SCIENCE CHINA
Information Sciences

• LETTER •

April 2024, Vol. 67, Iss. 4, 149102:1–149102:2
https://doi.org/10.1007/s11432-023-3925-2

A smart hybrid memory scheduling approach using neural models

Yanjie ZHEN[1], Huijun ZHANG[2], Yongheng DENG[1], Weining CHEN[1], Wei GAO[1], Ju REN[1] & Yu CHEN[1*]

[1]*Department of Computer Science and Technology, Tsinghua University, Beijing* 100084*, China;*
[2]*China Huaneng Clean Energy Research Institute, Beijing* 102209*, China*

Received 17 May 2023/Revised 3 September 2023/Accepted 26 December 2023/Published online 27 March 2024

Traditional dynamic random access memory is limited by its low capacity and high cost because of the increasing demand for the memory capacity of applications. Hybrid memory architectures, including new memory hardware and memory disaggregation, provide a feasible approach to expanding the memory capacity. However, different memory components within a hybrid memory system exhibit different attributes. It is necessary to implement an efficient dynamic data scheduler to predict the hotness of future data accesses and migrate hot data to fast memory on a timely basis while evicting cold data to slow memory.

Most existing schedulers [1, 2] use a history policy, assuming that the hot pages in the current epoch will also be the hot pages in the next epoch. However, this policy's predictive capability is limited due to the short address sequence used for prediction. Our experiments show that the history scheduler's effectiveness is only 42.6% of the optimal scheduler's. Several studies, such as Kleio [3], have explored the utilization of neural models for page hotness prediction to enhance the effectiveness of schedulers. These schedulers treat page hotness prediction as a time-series prediction problem and predict at the granularity of individual pages, deploying an individual recurrent neural network for each page. However, an application can encompass hundreds of thousands of pages, as detailed in Appendix A. The significant training and inference cost makes deploying such a vast number of models infeasible in real-world systems. Detailed experimental evidence of the limitations of existing schedulers is provided in Appendix B.

Therefore, it is hoped to propose a novel neural-model-based scheduler with high effectiveness and low cost. To accomplish this goal, we propose a collective-page predictor that enhances the current individual-page prediction strategy by predicting all pages collectively with a single neural model. However, predicting all pages with a single neural model is challenging due to the class explosion problem as the number of pages grows. We observe that some pages exhibit similar access patterns, leading to similar migration behaviors. Based on this observation, we propose a clustering-based similar page identifier to identify pages with similar access patterns and manage them as a cluster. In addition, we propose a page migration quantity estimator to avoid unnecessary page migrations. Overall, in this study, we propose a novel smart page scheduling (SmartS) based on neural models for hybrid memory management. SmartS not only achieves high effectiveness but also significantly reduces the cost of neural models.

Method. SmartS is designed as a plug-and-play component that can be seamlessly integrated into the operating system. It takes in a sequence of pages' hotness across previous scheduling epochs as input and predicts their hotness in the next epoch. Hot pages are migrated to fast memory, while cold pages are evicted from the fast memory to the slow memory using the LRU algorithm. Figure 1 illustrates its neural architecture. It primarily encompasses three key components.

Component 1. Similar page identifier. The explosion of memory addresses is a major challenge in memory address prediction tasks [4, 5]. We must use a single model to predict hundreds of thousands of pages when adopting collective-page hotness prediction. The model necessitates numerous parameters to achieve high accuracy, consequently leading to daunting training and inference costs. Fortunately, we have observed that many pages have a similar access pattern. These pages exhibit similar migration behavior, meaning they are either migrated simultaneously or not at all. Based on this insight, we propose a similar page identifier to identify pages with similar access patterns and group them as a cluster called Clustered-Page (CluP).

We measure the similarity between two pages by calculating the Manhattan distance between their hotness sequences. Each page's access pattern can be represented as a time series $\{h_1, h_2, \ldots, h_n\}$, where h_n denotes the page's hotness at the n-th epoch. Similarities of two pages are described as $S_{ij} = \sum_{n=1}^{N} |h_{in} - h_{jn}|$, where S_{ij} represents the similarity between Page$_i$ and Page$_j$, h_{in} represents the hotness of Page$_i$ in the n-th epoch, h_{jn} represents the hotness of Page$_j$ in the n-th epoch.

We employ K-means, a widely used clustering algorithm, for clustering pages with similar access patterns. We group all pages into c CluPs based on their Manhattan distance. The appropriate value for c will be discussed in Appendix D. We use the average hotness of the pages within a CluP as the hotness of the CluP. Once the CluPs are formed, all sub-

* Corresponding author (email: yuchen@tsinghua.edu.cn)

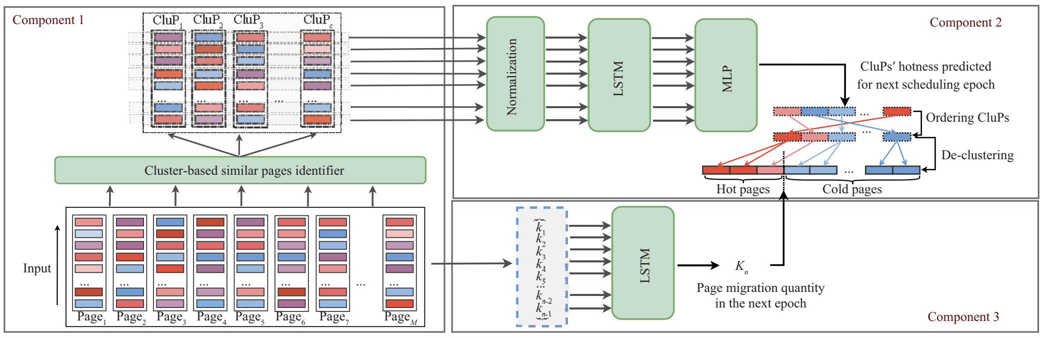
Component 1
CluP1 CluP2 CluP3 CluPc
Cluster-based similar pages identifier
Input
Page1 Page2 Page3 Page4 Page5 Page6 Page7 PageM
Component 2
Normalization
LSTM
MLP
CluPs' hotness predicted for next scheduling epoch
Ordering CluPs
De-clustering
Hot pages
Cold pages
k1 k2 k3 k4 k5 kn-2 kn-1
LSTM
Kn
Page migration quantity in the next epoch
Component 3

Figure 1 (Color online) Neural architecture of SmartS.

sequent training, inference, and migration are performed at the CluP granularity.

