Survey Inter-Reference Recency Based Page Replacement Policies to Cope with Weak Locality Workloads

Arjun Singh Saud
Central Department of Computer Science and IT, Tribhuvan University, Kritipur, Nepal
Corresponding Email: arjunsaud@cdcsit.edu.np

Abstract— **Least recently used (LRU) makes bold assumption on recency factor only which made LRU miss behave with weak locality workloads. If the "frequency", of each page reference is taken into consideration, it will perform better in the case where workload has weak locality. Frequency count leads to serious problem after a long duration of reference stream because it cannot cope with change in locality. Reuse distance or inter reference recency (IRR) of a block is equal to number of distinct pages accessed between recent consecutive or correlated access of that particular block. Many recent variations of LRU use IRR rather than recency such that LRU can be made friendly with weak locality workloads. This papers surveys LRU variants that use IRR to make page replacement decision.**

I. INTRODUCTION

LEAST recently used (LRU) is the page replacement policy that performs well in case of strong locality workloads. In addition, it adapts faster during change in working set with workloads having good locality of reference. But LRU makes bold assumption on recency factor only which made LRU miss behave with weak locality workloads. Due to this IRR (Inter-reference Recency) based page replacement algorithms are becoming popular. Here are some representative examples reported in the research literature to illustrate how poorly LRU behaves. [1][2]

a. **Frequency Based Pattern:** Suppose that a memory consists of 101 page frames. Consider a program in which there is a regularly repeated access to 100 pages and once in every 50 references there is a reference to a new page (e.g. access to a file block). In such case LRU replaces the frequently used page frame.
b. **Looping Pattern:** Let us suppose that memory consists of 5 page frames and considers a program which repeatedly references 6 pages in cyclic order.
c. **Sequential Scans:** Infrequently used blocks, such as sequential scans through large files, may cause the replacement of frequently referenced blocks in cache.

II. INTER-REFERENCE GAP

If the "frequency", of each page reference is taken into consideration, it will perform better in the case where workload has weak locality. One algorithm using "frequency" based prediction is LFU (Least Frequently Used). LFU replaces the page having least frequency of reference. Thus, LFU uses more history information for the prediction of future reference, whereas LRU uses only most recent history of pages. Frequency count leads to serious problem after a long duration of reference stream. Because when the locality changes, reaction to such certain change will be extremely slow. Many researches are performed to overcome the anomalous behavior of LRU with weak locality workloads [2][3][4].

The Inter-Reference Gap (IRG) of a block is the number of the references between consecutive references to the block. Phalke and Gopinath considered the correlation between history IRGs and future IRGs. [1] The past string of IRGs of a block is modeled by Markov chain to predict its next IRG. However, replacement algorithms based on a Markov model fail in practice because they try to solve a much harder problem than the replacement problem itself. [4][5]

III. LIRS PAGE REPLACEMENT POLICY

There are different modified versions of LRU algorithm. Among them LIRS page replacement algorithm identifies and eradicates the misbehaviors of LRU on weak locality of references. LRU only uses recency factor whereas LIRS uses additional factor called reuse distance for page replacement. Reuse distance or IRR of a block is equal to number of distinct pages accessed between recent consecutive or correlated access of that particular block. Strong part of LIRS algorithm is the IRR value which maintains recency as well as frequency factor. LIRS algorithm [1] uses two sets of pages based on IRR. Set of pages with low IRR value is taken as hot block and called low inter-reference recency set (LIRS). Set of pages with high IRR value is taken as cold block and called high inter-reference recency set (HIRS). Blocks that can be most probably used in future are taken as hot blocks whereas blocks that may not be used in near future are taken as cold blocks. Hence HIR blocks are always replaced and LIR blocks are never replaced. LIR page is always available in cache whereas HIR page may or may not be available in cache. HIR page that is available in cache is called resident HIR and HIR page that is not available in cache is called non-resident HIR. Hence a page which is accessed first time is taken as non-resident HIR. Fixed number of LIR block and resident HIR block is used which is equal to 99 % and 1 % of cache size respectively. Partition of cache doesn't obstruct the overall performance.

IV. WORKING OF LIRS

Stack S contains page reference accessed. Its main purpose is to maintain recency value. As we move toward bottom recency factor increases. Bottommost one is always LIR block, which is the oldest block having higher recency factor and topmost one is the recent block having recency factor equals to zero. Each stack node contains information about reference block. Here information of every page reference is not available in stack S due to the major event stack pruning.

Some information is also available in queue Q and some outdated information is left.

