OPT 置換判断に基づく LSTM 学習モデルによるキャッシュ置換

三浦 泰誠† 光来 健一‡ 山口 実靖†

† 工学院大学 工学研究科 〒165-8677 東京都新宿区西新宿 1-24-2
‡ 九州工業大学大学院 情報工学研究院 〒820-8502 福岡県飯塚市川津 680-4
E-mail: † cm20048@ns.kogakuin..ac.jp, ‡ kourai@ksl.ci.kyutech.ac.jp, † sane@cc.kogakuin.ac.jp

あらまし 近年のクラウドサービスの拡充に伴い，仮想化環境は重要なプラットフォームとなっている．階層的キャッシュ構造を持つ仮想化環境における下位キャッシュでは，キャッシュ置換アルゴリズムとして一般的な LRU(Least Recently Used)が効果的に機能しないという問題が存在している．また，最適なキャッシュ置換手法として OPT が提案されているが，この置換手法は将来のアクセスの情報を必要とするため，通常は実装が不可能となっている．そこで深層学習 LSTM(Long short-term memory)を用いて，過去のストレージアクセス履歴から将来のアクセスを予測する，OPT の置換判断を模倣したキャッシュ置換手法を提案し，そのシミュレーションによりその有効性を評価する．

キーワード Operating System, 仮想化, キャッシュ， キャッシュ置換， LRU, 深層学習, LSTM

1. はじめに

ホスト型仮想化環境のようにホスト OS に加えゲスト OS が稼働する仮想化環境では，ホスト OS，ゲスト OS の両 OS で, ストレージアクセス用のキャッシュ(ページキャッシュ)が動作し, ストレージアクセスに対して多重のキャッシュを介する階層型の構造を持つ．よって，両 OS キャッシュのヒット率がストレージアクセスにおける I/O 性能にとって重要となる．

キャッシュは格納されたデータをブロック単位で管理し，キャッシュ内にあるキャッシュブロックを一定の法則に則って置換しながら動作する．そして，置換対象選択の改善によりキャッシュヒット率の向上を図る．キャッシュ置換アルゴリズムとして，参照の時間的局所性に強い適応性のある LRU(Least Recently Used)が一般的に用いられてきたが，ホスト OS 型仮想化環境のような階層型のキャッシュ構造を持つ環境における下位キャッシュにおいて LRU は効果的に機能しないことが知られている[1].

また，理論的に最高のキャッシュヒット率を実現できるキャッシュ置換手法として OPT(Optimal)が提案されている[2]. この手法はキャッシュ構造に関わらず最高のキャッシュヒット率を達成できるが，この置換手法は将来のアクセス情報を必要とするため，通常の計算機や OS に置いては実装は不可能である．

本稿では, 過去のアクセスの情報から OPT の置換を再現し，そのアクセス履歴を深層学習 LSTM(Long short-term memory)により学習し OPT を模擬する学習モデルを作成する．そして，学習モデルを用いて OPT の置換対象を予測することで，OPT による置換判断を疑似的に実現したキャッシュ置換手法を提案する．そして，その性能をキャッシュヒットシミュレーションにより評価する．

本稿の構成は以下の通りである．2 章では関連研究を紹介する．3 章では将来のアクセス情報を予測する LSTM 予測器と，予測器を用いて OPT を疑似的に実現したキャッシュ置換手法の提案を行う．4 章では，シミュレーションにより提案手法と LRU や既存の深層学習によるキャッシュ置換手法との比較評価を行う．そして，5 章にて本稿をまとめる．

2. 関連研究

2.1. LRU

LRU[3]は，キャッシュから破棄するブロックを選択する際に，最後に参照されてからの時間が最も長いブロックを選択するキャッシュ管理手法であり，多くのシステムソフトウェアにてキャッシュ置換手法として利用されている．多くのアプリケーションから発行されるデータアクセス要求には参照の時間的局所性[4]が存在し，その様な局所性を有するアクセス要求に対して LRU は効果的に機能する．しかし，仮想化環境におけるホスト OS キャッシュである下位キャッシュ[5]や CPU キャッシュ[6]では，この置換アルゴリズムが効果的に機能しないことが報告されている．

