Improving the Performance Competitive Ratios of Transactional Memory Contention Managers

Gokarna Sharma[1] Costas Busch[1]

Abstract

In this work, we consider contention management in transactional memory in the context of *balanced workloads* which exhibit better performance potential. Balanced workloads include transactions whose number of exclusive accesses to the shared resources for writes are within a constant fraction of the total accesses (read or writes) to the shared resources. We explore the theoretical performance boundaries of contention management for balanced workloads from the worst-case perspective. We present and analyze two new randomized contention management algorithms which achieve competitive ratio very close to $O(\sqrt{s})$, where s is the number of shared resources. We also prove a matching lower bound in competitive ratio by showing that no contention management algorithm can be better than $O(\sqrt{s})$-competitive ratio for balanced workloads, unless NP$\subseteq$ZPP. To our knowledge, these results demonstrate a significant improvement over the previously best bound of $O(s)$.

1 Introduction

A *contention management* policy for ensuring progress in transactional memory (TM) is often evaluated by its *competitive ratio* - the ratio of the *makespan* (the total time to perform all transactions) of the policy, in comparison to the makespan of the optimal offline policy. A major challenge and research focus in contention management is to schedule transactions in such a way that reduces the makespan, since transactions are often aborted and restarted. The makespan primarily depends on the *workload* $-$ the set of transactions, and their attributes such as their arrival times, durations, and the resources they read and modify [2].

The first formal analysis of the performance of a contention manager was given by Guerraoui *et al.* [3] where they present the Greedy contention manager and prove it achieves $O(s^2)$ competitive ratio for n concurrent transactions that share s objects. Attiya *et al.* [1] improved the result of [3] to $O(s)$, for a type of non-clairvoyant schedulers that don't assume any knowledge about the transactions during execution. They also proved a matching lower bound of $\Omega(s)$ for their scheduler model. Recently, Attiya *et al.* [2] proposed the Bimodal scheduler which achieves also $O(s)$ competitive ratio on bimodal workloads with equi-length transactions, which contains only early-write and read-only transactions.

Therefore, the best competitive ratio proven so far to be achievable using simple (and greedy) contention managers is $\Theta(s)$. These results are not encouraging, e.g., for the $O(s)$ competitive ratio bound of [1], when s increases, the performance degrades linearly. A natural question is if we can obtain better asymptotical bounds. A difficulty in obtaining better competitive ratios is that the scheduling problem of n concurrent transactions is directly related with vertex coloring. Since it is known that computing the chromatic number of an arbitrary graph is a very hard problem to approximate, the optimal schedule for the transaction scheduling problem is very hard to approximate too [4].

Here, we are interested in finding the best possible bounds for transaction schedulers within the limitations posed by the vertex coloring problem. One challenge is to determine what can or cannot be achieved using reasonable assumptions for the contention management policies. To address the challenge, we explore schedulers on *balanced transactions* were each transaction performs total number of writes linearly proportional to the total number of shared resources that it accesses. We call these types of transaction workloads as *balanced transaction workloads*. Using balanced workloads we are able to obtain an $O(\sqrt{s})$-competitive ratio. We also obtain a matching lower bound by reducing the vertex coloring problem to the balanced transaction workload problem. This is a significant improvement over the previously known bounds.

2 Results

Balanced Workloads. We define balanced transaction workloads as follows. Consider a set of transactions $\mathcal{T}$. For any transaction $T_i \in \mathcal{T}$, let $R_w(T_i)$ and $R_r(T_i)$ denote the set of transactions which are being accessed by T_i for write and read, respectively. Let $R(T_i) = R_w(T_i) \cup R_r(T_i)$. A transaction T_i is *balanced*

[1]Department of Computer Science, Louisiana State University, Baton Rouge, LA 70803, USA, {gokarna, busch}@csc.lsu.edu

if the number of writes to the shared resources, and the total number of accesses (the number of total reads and writes) to the shared resources, are within a constant fraction of each other. Formally, $\frac{|\mathcal{R}_w(T_i)|}{|\mathcal{R}(T_i)|} \geq \beta$, where $\frac{1}{s} \leq \beta \leq 1$ is some constant. In other words, we say that the transaction T_i needs exclusive access to resources for writes to at most a β-fraction of its total accesses. For a *read-only* transaction, $|\mathcal{R}_w(T_i)| = 0$, whereas $|\mathcal{R}_r(T_i)| = 0$, for a *write-only* transaction. The relation $\beta \geq \frac{1}{s}$ is due to the assumption that there will be at least one exclusive write that is performed by a balanced transaction. A workload (transaction set) $\mathcal{T}$ is *balanced* if it contains only balanced transactions, and we also say that a transactional memory system is balanced if it executes only balanced workloads.

