Understanding Performance Bottlenecks of Large-Scale Agent-Based Simulations

Amina Sokoli*, Abdullah Giray Yaglikci*

*CISPA Helmholtz Center for Information Security

ABSTRACT

Agent-based simulations are growing in scale and complexity, yet their hardware performance characteristics remain poorly understood. To understand the data movement-related performance bottlenecks of agent-based simulations, we profile BioDynaMo, the state-of-the-art agent-based simulation framework, on Google Cloud servers adopting DAMOV's characterization methodology. Our results demonstrate that memory bandwidth and capacity are the primary bottlenecks limiting scalability and motivate a more rigorous investigation.
KEYWORDS: agent-based simulation; high-performance computing; scalability

1 Introduction

Agent-based simulation is a bottom-up modeling approach, where a complex system is modeled as a set of agents and interactions across those agents. It has been applied across diverse domains including biology, epidemiology, finance, and social sciences. Enabling fast and efficient simulations of large ABMs can lead to significant leaps and breakthroughs in a diverse set of research fields (e.g., [3, 4]). Unfortunately, *no* prior works rigorously and methodically investigate the performance bottlenecks of agent-based simulations that prevent them from scaling to multi-billions of agents efficiently.

We present the first rigorous and methodical evaluation of performance bottlenecks of agent-based simulations at its early stage. We hypothesize that data movement is a major component of performance and scalability limitations similar to many other workloads [5]. To test this hypothesis, we create a realistic test case of agent-based simulations, where we simulate epidemiology using the state-of-the-art agent-based simulation framework, BioDynaMo [1, 2], on virtual machines on Google Cloud. We sweep the number of agents from a modest 10 million to 200 million and measure data movement-related performance metrics [5].[2] Our results demonstrate that 1) memory capacity is a major limitation of agent count scaling; 2) despite good L1-cache locality, all runs exhibit significant memory boundness; and 3) prefetchers and higher level L2 cache perform poorly.

[1]E-mail: amiamina559@gmail.com, giray@cispa.de
[2]Upper bound of agent count in this study is enforced by the memory capacity.

2 Methodology

Experimental Setup and Reproducibility. Table 1 shows the specifications of the virtual machine (VM) instances that we used on Google Cloud. To ensure low-noise measurements on a cloud environment, we conduct two analyses. First, we executed a representative run for more than 24 hours on weekdays and observed less than 8% coefficient of variation across measurements. Second, we spawn three separate VMs for each run and report results from all three VMs. BioDynaMo runs natively on each VM instance. Each of our simulations run for 3000 iterations and we repeat each simulation three times to account for measurement noise.

Table 1: Google Cloud C4 VM Specs

Processor	16-core Intel Sapphire Rapids
Memory	64GB with four NUMA domains
OS	Ubuntu 22.04 LTS

Metrics. We use `perf` at 1 Hz sampling rate to collect performance monitoring unit (PMU) statistics. Table 2 lists the collected PMU statistics about the processor and memory instructions, cycles, cache behavior, and memory performance metrics throughout the simulation.

Table 2: PMU Statistics

$instructions$	Number of instructions
$cycles$	Number of CPU cycles
$L2\_RQSTS.MISS$	Number of L2 misses
$L2\_RQSTS.REFERENCES$	Number of requests sent to the L2 cache
$MEM\_LOAD\_COMPLETED.L1\_MISS\_ANY$	Number of load instructions that completed execution but missed in the L1 data cache
$MEM\_LOAD\_RETIRED.L1\_HIT$	Number of retired instructions that cause an L1 hit
$MEM\_LOAD\_RETIRED.L2\_HIT$	Number of retired instructions that cause an L2 hit
$MEM\_LOAD\_RETIRED.L1\_MISS$	Number of retired instructions that cause an L1 miss
$MEM\_LOAD\_RETIRED.L2\_MISS$	Number of retired instructions that cause an L2 miss
$TOPDOWN.SLOTS\_P$	Total number of slots in the processor pipeline
$TOPDOWN.MEMORY\_BOUND\_SLOTS$	The number of processor pipeline slots that are memory bound
$MEMORY\_ACTIVITY.STALLS\_L1D\_MISS$	Number of cycles wasted due to L1 data cache misses
$MEMORY\_ACTIVITY.STALLS\_L2\_MISS$	Number of cycles wasted due to L2 cache misses