2.2. MRU

MRU(Most Recently Used)は，キャッシュ内のデータブロックのうち，最新のアクセス要求からの時間が最も短いブロックを破棄するキャッシュ置換手法である[7][8]．MRU では，キャッシュに実質，固定的にデータブロックを保持し続ける．

2.3. OPT

理論上最適な(最もヒット率が高い)キャッシュ置換アルゴリズムとして OPT(Optimal)が提案されている[2]．OPT では，キャッシュ内のブロックを置換する必要が発生した時(キャッシュミスが発生した時)，キャッシュ内の各ブロックの次にアクセスが行われるま

での時間を調べる．そして，最も長い期間アクセスされないブロックをキャッシュから破棄する．この様に，キャッシュ置換を行うためにはキャッシュ内の各ブロックの次にアクセスされる時刻(つまり未来のアクセス情報)を保持している必要があるため，現実の OS に対して OPT アルゴリズムを実装することは現実的ではない．OPT は，アクセス情報が与えられている状況におけるキャッシュ置換手法の性能評価などにおける比較対象などとして用いられる．

2.4. 参照の時間的局所性

通常のアプリケーションからメモリやストレージへのアクセスには参照の時間的局所性[4]が存在することが確認されており，このような状況では LRU が効果的に機能することが期待できる．一方で，上位キャッシュが LRU を用いる二重キャッシュ環境の下位キャッシュにおいてはアクセスに負の参照の局所性が存在し[1][9][10]，LRU が効果的に機能しないことが確認されている．

2.5. 深層学習を用いたキャッシュ置換

深層学習をキャッシュ置換に活用する取り組みも行われている．Shi らは LSTM をキャッシュ置換に活用する取り組みを行っている[11]が，仮想化環境などの二重キャッシュ環境への適用については十分に考察されていない．

我々は過去に，LSTM を用いたキャッシュ置換手法を提案している[12]．この手法は，現時刻を基準として，次時刻において各ブロックがアクセスされる確率を深層学習(LSTM)により予測する予測器を元に実装される．過去のストレージアクセスパターンの特徴と，未来のアクセスパターンの特徴に類似性があるとの前提のもと，予測器の予測モデルは過去のストレージアクセス履歴を LSTM に入力し学習させて構築する．

この予測器では，現時刻においてアクセスされたブロックのブロック番号に該当するベクトルを 1，それ以外の全ブロックの番号に該当するベクトルを 0 とした One-Hot ベクトルを入力値とし，各ブロックの次時刻のアクセス確率を表現するベクトルを出力値として持つ．この提案手法では，予測できる範囲が次時刻のアクセスのみであることや，仮想化環境下のホスト OS において，アクセスデータを表現する One-Hot ベクトルでは LSTM による予測を行えるための特徴量を十分に得ることが難しく精度が低くなるという問題がある．また，最も優れた破棄対象は前節の OPT 手法により選択されたブロックであるが，本手法では次時刻においてアクセスされる確率が最も低いブロックを破棄対象としている点にも改善の余地がある．

本稿では，この手法を既存手法として，提案手法および LRU との比較調査を行う．

2.6. 階層構造における排他的キャッシュ方式

多くの階層的な構造を持つキャッシュにおいて，上位キャッシュ，下位キャッシュの双方は LRU のようなキャッシュ管理ポリシーを使用している．そのため，上位キャッシュと下位キャッシュでデータブロックの重複が発生し，両キャッシュのうちキャッシュサイズが大きい方のキャッシュのみが存在しているように動作し，メモリ資源を大きく浪費しています．

そこで，Wong 氏らは[13]，アクセスが行われたデータブロックは上位キャッシュに複製し，下位キャッシュには上位キャッシュから破棄されたデータブロックを複製することで，上位キャッシュと下位キャッシュの両方で単一の大きなキャッシュとして動作する，排他的キャッシュ方式を提案し，シミュレーション上で通常のキャッシュ置換ポリシーよりも優れた性能を保持することを示した．

