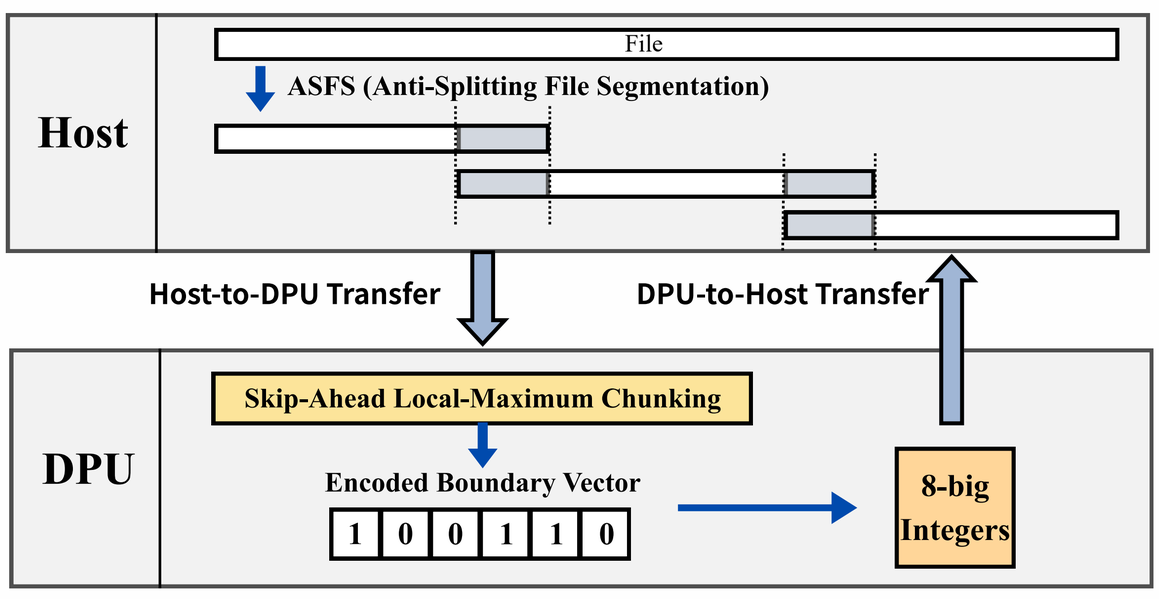
Compute-Efficient Chunking for Deduplication on Processing-in-Memory Systems

組員：陳星宇　指導老師：何建忠

Abstract

資料去重能提升儲存效率，但傳統分塊方法需掃描整個資料集，導致大量資料搬移與效能下降，此問題源自馮紐曼架構的計算與儲存分離，為解決此瓶頸，我們採用 UPMEM 的記憶體內運算（PIM）技術，於記憶體中直接執行計算以減少搬移。但是DPU 去重系統面臨跨 DPU 資料共享受限、缺乏原生的乘法支援，以及 DPU-CPU 通訊開銷等挑戰。因此我們提出一種架構透過 (1) 新增重疊區、 (2)平行化且對 DPU 友善的分塊演算法以及 (3) 降低傳輸量來優化整體效能。實驗結果顯示此架構在維持 100% 去重準確度下，分塊效能較 CPU 系統有顯著提升。