The major function stack pruning is conducted during status change. Bold assumption of the algorithm is that LIR block always contains in the bottom. Bottom of stack S is always LIR block. While changing status, the page in bottom of stack S is demoted to HIR resident for that it is kept in queue Q. At that time next LIR bottom is chosen which is nearer from bottom of stack S and all other HIR bottom are removed one by one. Information of thus removed HIRs is available in queue Q, if it is resident. Stack pruning is also conducted if the accessed block X is the bottom LIR because recent block is always moved to top of stack S. Stack pruning decreases the size of stack hence the stack doesn't keep track of outdated references. Also outdated HIR can't be promoted if its history information is unavailable even in Q.

Queue Q contains collection referenced page that are available in cache. But it only tracks resident HIR blocks. Hence size of HIR cache partition determines the size of Queue Q. The block in the Queue can be removed from anywhere if it is promoted to LIR. In that case the bottom most one LIR block of stack is inserted to end of Q then it becomes resident HIR as it is now in Queue. Block in the front of Queue is removed, now the removed block demotes to non-resident HIR. Comparing IRR and recency value is automatically done by the use of Q which increases performance.

CLOCK-Pro takes the same principle as that of LIRS (it uses the reuse distance (called IRR) rather than recency in its replacement decision) based on CLOCK infrastructure.[3] Generally in various replacements algorithms even in LIRS the movement of pages needed even the page hit occur but in CLOCK-Pro in this situation movement of pages never take place. Here pages categorized into two groups: cold pages and hot pages based on their reuse distances (or IRR). When a page is accessed, the reuse distance is the period of time in