2.7. 2Q, MQ

アプリケーション実行時に得られるアクセスパターンには，最近アクセスされたデータブロックは近い将来再度アクセスされるという性質を持つものが多い．この時間的な局所性に対して LRU は強い適応性があるが，それをさらに時間的局所性に対応するために 2Q(Two Queue)アルゴリズムが提案されている[14]．このアルゴリズムでは，キャッシュ内に LRU が機能するバッファと，アクセスされた順番が最も古いキャッシュから順に破棄していく，FIFO(First In First Out)が機能するバッファを持っている．LRU がアクセスパターンのうちアクセス頻度が高いデータブロックをキャッシュに保持するように働くアルゴリズムに対し，FIFO は同じデータブロックへのアクセスが短期間に何度も繰り返される場合に有効に働くアルゴリズムである．よって 2Q は，アクセスの最新性と頻度を考慮して機能するアルゴリズムであり，LRU と比べてアクセスの時間的局所性により適応できるアルゴリズムと言える．

また，サーバコンピュータのキャッシュの下位にあるネットワークストレージ機器のキャッシュなど，負の参照の局所性が存在する状況に適したキャッシュ置換手法として複数の LRU キューを組み合わせて使用する MQ[10]が提案されている．

2.8. ARC

アクセスパターンの変化に動的に対応できるキャッシュ置換ポリシーとして ARC(Adaptive Replacement Cache)が提案されている[15]．ARC では 1 つのキャッシュ内でデータブロックを固定的に保持する MRU と同様の動作をする範囲と，最も使われていないデータブロックから破棄する LRU と同様の動作をする範囲を持っており，それらの大きさを動的に調整することで最適化を図る．つまり，時間的に局所性の大きいア

クセスが行われる時には LRU 的な動作をする範囲が大きくなり，時間的局所性のないアクセスが行われている時には MRU 的な動作をする範囲が大きくなる．これにより ARC は様々なアクセスパターンに適応することができ，LRU よりも優れた置換性能を実現する．

3. LSTM による疑似 OPT キャッシュ置換

本章にて，深層学習 LSTM を用いた疑似 OPT に基づいたキャッシュ置換手法の提案を行う．提案手法では，過去のストレージアクセス履歴より OPT の置換対象(次にアクセスされるまでの時間が最長であるブロック)を予測する予測器を構築し，その予測結果を用いてキャッシュ置換を行う．

3.1 LSTM 予測器

本節では，LSTM を用いたストレージアクセス予測器について述べる．予測器として扱う LSTM の学習モデルは，以下のようにして構成される．

アプリケーション実行中にアクセス対象となる全ブロックに対して，次にアクセスされるまでの時間を予測する．予測器への入力ベクトルは，アクセス対象となる全ブロックと同数の次元を持ち，そのベクトルの各要素は最後にアクセスされた時刻から現在までの時間を表している．予測器への出力ベクトルも対象となる全ブロックと同数の次元を持ち，そのベクトルの各要素は現時刻を基準に各ブロックが次にアクセスされるまでの時間を表している．

予測器として扱う LSTM モデルにおいて，学習過程では損失関数として順列確率損失(PPL)を用いる．順列確率損失は，ランク学習にて用いられる損失関数であり，出力データの並び順の起こりやすさを確率分布にした順列確率分布と教師データの並び順とで求める交差誤差であり，(1)式によって表される．(1)式のうち，$\boldsymbol{\pi}$ は並び順，$\mathbf{s}$ はブロックごとの頻度，$\bar{s}$ は s の平均を表す．

$$PPL=-\sum P(\boldsymbol{\pi}|\bar{s})\log P(\boldsymbol{\pi}|s) \quad \cdots (1)$$

予測器として扱う LSTM モデルのその他のパラメータは表 1 に示す通りである．

表 1 LSTM モデル各種パラメータ

Parameters of the constructed LSTM model	
Parameter name	***value***
number of input dimensions	**500**
number of output dimensions	**500**
Number of middle layers	**1**
number of dimensions in the middle layer	**100**
activation function	**ReLu, Softmax**
optimization function	**Adam**
loss function	**Permutation Probability Loss(PPL)**

