Coeus: Clustering (A)like Patterns For Practical Machine Intelligent Hybrid Memory Management

Thaleia Dimitra Doudali (IMDEA Software Institute)*
Ada Gavrilovska (Georgia Tech)

@ CCGrid 2022

* Work done while at Georgia Tech (PhD).
The Era of Data

“More than 65 ZB of data will be created, captured, copied, and consumed in the world this year.”

Source: International Data Corporation, March 2021.

Data Analytics Pipeline

Need for speed and massive storage capacities!
The Era of Heterogeneous Hardware

Examples of other heterogeneous memory technologies.

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Technology</th>
<th>Hardware Vendors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low Latency</td>
<td>MRAM</td>
<td></td>
</tr>
<tr>
<td>High Bandwidth</td>
<td>HBM</td>
<td></td>
</tr>
<tr>
<td>Persistence</td>
<td>PMEM</td>
<td></td>
</tr>
</tbody>
</table>

Source: memverge.com
Source: intel.com
Hybrid Memory Management is Complex

It is a complex decision mix to manage the data allocated across memories.

E.g., Which / How much / Where / When to move data?

Why do we need more intelligent systems?

- ZBs
- Application data sizes
- Complex data access patterns
- Exploded system parameter space
- Hard to balance
- performance
- practicality

Hybrid Memory

System Level

Hardware Level

Resource Management

Software

Runtime

Application Level

Data Allocated across Memories

Dynamic Data Movements!
Machine Intelligent Hybrid Memory Management

The Vision.

Applications

System-level Resource Manager

ML training

Data Access Monitoring

Re-train?

prediction

Trained Models

Resource Management

no

Practical

Hybrid Memory

Heterogeneous Hardware

practical
Machine Intelligent Hybrid Memory Management

Laying the grounds for the practical integration of ML.

- Applications
  - Exploded Data Sizes
    - Which data needs ML?
      - Where to use ML?
        - Which ML method?
          - What to predict?
            - How to practically use ML?
  - System-level Resource Manager
    - Resource Management
      - ML training
        - Trained Models
      - no
        - How to practically use ML?
  - Heterogeneous Hardware
    - Hybrid Memory

- Machine Intelligent Hybrid Memory Management
- Laying the grounds for the practical integration of ML.
Kleio is a hybrid memory page scheduler with machine intelligence. [Best Paper Award Finalist at HPDC 2019.]

**System Design of Kleio**

Kleio extends existing lightweight hybrid memory management with the necessary amount of machine intelligence to boost application performance.

1. Page Access Monitoring

2. Page Hotness Prediction

   - ML-based predictions (Per page RNN models)
   - History-based predictions

3. Page Migration Selection

   - Calculate hot vs. cold pages

Applications

- Page Selector
  - small subset
  - Pages for ML
  - bigger subset
  - Pages for History

System-level Resource Manager

- Page Hotness

Heterogeneous Hardware

- Page Migrations
  - Hybrid Memory
  - cold pages
  - hot pages

Not all pages “need” ML.
How “Small” is the Page Subset for ML?

Kleio’s evaluation trains is limited to training 100 RNNs.

The current design of Kleio (1 model per page) allows for a big variability in the number of ML models (resource and time overheads) required to maximize application performance.
Scaling ML-based Management

Can we scale the RNN models to learn patterns of more pages?
Clustering? Let’s use Machine Learning!

**K-means** is a very popular unsupervised learning data clustering method.

**Challenge 1:** Decide the number of clusters ($k$).

Lower inertia means higher similarity within a cluster. Inertia = 0: data in cluster is identical.

- No single value of $k$ works best for all.
- It is not obvious which value of $k$ to choose.
Clustering? Let’s use Machine Learning!

Let’s assume we know how many clusters to create.

**Challenge 2:** What input to use for RNN training over a cluster of pages?

**Option 1:** input data from all pages.

- Number of Pages
- Learning Overheads

**Option 2:** input data from the “centroid” page.

- Cluster Dissimilarity
- ML Prediction Accuracy

*Ineffective clustering may lead to large learning overheads and reduced prediction accuracy levels.*

Clustering with ML methods introduces non trivial complexity and overheads.

How can we Keep it Smart but Simple?
Insight on Similar vs. Identical Patterns

Let’s take a step back and observe the per page patterns.

*Input to the ML model is the sequence of page hotness across periods of time.*

![Memory Access Trace of backprop](image)

**Short Periods**
Sequences are slightly different.

**Long Periods**
Sequences are identical.

*It is all about the right granularity!*
Which is the Right Granularity?

 Longer periods result in more pages having identical patterns of page access hotness across time.

 Longer periods may result in insufficient data movements and impact application performance.

 Periods that align with the page reuse distance, maximize performance.

*Page Reuse Distance = The time gap between two accesses to the same page.

Insight from Cori: Dancing to the Right Beat of Periodic Data Movements over Hybrid Memory Systems. [IPDPS 2021]

Cori is a lightweight tuning solution for hybrid memory page schedulers, that we will use to determine the “right granularity”.
**System Design of Coeus**

*Coeus* is a page grouping mechanism for machine intelligent page schedulers.

---

**Pattern Analyzer**

- **Memory Access Trace of backprop**
- **Page Access Patterns**
- **Page Clustering**

**Pattern Selector**

- **Kleio’s Page Selector**
- **Unique Page Patterns**

---

**Page Cluster** = **Pattern** = **Page unique identity** = **hotness sequence across periods**
Scaling ML to More Pages and Improving Performance

Baseline: Kleio

Almost 3x more pages for the same number of ML models

Coeus

2x from tuned periodicity.
1x from more pages under ML.

Baseline: Kleio

The higher, the better
Reducing Runtime Overheads of ML-based Management

Coeus enables:

- Quick page clustering process.
- Fewer ML models deployed.

Baseline: Coeus

3x reduction

The lower, the better

Diagram showing Runtime Overheads with Coeus and Kleio compared.
Summary of Coeus

Takeaways:

• Keep it Smart, but Simple.
  • ML is not always necessary.

• It is all about the right granularity.
  • For patterns and performance.

• Coeus reduces ML overheads by 3x.
  • Quick clustering.
  • Reduces total number of ML models deployed.

• Improves application performance by 3x.
  • From allowing more pages to be managed with ML.
  • From the tuned periodicity of the management.

Greek Trivia: According to the ancient Greek mythology, Coeus was the titan god of intelligence.