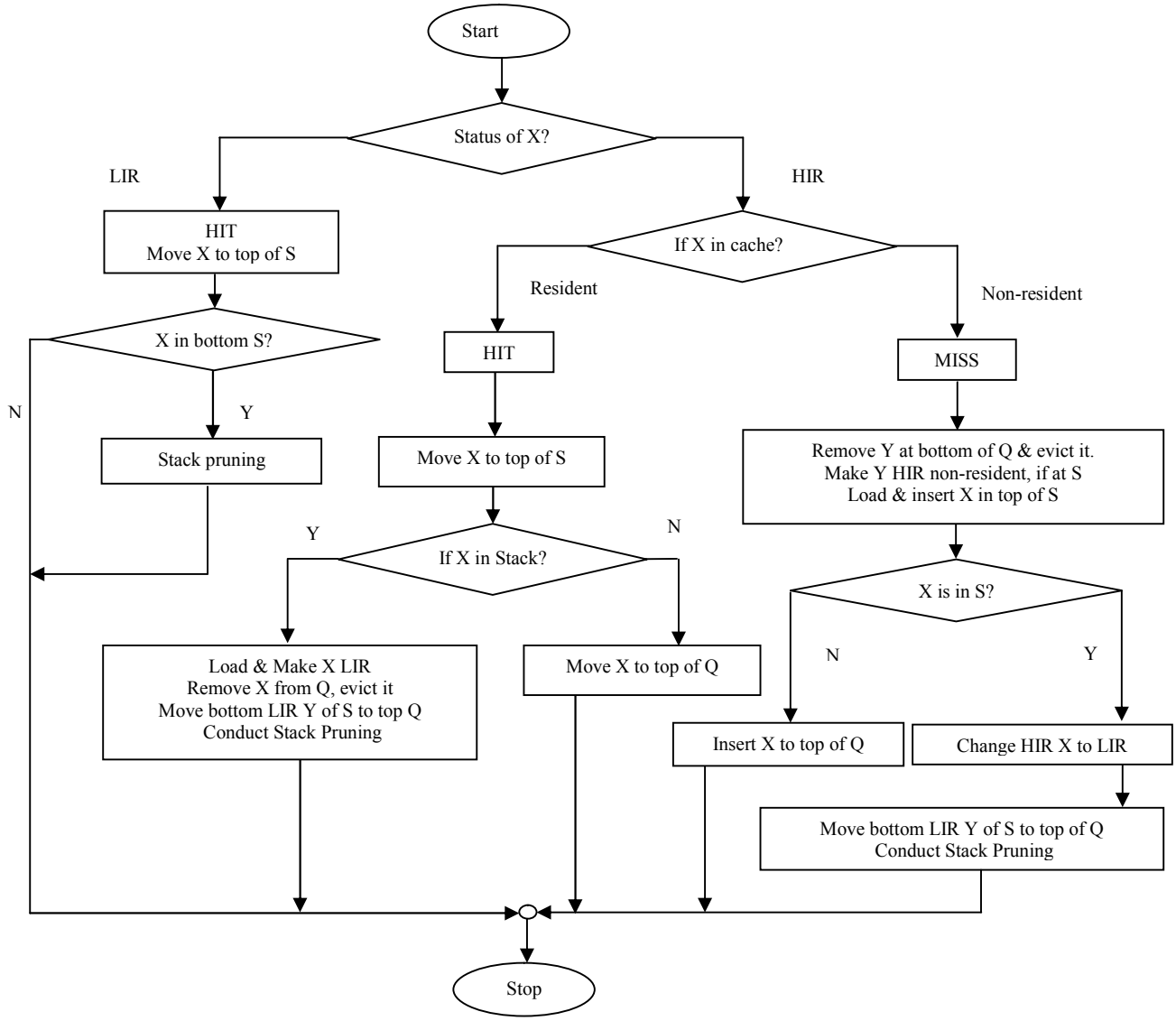

Fig.1. Flowchart of LIRS Algorithm

terms of the number of other distinct pages accessed since its last access. Although there is a reuse distance between any two consecutive references to a page, only the most current distance is relevant in the replacement decision. This algorithm uses the reuse distance of a page at the time of its access to categorize it either as a cold page if it has a large reuse distance, or as a hot page if it has a small reuse distance. Then mark its status as being cold or hot.

V. WORKING OF CLOCK-PRO PAGE REPLACEMENT

The algorithm puts all the accessed pages, either hot or cold, into one single list in the order of their accesses. In the list, the pages with small recencies are at the list head, and the pages with large recencies are at the list tail. To give the cold pages a chance to compete with the hot pages and to ensure their cold/hot statuses accurately reflect their current access behavior, CLOCK-Pro grant a cold page a test period once it is accepted into the list. Then, if it is re-accessed during its test period, the cold page turns into a hot page. If the cold page passes the test period without a re-access, it will leave the list. Note that the cold page in its test period can be replaced out of memory; however, its page metadata remains in the list for the test purpose until the end of the test period or being re-accessed. When it is necessary to generate a free space, this algorithm replaces a resident cold page. The key question here is how to set the time of the test period. When a cold page is in the list and there is still at least one hot page after it (i.e., with

