>
</tr>
</thead>
<tbody>
<tr>
<td>$m^6_{1,t}$</td>
<td>$m^6_{2,t}$</td>
<td>none</td>
</tr>
<tr>
<td>$m^6_{2,t}$</td>
<td>$m^6_{1,t}$</td>
<td>$m^6_{1,t}$</td>
</tr>
<tr>
<td>$m^6_{4,t}$</td>
<td>$a_2 - c_2$</td>
<td>none</td>
</tr>
<tr>
<td>$m^6_{5,t}$</td>
<td>$m^6_{4,t}$</td>
<td>$m^6_{4,t}$</td>
</tr>
<tr>
<td>$m^6_{6,t}$</td>
<td>$m^6_{5,t}$</td>
<td>$m^6_{5,t}$</td>
</tr>
<tr>
<td>$m^6_{7,t}$</td>
<td>$m^6_{3,t} + m^6_{6,t}$</td>
<td>$m^6_{6,t}$</td>
</tr>
<tr>
<td>$m^6_{8,t}$</td>
<td>$m^6_{7,t} + b_1$</td>
<td>$m^6_{7,t} + b_1$</td>
</tr>
<tr>
<td>$m^6_{9,t}$</td>
<td>$m^6_{8,t}$</td>
<td>$m^6_{8,t}$</td>
</tr>
<tr>
<td>$m^6_{10,t}$</td>
<td>$m^6_{9,t}$</td>
<td>$m^6_{9,t}$</td>
</tr>
</tbody>
</table>

### Capturing Interdependencies between cu’s

- $\tilde{n}_4 = (0, 0, 1, 0, 0, 0, 0, 0, 0, 0)$
- $\tilde{n}_5 = (0, 1, 0, 1, 0, 0, 0, 0, 0, 0)$
- $\tilde{n}_6 = (1, 0, 0, 0, 0, 0, 0, 0, 0, 0)$
- $\tilde{n}_{10} = (0, 0, 0, 0, 0, 0, 1, 0, 1, 0)$

**Interaction graph capturing interdependencies between $\tilde{m}$’s**

(Directed acyclic graph – DAG)
Algorithm: Reduction in Redundant Computations (RRC)

**Step 1:** Decompose Ensemble Clustering ML algorithm into collection of computational units $\mathcal{M}$.

**Step 2:** Identify and collect all distinct computational units from $\mathcal{M}$ into $\tilde{\mathcal{M}}$ along with interdependencies.

**Step 3:** Find an aggregation of cu’s, $\mathcal{C}$, with execution cost lower than that of $\mathcal{M}$.

**Step 4:** FTiP, Floor Tile Planning - Find an efficient execution plan for $\mathcal{C}$ on the compute resource.
Step 3 (RRC): Find an aggregation of cu’s, $\mathcal{C}$, with execution cost lower than that of $M$

The choice of aggregation of cu’s $\mathcal{C}$ is limited by computer resources, some of which are as follow:

- Machine with *restrictive compute* and *infinite memory*
- Machine with *infinite compute* and *restrictive memory*
- Machine with *restrictive compute* and *restrictive memory*
Definition: $r$-$s$ Set Cover

Let $\widetilde{M} = \{\widetilde{m}_1, \widetilde{m}_2, \ldots, \widetilde{m}_\omega\}$ be a set of distinct cu’s and $\widetilde{G}$ a directed acyclic (interaction) graph corresponding to $\widetilde{M}$. An ordered list $\mathbf{C} = (A_1, A_2, \ldots, A_\delta)$, $A_i \subseteq \widetilde{M}$, $i = 1, 2, \ldots, \delta$, is said to be an $(r, s)$ set cover of $\widetilde{M}$, where $r$ and $s$ are positive integers, if for all $i = 1, 2, \ldots, \delta$ sets $A_i$ are nonempty and the following conditions are satisfied:

(i) $\bigcup_{i=1}^{\delta} A_i = \widetilde{M}$,

(ii) cardinality of $A_i$ is at most $r$, i.e., $|A_i| \leq r$, and

(iii) an arbitrary $m_\alpha \in A_i$ can be executed given a subset, possibly empty, of tasks from $A_i, A_{i-1}, \ldots, A_{i-s}$. 
Connection with the Directed Bandwidth Problem

**Question:**
Given $r \geq 1$, what is the smallest $s$ for which $(r,s)$ set cover of $\bar{M}$ exists?

**Def:** Given a directed acyclic graph $\bar{G}$ associated with a collection of cu’s in $\bar{M}$, the *directed bandwidth of $\bar{G}$*, $\overrightarrow{Bw}(\bar{G})$, is given by

$$\overrightarrow{Bw}(\bar{G}) := \min \left\{ Bw(\bar{G}, \sigma) : \sigma \text{ is a layout of } \bar{G} \text{ and } \sigma(m_\alpha) < \sigma(m_\beta) \text{ for all directed arcs from } m_\alpha \text{ to } m_\beta \right\}.$$
Simulation Study – Experimental Setup

- Based on ensemble clustering with only $k$-means with variation for data and number of clusters
- Variation in $\widehat{M}$ studied for occurrence of RR under different random distributions
- Built a generic FTiP – Simulator from scratch for, studying RR computation across different ML models
- Simulator written using python, neo4j graph database
Simulation Study – Results

- DAGs resulting from cu’s in (a) $M$, (b) $M_a$, and (c) $\tilde{M}$, with $RR(M) \leq RR(M_a) \leq RR(\tilde{M})$
- Variation in $\tilde{M}$ i.e. $M_a$ generated for RR under normally distributed randomization
Simulation Study – Results (Contd.)

• Metrics on DAGs resulting from cu’s in $M$, $M_a$, and $M$ for $|D^{(t)}| = 8, 16, 32$, corresponds to a variation in RR
• Results clearly indicate for presence of lower and upper bounds
• Clear evidence for existence of opportunities to optimize
CONCLUSIONS & FUTURE WORK

• Explored avenues for further improvements in computational performance in EM algorithms
• We work at the intersection of high-performance computation, compiler, and reinforcement learning algorithms
• Proposed a novel theoretical framework which can help us identify previously undetected redundancies
• Solve the RRC problem for a restricted case
• Extend simulation experiments for various randomization of RR
• Study a wider ML models for RRC
Thanks to NIH, NSF, WMU and URI for partial support. Thanks to reviewers for valuable feedback that resulted in improved paper.

Q&A