System Architecture

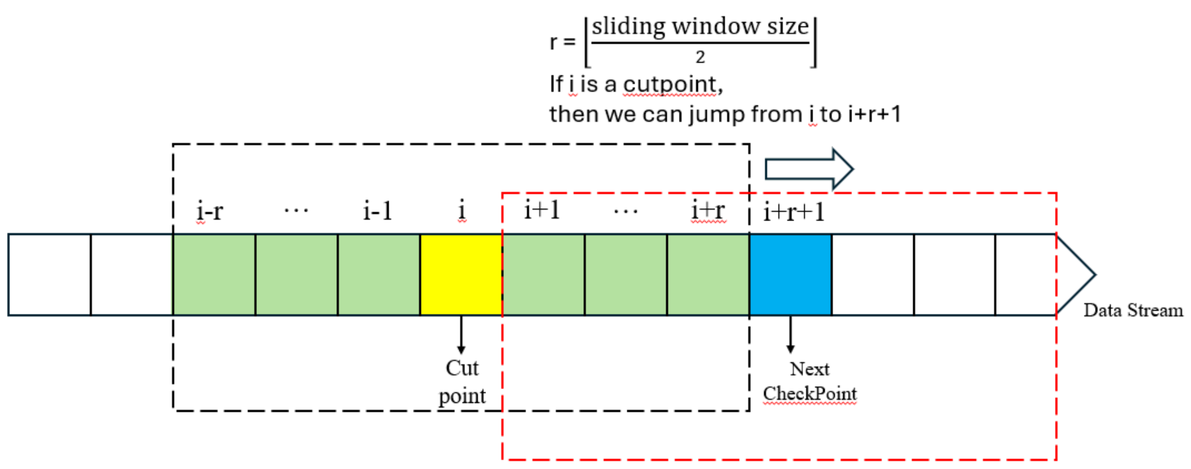


Method

我們的分塊系統採用三種機制如下：

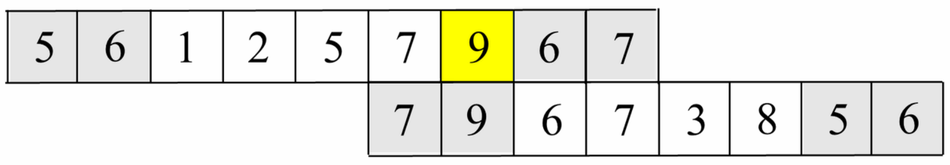
Skip-Ahead Local-Maximum Chunking

採位元比較來尋找區間最大值作為分塊邊界，並結合跳越機制來減少冗餘的窗口檢查



Anti-Splitting File Segmentation

在檔案段落間引入重疊區域，解決 DPU 之間因缺乏直接通訊而無法識別跨邊界標記的問題，確保在多處理器並行環境下仍能維持 100% 的分塊一致性

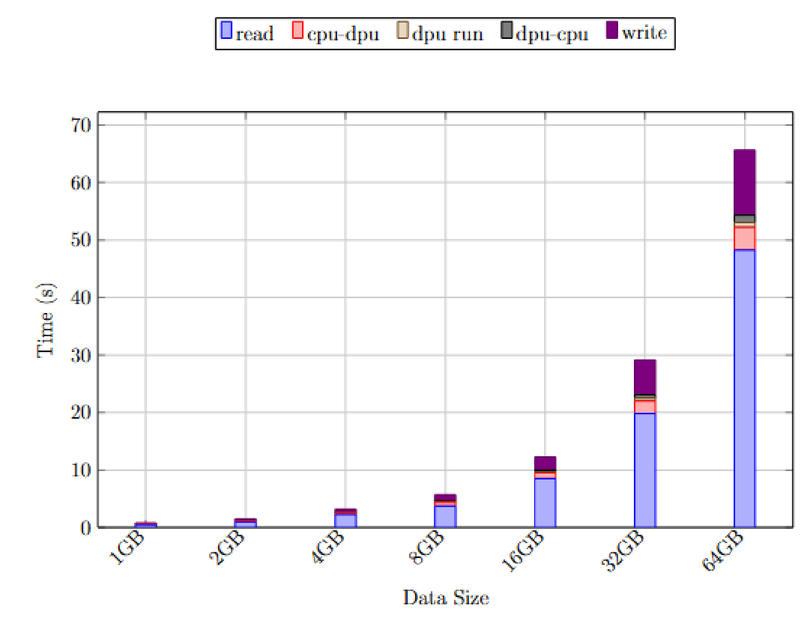


Encoded Boundary Vector

將 DPU 識別出的標記位置記為二進位陣列，並將每 8 位元壓縮成 1 個位元組的整數，從而將回傳至主機的資料量減少至原始大小的八分之一

Experiment

實驗一：固定DPU數量，觀察不同資料集對效能的影響



實驗二：在多個真實資料集下的用時與SpeedUp

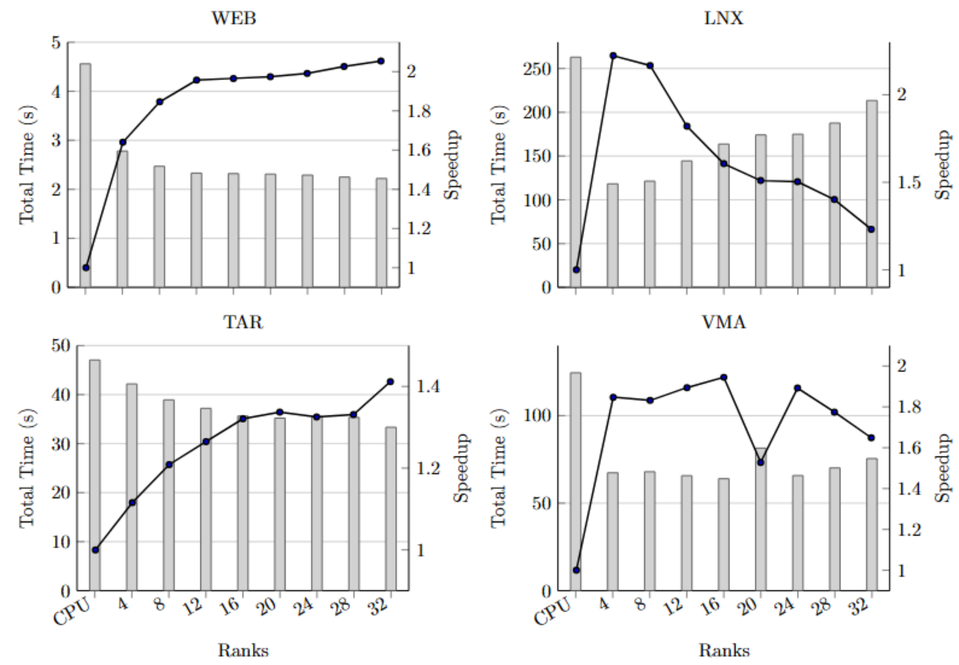


Figure 1: Total Time bars and speedup curves using CPU as the baseline

實驗三：在多個真實資料集下的去重率

Table 1: Deduplication Ratio under Different Window Sizes (byte)

Dataset	129	257	385	513	641	769	897	1025
WEB	94.33%	90.02%	88.26%	85.00%	77.15%	75.45%	73.77%	72.32%
