==========

NOSSDAV 2012 Review #25A
Updated Thursday 29 Mar 2012 8:07:44pm EDT

Paper #25: LEARS: A Lockless, Relaxed Atomicity State Model for Parallel Execution in Game Servers

---

Overall merit: 1. Reject
Reviewer expertise: 4. Expert
Novelty: 1. Published before

===== Paper summary =====

This paper adresses the scalability limitations of single-threaded MMOG servers, and proposes a parallelization scheme of the game event loop on multicore processors. The work is based on the observation that in games, events occurring during the same tick (the smallest time period in a game) are considered as simultaneous, meaning that they can be parallelized or reordered. As such, the authors' main focus is to achieve high scalability in parallelized game servers by avoiding the synchronization overhead of concurrent events scheduled for execution during the same tick. Based on the argument that game servers are, anyhow, inaccurate simulators in which game semantics are oftentimes relaxed, the authors propose the following approach to concurrency: entities can only update their own state and read any state without locking. If an entity needs to update the state of another entity, the state update request is passed on to the target entity via update message queues.

===== Comments for authors =====

On the positive side, parallelizing game server software is a very interesting and timely topic. On the negative side, however, the present work has several major weaknesses summarized below:

- Key elements of the proposed approach are not novel. The main observation on which the present work is based (i.e., events occurring during the same tick can be parallelized/reordered) has already been exploited in [1,2]. Similarly, the single-writer approach for ensuring that write operations do not conflict, with entities communicating only through "assignment" requests, has also been proposed in [1,2].

- Another major issue with the proposed approach is giving up on synchronizing concurrent read/write operations. This will result in inconsistent views of the game state during the same (logical) time unit, which can significantly impact the user Quality of Experience (QoE). Also, although state updates are only performed by the entity itself, update requests that are enqueued for later execution are generated in parallel during the same tick. How is the execution of an entity's updates guaranteed to produce the correct value, such as the effect of an update cancelled by another?

- From the above, the proposed solution simply reduces to ignoring concurrency issues. While the authors argue that game state consistency is not always needed, no rigorous study is conducted to support this claim or to give insight into the game events for which atomicity can/cannot be relaxed. Also, the authors state that "the end result of the proposed design philosophy is that there is no synchronization in the server under normal running conditions" without, however, providing any clear identification of the "normal running conditions" under which synchronization is claimed unnecessary. Overall, the discussion is very high level, and several important details at the heart of the proposed solution are missing. In comparison, the solution proposed in [1,2] allows a high degree of parallelism, and guarantees consistent reads and writes while avoiding the high cost of lock-based synchronization or rollbacks incurred by traditional concurrency control mechanisms.

- The evaluation of the work does not provide more insight into the above mentioned issues. It would have been crucial to evaluate the inconsistencies incurred by the absence of synchronization and their impact on the game play, and to compare the proposed approach with other existing multi-threaded solutions in terms of the overall user QoE. Without such evaluations, it is not possible to assess the value of the present work.

[1] Guozhang Wang, Marcos Vaz Salles, Benjamin Sowell, Xun Wang, Tuan Cao, Alan Demers, Johannes Gehrke, Walker White. Behavioral Simulations in MapReduce. PVLDB 3(1): 952-963 (2010).

[2] Benjamin Sowell, Alan Demers, Johannes Gehrke, Nitin Gupta, Haoyuan Li, and Walker White, From Declarative Languages to Declarative Processing in Computer Games. In Proc. of the 2009 CIDR Conf. on Innovative Data Systems Research (CIDR 2009).

==========

NOSSDAV 2012 Review #25B
Updated Tuesday 3 Apr 2012 1:47:45am EDT

Paper #25: LEARS: A Lockless, Relaxed Atomicity State Model for Parallel Execution in Game Servers

---

Overall merit: 3. Weak accept
Reviewer expertise: 3. Knowledgeable
Novelty: 4. Novel

===== Paper summary =====

This paper presents LEARS, which aims for a lockless and highly parallel execution model for MMOGs to increase the scalability of such games. I find the proposed model simple yet useful. It would be controversial and I believe it would draw interesting discussions on the workshop.

===== Comments for authors =====

- One key point for the model to work properly is to define work units properly. The authors should explain more and give examples (the more the better) on how they define work units.