Upper Bound. In this work, we explore upper bounds of the performance of contention management on balanced transactional memory. We present and analyze two new randomized contention management algorithms. The first algorithm Clairvoyant gives an $O(\sqrt{s})$ competitive ratio, where s is the number of shared resources. This algorithm depends on knowing the conflict dependencies (i.e. conflict graph) which evolve while the execution of the transactions progresses. The second algorithm Non-Clairvoyant gives an $O(\sqrt{s} \cdot \log n)$ competitive ratio, with high probability (at least $(1 - n^{-1})$), which is only a $O(\log n)$ factor worse, but does not require knowledge of the conflict graph, where n is the number of transactions.

Both of the algorithms rely on knowing ahead the execution duration of a transaction and also the number of resources that a transaction accesses. Let $\tau_{\max}$ and $\tau_{\min}$ denote the worst and best, respectively, execution duration of any transaction. The idea behind both of the algorithms is to divide the set of transactions $\mathcal{T}$ into ℓ groups $A_0, A_1, \cdots, A_{\ell-1}$ according to execution time, where $\ell = \left\lceil \log\left(\frac{\tau_{\max}}{\tau_{\min}}\right)\right\rceil + 1$, in such a way that A_i contains transactions whose execution time is in range $[2^i \cdot \tau_{\min}, (2^{i+1} - 1) \cdot \tau_{\min}]$, for $0 \leq i \leq \ell - 1$. Each group of transactions A_i is then again divided into κ subgroups $A_i^0, A_i^1, \ldots, A_i^{\kappa-1}$, where $\kappa = \lceil \log s \rceil + 1$, such that each transaction $T \in A_i^j$ accesses (for read and write) a number of resources in range $\lambda^j \in [2^j, 2^{j+1} - 1]$, for $0 \leq j \leq \kappa - 1$.

We assign a total order in the subgroups such that $A_i^j < A_k^l$ if $i < k$ or $i = k \wedge j < l$. The total order is used to give priorities to the transactions in case of conflicts, so that lower order subgroups have higher priority. In our first algorithm, the priorities within a subgroup by which transactions commit at the time of conflict are determined according to the dynamic conflict graph, while our second algorithm uses the variation of RandomizedRounds algorithm of [5] to resolve conflict without knowledge of the conflict graph. Both of the algorithms are greedy.

In the analysis, we are able to prove that each subgroup of transactions A_i^j, if executed in isolation, has competitive ratio $O(\min(\lambda^j, \frac{s/\beta}{\lambda^j}))$. When we combine all the subgroups of a group A_i, we obtain competitive ratio $O(\sqrt{\frac{s}{\beta}})$ for a group. When we combine all the groups the competitive ratio becomes $O(\ell \cdot \sqrt{\frac{s}{\beta}})$. For constant ℓ and β, we obtain the final result of $O(\sqrt{s})$ for the first algorithm. The second algorithm is randomized and has an additional $\log n$ factor due to randomization.

Lower Bound. We prove a lower bound in the competitive ratio of the first algorithm Clairvoyant which shows that it is worst-case optimal, that is, there exists scenarios for which no contention management algorithm can do better than Clairvoyant in balanced workloads of our model, unless NP⊆ZPP. This lower bound is proved by reducing the well known NP-complete problem VERTEX COLORING, which asks whether a given graph G is k-colorable, to BALANCED TRANSACTION SCHEDULE, the transactional scheduling problem, which asks whether a given dependency conflict graph of balanced transactions G' can produce a schedule of length k. The number of nodes in G is n bounded by $s \leq n^2$. The lower bound follows since it is known that no better than n approximation exists for vertex coloring unless NP⊆ZPP.

References

[1] Attiya, H., Epstein, L., Shachnai, H., Tamir, T.: Transactional contention management as a non-clairvoyant scheduling problem. Algorithmica 57(1), 44–61 (2010)

[2] Attiya, H., Milani, A.: Transactional scheduling for read-dominated workloads. In: OPODIS '09. pp. 3–17 (2009)

[3] Guerraoui, R., Herlihy, M., Pochon, B.: Toward a theory of transactional contention managers. In: PODC '05 (2005)

[4] Khot, S.: Improved inapproximability results for maxclique, chromatic number and approximate graph coloring. In: FOCS '01. pp. 600–609. IEEE Computer Society, Washington, DC, USA (2001)

[5] Schneider, J., Wattenhofer, R.: Bounds on contention management algorithms. In: ISAAC '09 (2009)