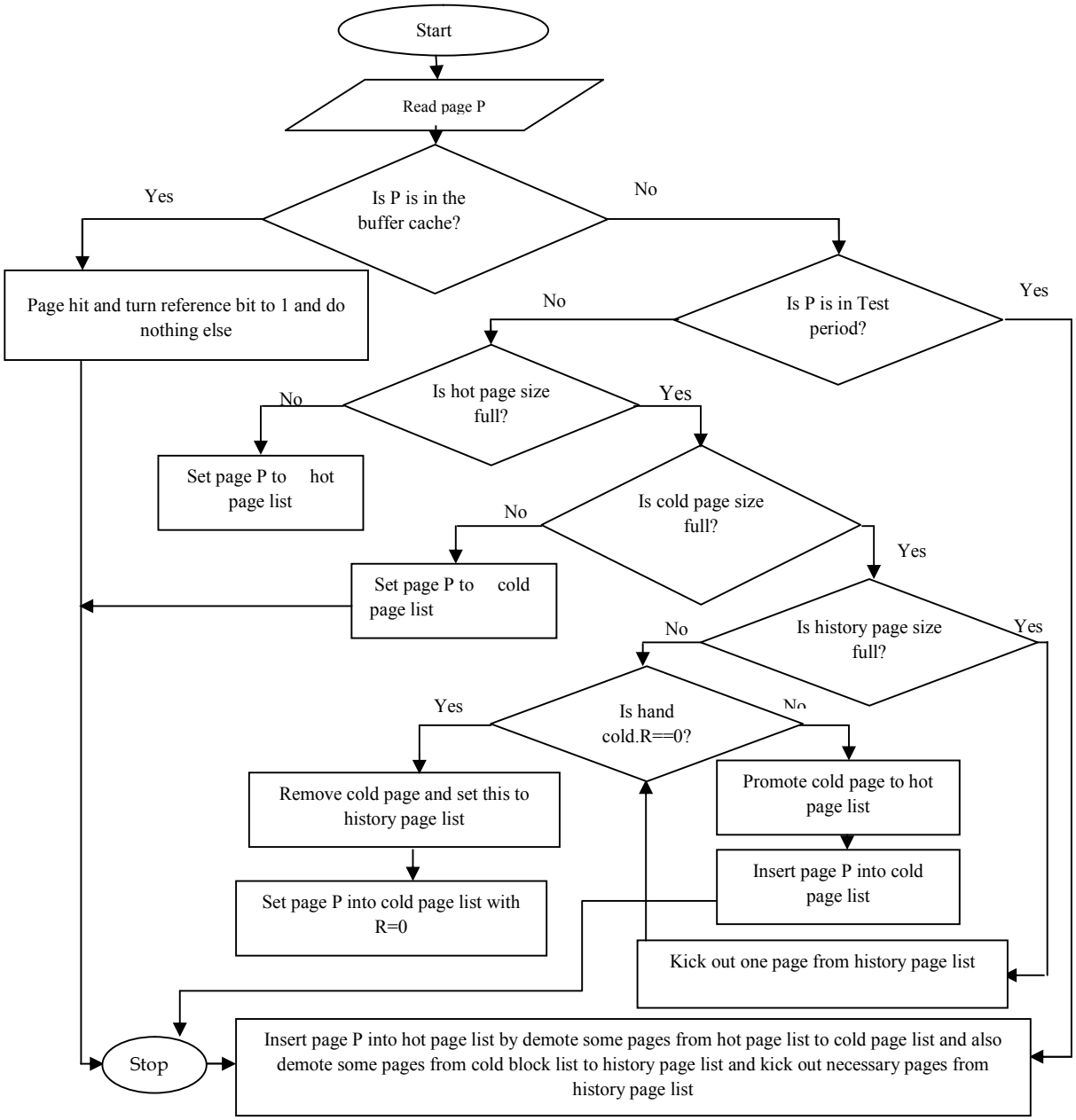

Fig. 2. Flowchart of CLOCK-Pro Algorithm

a larger recency), it should turn into a hot page if it is accessed, because it has a new reuse distance smaller than the hot page(s) after it. Accordingly, the hot page with the largest recency should turn into a cold page. So the test period should be set as the largest recency of the hot pages. If we make sure that the hot page with the largest recency is always at the list tail, and all the cold pages that pass this hot page terminate their test periods, then the test period of a cold page is equal to the time before it passes the tail of the list. Thus, all the non-resident cold pages can be removed from the list right after they reach the tail of the list.

There are three hands: Hand-hot for hot pages, Hand-cold for cold pages and Hand-test for running a reuse distance test for a block. The allocation of memory pages between hot pages (Mhot) and cold pages (Mcold) are adaptively adjusted. (M=Mhot+Mcold). Here all hot pages are resident; some cold pages are also resident and also keep track of recently replaced pages.

VI. CONCLUSION

Reuse distance or inter reference recency (IRR) of a block is equal to number of distinct pages accessed between recent consecutive or correlated access of that particular block. Many recent variations of LRU use IRR rather than recency such that LRU can be made friendly with weak locality workloads. Beauty of IRR based replacement policies is that they do not need deep past history information. Even though IRR successfully cope with weak locality workloads and strong locality workloads, most of IRR based page replacement algorithms have high implementation overhead. Thus IRR based page replacement algorithms are hot area of research in memory management.

VII. REFERENCE

[1] Song Jiang and Xiaodong Zhang (2005), Making LRU Friendly to Weak Locality Workloads: A Novel Replacement Algorithm to Improve Buffer Cache Performance, IEEE Transactions on Computers, Vol. 54, and No. 8.

[2] S. S, Arjun and Bhatt J.(2012), Recency and Prior Probability (RPP) based Page Replacement Policy to cope with Weak Locality Workloads having Probabilistic Pattern, IJCA.

[3] Jiang, S., Chen, F., Zhang, X, CLOCK-Pro (2005): An effective improvement of the CLOCK replacement. In Proceedings of the 10th Annual USENIX Technical.

[4] V. Phalke and B. Gopinath (1995), "An Inter-Reference Gap Model for Temporal Locality in Program Behavior," Proc. ACM SIGMETRICS Conf. Measuring and Modeling of Computer Systems, 1995.

[5] D. Lee, J. Kim, S. Noh, S. Min, Y. Cho, and C. Kim, (1999), "On the Existence of a Spectrum of Policies that Subsumes the Least Recently Used (LRU) and Least Frequently Used (LFU) Policies", *Proc. ACM SIGMETRICS Conf. Measuring and Modeling of Computer Systems*.

VIII. BIOGRAPHY

Arjun Sing Saud is faculty member of Central Department of Computer Science and Information Technology, TU, Kirtipur. He has completed his M.Sc. CSIT degree from the same department in distinction division. He has devoted more than 10 years in academic field and more than 9 years as university teacher.