- Figure 1 is very unclear. Why the "position update" is separate from others (which should be part of Character Update and Projectile Update)? Why "Execute Workload" is a separate work unit? Also, what do "network worker" and "network selector" do? What are the relationships between these work units and the thread pool is not illustrated. Thus, Figure 1 contains no new information to me.

- One burning question is from the performance comparison of single-threaded server and multi-threaded server. I wonder how many instances the authors created for the single-threaded server. This is very important. If you only created one instance, it's certainly the single-threaded server performs worse. However, if you created eight instances, the result of performance comparison may be very different. The authors should explicitly explain this point and evaluate the effect of the number of instances on the performance of single-threaded server, or the comparison is not fair.

- Figure 6 is unclear. Since we focus on the left most part (when the size of thread pool is small), the authors should zoom in the left most part, or provide another figure for the part. It is not clear how the data in Figure 6 is obtained (e.g., how many iterations). In addition, the increasing 5-percentile when thread pool size > 150 is worth mentioning.

- In Section 4.3, the authors mentioned that threads are occasionally waiting for I/O operations from Figure 6. Given that a thread pool with size 40 is finally

sufficient to saturate a 8-core machine, I wonder how many work units are involved with I/O operations. Normally in most ticks, only reading from/writing to sockets are involved in I/O operations, which I am not sure whether belonging to work units or not. The authors should explain more on this (how work units are defined and why so many work units are involved with I/O operations).
- This model would incur many more context switches than in traditional models. The authors are recommended to evaluate the overhead on context switching in their future works.

- The authors are much encouraged to apply their model to more realistic games, such as one of the open-sourced MMOGs, to validate their model.

- Typos
  * the second line of Section 4.2, "multhreaded" => "multi-threaded"
  * the first paragraph of Section 5, "investigate" => "investigates"
  * the first paragraph of Section 6, "invastigates" => "investigates"
  * the second paragraph of Section 6, "divide" => "divides"

==========

NOSSDAV 2012 Review #25C
Updated Thursday 5 Apr 2012 2:29:20am EDT

Paper #25: LEARS: A Lockless, Relaxed Atomicity State Model for Parallel Execution in Game Servers

---

Overall merit: 2. Weak reject
Reviewer expertise: 3. Knowledgeable
Novelty: 2. Very incremental

===== Paper summary =====

The paper proposes the design and implementation of LEARS, a model to allow a better resource allocation for the execution of game servers in multiprocessor machines. The model is based on splitting the game server simulation into small work units that are scheduled and executed by using a pool of existing threads (thread pool design pattern). Each work unit runs independently in a lockless message-based fashion. The authors implement the presented model and conduct measurements on a prototype game, comparing the classic single-threaded execution with the LEARS multithreaded one. Finally they present the results and draw the conclusions.

===== Comments for authors =====

Comments and evaluation (weak, strong points)

The paper has a clear structure and is well written in general. The topic is current and relevant. The paper describes an interesting solution to the problem of supporting an higher number of players by a single authoritative machine that executes the game simulation. The solution proposed with LEARS is orthogonal to many other solutions that achieve scalability through interest management.

However, in few parts the paper could be made more readable. Starting from the figures where Fig. 1 lacks a description and/or a reference in the text, and Fig. 4 is not readable when printed in black&white. Moreover, the authors are very sloppy in the justification of the relaxation of the consistency constraints at the end of Section 2.

The paper lacks of a exhaustive description of communication between the two work units, in particular what a "message" encapsulates. The transition between a chain of read/write accesses to two active elements (or entities) performed by the game server during an ordinary simulation (physics, collisions, etc.) is not intuitively translatable into a "message". This makes the limitations presented in Section 5 not clear. A simple two lines example would have been enough to clarify the communication.

The measurements, although convincing and intuitive, are conducted only for latency. However, Quality of Experience (QoE) is not only about latency but also consistency. The paper should also measure how relaxing the consistency constraints impacts the QoE. A starting point would be to measure how the "precision" described by Claypool et al. in [6] is affected and whether plausibility and fairness [Steed2009] are maintained in the "embarrassingly parallel" environment.

References

[Steed2009] Anthony Steed, Manuel Fradinho Duarte de Oliveira: Networked Graphics - Building Networked Games and Virtual Environments. Academic Press 2009, http://www.networkedgraphics.org/