LNX	91.34%	80.24%	75.43%	64.84%	60.56%	56.70%	52.95%	49.44%
TAR	14.34%	13.97%	13.69%	13.43%	13.14%	12.79%	12.35%	11.86%
VMA	69.67%	69.52%	69.36%	69.19%	68.98%	68.72%	68.40%	68.01%

Conclusion

研究提出專為 UPMEM DPU 優化的 ASFS、SALM 及資料壓縮創新技術成功克服了 PIM 架構在跨處理器數據共享、硬體運算限制與通訊頻寬上的瓶頸，在確保 100% 分塊結果一致性的前提下，相較於傳統基於 CPU 的系統有顯著提升。

References

[1] C.-L. Yeh, L.-C. Chen, C.-C. Ho, Y.-M. Chang, and D.-W. Chang, “PIMDup: An optimized deduplication design on a real processing-in-memory system,” in Proceedings of the 62nd ACM/IEEE Design Automation Conference (DAC), Jun. 2025.

[2] W. Xia, Y. Zhou, H. Jiang, D. Feng, Y. Hua, Y. Hu, Y. Zhang, and Q. Liu, “FastCDC: A fast and efficient content-defined chunking approach for data deduplication,” in Proceedings of the 2016 USENIX Annual Technical Conference (USENIX ATC ’16), Jun. 2016.