学習においては，過去のストレージアクセス履歴を教師データとして入力して予測モデルを作成する．学習では説明変数としてその時刻における各ブロックの最後にアクセスされてからの時間を表すベクトルを与え，目的変数としてその時刻における各ブロックの次にアクセスされるまでの時間を表すベクトルを与える．

3.2 置換手法

提案手法では，LSTM によるストレージアクセス予測器を用いて，キャッシュ置換を行う．

アクセス 1 回ごとに予測器から各ブロックの次にアクセスされるまでの時間の予測結果を得る．そして，キャッシュ内にあるブロックのうち次にアクセスされるまでの時刻の予測値が最も長いブロックをキャッシュから破棄する．

4. シミュレーション評価

提案したキャッシュ置換手法について，シミュレーションによるキャッシュヒット率の評価を行う．評価では，下位キャッシュにおけるキャッシュヒット率を LRU や既存手法(LSTM によるキャッシュ置換手法[12])と比較する．

LSTM 予測器の学習とシミュレーション評価では，Zipf 分布に基づいた乱数を生成し，それをアプリケーションから発行されるアクセスの対象とした．

当該乱数 x は $0 \leqq x < 500$ の範囲を持ち，再アクセス間隔は図 1 に示すような分布となっている．図から，再アクセス間隔が非常に短いアクセスが多い(あるブロックへのアクセスから短い時間の後に再度同じブロックにアクセスした事例が多い)アクセス分布であることが分かる．

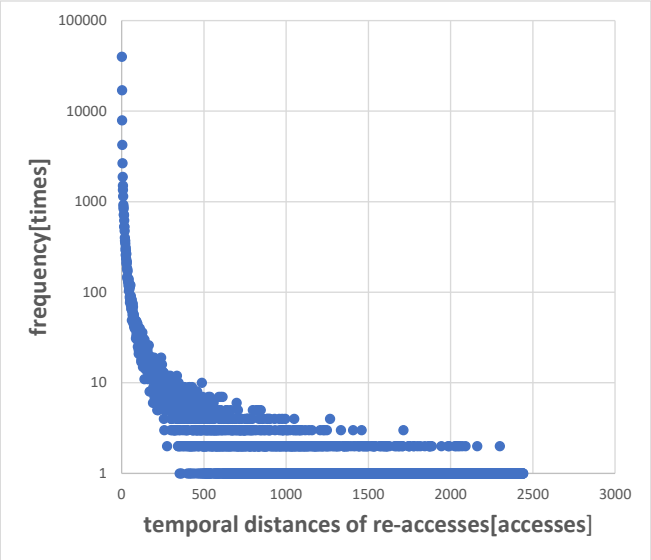

図 1 Zipf 分布に基づくアクセスデータ
における再アクセス間隔

4.1 非二重キャッシュ環境

Zipf 分布を基に生成したアクセスを用いて提案手法，LRU，既存の LSTM による置換手法のキャッシュヒット率をシミュレーションにより評価した．アクセスの対象となる全ブロックは 500 [block]であり，キャッシュサイズを 100 [blocks]から 500 [blocks]に変化させて評価を行った．

シミュレーション結果を図 2 に示す．図より，階層的なキャッシュ構造を持たない非二重キャッシュ環境においては全てのキャッシュサイズにおいて LRU による置換性能が他の置換手法よりも優れた性能を持つことがわかった．このことから，通常の環境(二重キャッシュでない環境など)においては伝統的に使用されてきた LRU が近年注目を集めている深層学習を用いる手法よりも優れた性能(キャッシュヒット率)を実現できることが分かる．今回は評価対象としていないが，LRU は深層学習と比べて CPU 資源消費が大幅に少ないことが期待でき，LRU は少ない資源消費で高い性能を達成したと言える．

LSTM を用いる 2 手法(提案手法と既存手法)を比較すると，既存の LSTM によるキャッシュ置換手法ではキャッシュサイズが小さくなるほど大きな性能の低下がみられるが，提案手法はキャッシュサイズが 100[blocks]であっても 80%ほどのキャッシュヒット率を保つことができ，既存手法と比較するとおよそ 4 倍の性能差となることを確認できた．