Component 2. Collective-page predictor. We improve the previous individual-page prediction to collective-page prediction to reduce the cost of neural models. Appendix C analyzes why it should be improved in this way.

Following the similar page identifier, we aim to predict the hotness of all CluPs and select pages within hot CluPs for migration. Specifically, given the obtained CluPs $= \{\mathrm{CluP}_1, \mathrm{CluP}_2, \ldots, \mathrm{CluP}_c\}$ and the hotness sequence HS $= \{H_1, H_2, \ldots, H_n\}$ of all the CluPs at time 1 to n, where $H_i \in$ HS and $H_i = \{h_{\mathrm{CluP}_1}, h_{\mathrm{CluP}_2}, \ldots, h_{\mathrm{CluP}_c}\}$, the objective is to predict the hotness of all CluPs.

Predicting CluP hotness is a typical time series prediction problem. We select long short-term memory (LSTM), a model extensively utilized for time series prediction, to predict CluP hotness. We employ LSTM to acquire sequential information from the hotness sequence HS. First, we fed the hotness sequence HS into the LSTM, $s_t = \mathrm{LSTM}(H_t,\ s_{t-1})$, where $H_t \in$ HS, s_t and s_{t-1} represent the hidden states for the hotness sequence at time t and time $t-1$, respectively, $1 \leqslant t \leqslant n$. We take the hidden state s_n as the output and then apply a two-layer fully connected feed-forward network to obtain the correlation among CluPs and get the final output: $y = \mathrm{ReLU}(s_n \times W_1 + b_1) \times W_2 + b_2 \ \in \mathbb{R}^c$, where $W_1,\ W_2 \in \mathbb{R}^{c\times c}$ and $b_1,\ b_2 \in \mathbb{R}^c$ are trainable parameters with activation function ReLU. In particular, we apply Layernorm after the first fully connected layer to accelerate the convergence rate. Thus far, we have obtained the hotness of all CluPs in the next epoch and sorted them according to their hotness.

Component 3. Page migration quantity estimator. Avoiding unnecessary page migrations is essential when using a hybrid memory scheduler. We must consider whether the benefits of page migration outweigh the overhead incurred by the migration. Current studies primarily rely on the pages' hotness to determine their worthiness for migration. A page is deemed worthy of migration when its hotness surpasses a certain threshold. However, we observe a deviation between predicted and true hotness. Fortunately, the ranking of predicted hotness is mostly consistent with the ranking of true hotness. This is due to the neural networks' superior capability in capturing relative relationships and trends within the data. So, we migrate pages based on their hotness ranking, selecting the top-k pages rather than considering their absolute hotness value.

SmartS introduces a neural-model-based approach to predicting the k-value. It calculates the k-value of historical epochs based on the given cost and benefit of page migration and predicts the k-value for the forthcoming epoch based on the historical k-value. Specifically, SmartS employs an LSTM model to capture the changing pattern of k-values. It takes a sequence of k-values KS $= \{k_1, k_2, \ldots, k_n\}$ as input and predicts the value k_{n+1}, where k_i represents the page migration quantity in the i-th epoch.

Experiments and results. Extensive experiments show that SmartS significantly outperforms existing schedulers in both effectiveness and cost. Detailed experimental settings and results are presented in Appendix D.

Conclusion. SmartS is a novel solution for hybrid memory scheduling using neural models. It proposes a novel collective-page prediction approach, effectively reducing training and inference costs. It also proposes a clustering-based approach to address the class explosion problem. Experiments show that SmartS improves hybrid memory effectiveness significantly. It also reduces the cost of neural models to allow their practical deployment in real-world hybrid, representing a substantial step towards practical neural-model-based scheduling.

Acknowledgements This work was supported by Pre-research Project (Grant No. 31511100101) and National Excellent Youth Scientists Fund of China (Grant No. 62122095).

Supporting information Appendixes A–E. The supporting information is available online at info.scichina.com and link.springer.com. The supporting materials are published as submitted, without typesetting or editing. The responsibility for scientific accuracy and content remains entirely with the authors.

References

1 Yang D, Liu H K, Jin H, et al. HMvisor: dynamic hybrid memory management for virtual machines. Sci China Inf Sci, 2021, 64: 192104

2 Meswani M R, Blagodurov S, Roberts D, et al. Heterogeneous memory architectures: a HW/SW approach for mixing die-stacked and off-package memories. In: Proceedings of the 21st International Symposium on High Performance Computer Architecture (HPCA), 2015. 126–136

3 Doudali T D, Blagodurov S, Vishnu A, et al. Kleio: a hybrid memory page scheduler with machine intelligence. In: Proceedings of the 28th International Symposium on High-Performance Parallel and Distributed Computing, 2019. 37–48

4 Shi Z, Jain A, Swersky K, et al. A hierarchical neural model of data prefetching. In: Proceedings of the 26th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, 2021. 861–873

5 Hashemi M, Swersky K, Smith J, et al. Learning memory access patterns. In: Proceedings of International Conference on Machine Learning, 2018. 1919–1928