4.2 二重キャッシュ環境

次に，二重キャッシュ環境における下位キャッシュでのキャッシュヒット率をシミュレーションにより評価する．下位キャッシュのサイズを 100[blocks]とし，上位キャッシュを 10[blocks]から 250[blocks]に変化させてシミュレーション評価を行った．

シミュレーション結果を図 3 に示す．図 3 より，どの置換手法も上位キャッシュサイズの増加とともに下位キャッシュのヒット率が低下していることが確認できる．ただし，提案手法は上位キャッシュサイズが増加しても 20%ほどのキャッシュヒット率を保つことができ，強い負の参照の局所性があってもある程度のヒット率を達成できることが分かる．

一方で上記キャッシュサイズが小さい状況では，負の参照の局所性が弱く通常のストレージアクセスと類似の状況となり，4.1 節同様に LRU が最も優れる結果となった．

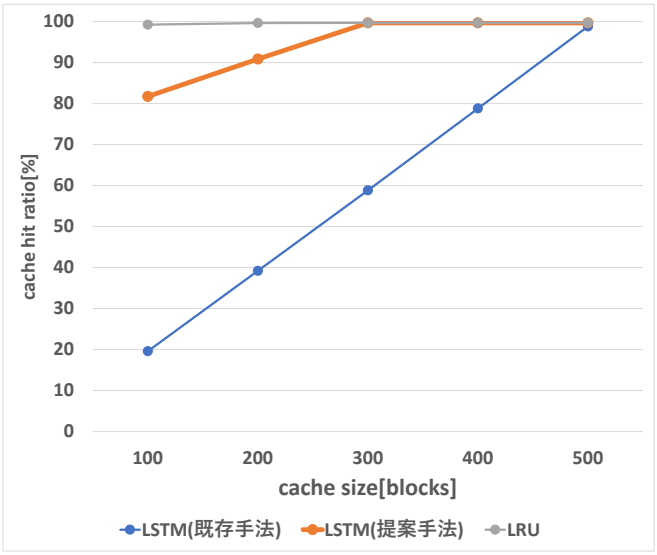

図 2　非二重キャッシュ環境における
キャッシュヒット率

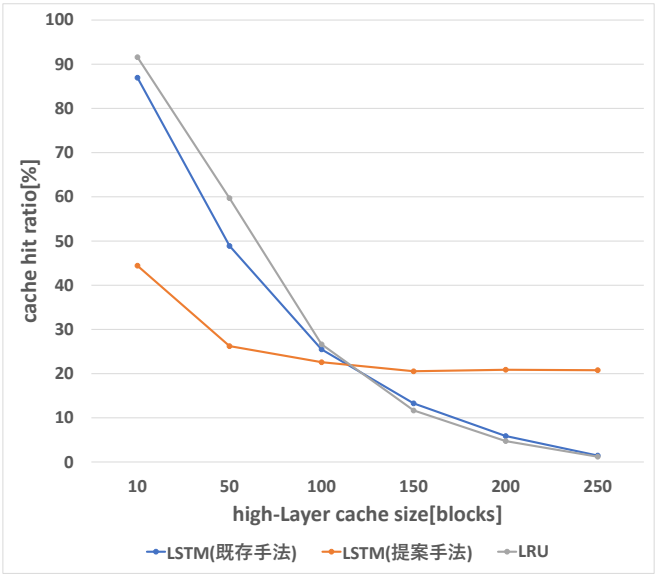

図 3　二重キャッシュ環境における
キャッシュヒット率

5. おわりに

本稿では，仮想化環境の下位キャッシュにおいて効果的に機能する新たなキャッシュ置換手法として，深層学習 LSTM を活用して最適置換手法 OPT を疑似的に実現した手法を提案し，キャッシュヒット率のシミュレーションによりその有効性を評価した．

評価の結果，二重キャッシュ環境においては，LRU および既存手法では上位キャッシュのキャッシュサイズが増加するとヒット率が大きく低下する(本稿の実験では 1.5%程度に低下)が，提案手法ではある程度(20%程度)のキャッシュヒット率を維持できることが確認できた．

また，非二重キャッシュ環境下においては，LRU よりはヒット率が低いことが確認されたが，既存の深層学習を用いたキャッシュ置換手法よりも大幅に高いヒット率を実現できることが確認できた．

今後は，IOTTA[16]などの一般公開されている実ストレージアクセス履歴を用いての評価を行う予定である．

参 考 文 献

[1] Y. Nagasako and S. Yamaguchi, "A Server Cache Size Aware Cache Replacement Algorithm for Block Level Network Storage," *2011 Tenth International Symposium on Autonomous Decentralized Systems*, Tokyo & Hiroshima, 2011, pp. 573-576. doi: 10.1109/ISADS.2011.80

[2] L. Belady. A study of Replacement Algorithms for a Virtual-storage Computer. IBM Systems Journal, 1966.

[3] Elizabeth J. O'Neil, Patrick E. O'Neil, and Gerhard Weikum. 1993. The LRU-K page replacement algorithm for database disk buffering. SIGMOD Rec. 22, 2 (June 1, 1993), 297–306. DOI:https://doi.org/10.1145/170036.170081

[4] Peter J. Denning. 2005. The locality principle. Commun. ACM 48, 7 (July 2005), 19–24. DOI:https://doi.org/10.1145/1070838.1070856

[5] P. Lu, K. Shen, “Virtual Machine Memory Access Tracing with Hypervisor Exclusive Cache,” In USENIX Annual Technical Conf. (USENIX), pp. 29-43, June 2007.

[6] Yuanyuan Zhou, James Philbin, Kai Li, “The Multi-Queue Replacement Algorithm for Second Level Buffer Caches,” In Proceedings of the General Track: 2001 USENIX Annual Technical Conference. USENIX Association, USA, pp. 91–104, 2001

[7] Denning, P.J. “The Working Set Model for Program Behavior,” 1968 Communications of the ACM (May. 1968).

[8] Coffman, E.G. Jr., and Denning, P.J. “Operating Systems Theory, Prentice Hall Professional Technical Reference.” (Oct. 1973).

[9] Hiroki Sugimoto, Kenichi Kourai, and Saneyasu Yamaguchi. 2015. Host OS page cache hit ratio improvement based on guest OS page drop. In Proceedings of the 17th International Conference on Information Integration and Web-based Applications & Services (iiWAS '15). Association for Computing Machinery, New York, NY, USA, Article 77, 1–4. DOI:https://doi.org/10.1145/2837185.2837267

[10] Y. Zhou, Z. Chen and K. Li, "Second-level buffer cache management," in IEEE Transactions on Parallel and Distributed Systems, vol. 15, no. 6, pp. 505-519, June 2004, doi: 10.1109/TPDS.2004.13.

[11] Zhan Shi, Xiangru Huang, Akanksha Jain, and Calvin Lin. 2019. Applying Deep Learning to the Cache Replacement Problem. In Proceedings of the 52nd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO '52). Association for Computing Machinery, New York, NY, USA, 413–425. DOI:https://doi.org/10.1145/3352460.3358319

[12] Taisei Miura, Kenichi Kourai, Saneyasu Yamaguchi, "Cache Replacement Based on LSTM in the Second Cache in Virtualized Environment," In Proceedings of the 11th International Workshop on Advances in Networking and Computing (WANC 2020), November 27, 2020, Okinawa (virtual).

[13] Theodore M. Wong, John Wilkes, "My Cache or Yours? Making Storage More Exclusive," In Proceedings of the General Track of the annual conference on USENIX Annual Technical Conference (ATEC '02). USENIX Association, USA, pp. 161–175, 2002.

[14] T. Johnson and D. Shasha, “2Q: A Low Overhead High Performance Buffer Management Replacement Algorithm,” in 1994 Proceedings of the 20th International Conference on Very Large Data Bases, Sep. 1994.

[15] Nimrod Megiddo and Dharmendra S. Modha. ARC: a self-tuning, low overhead replacement cache. In FAST’ 03: Proceedings of the 2nd USENIX Conference on File and Storage Technologies, pages 115–130, Berkeley, CA, USA, 2003. USENIX Association.

[16] SNIA - Storage Networking Industry Association: IOTTA Repository Home http://iotta.snia.org/ <accessed Dec. 24, 2020>