We use these statistics to analyze four key metrics: 1) instructions per cycle (IPC), 2) misses per kilo instructions at the last level cache (LLC MPKI), 3) fraction of pipeline slots bound by memory (Memory-bound fraction), and 4) last-to-first miss ratio (LFMR), showing the fraction of L1 misses that end up accessing main memory.

Benchmarks. We simulate epidemiology on BioDynaMo, modeling the spread of infectious diseases across a population of agents. We scale the population from 10 million up to 1 billion agents with data points at 10M, 50M, 100M, 200M, 225M, 250M, 500M, and 1B agents.

3 Results and Analysis

We characterize the performance of BioDynaMo's epidemiology use case across four agent counts (10M: blue, 50M: orange, 100M: green, 200M: red) using DAMOV-aligned metrics.[3] Figs. 1 and 2 present IPC, LLC MPKI, memory-bound fraction, and LFMR, respectively. In each figure, the X-axis scales the number of agents, the Y-axis reports the corresponding metric. Each colored point represents a measurement from one of the three independent VM instances. Gray data points show the average measurement across three VMs, and the solid line connects the per-agent-count medians.

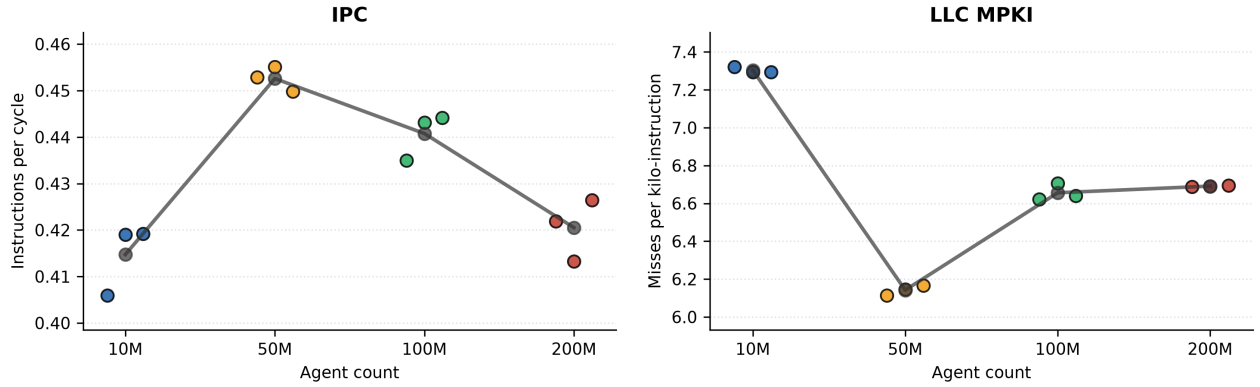

Figure 1: IPC (`instructions/cycles`) and LLC MPKI, derived from $MEM\_LOAD\_RETIRED.L2\_MISS$

IPC and LLC MPKI. We observe that 1) IPC remains consistently low (<0.5) across all runs; 2) LLC MPKI remains consistently stable across all agent counts, ranging in values from 6 to 8; 3) both IPC and LLC MPKI exhibits a non-monotonic trend with worst result at 10M agents, best at 50M agents, and monotonically worsening results at larger agent counts. The cause of the non-monotonic behavior remains under investigation and is identified as a key direction for future work. From Fig. 1 we conclude that this simulation has a large room for improvement and memory can be a performance bottleneck.

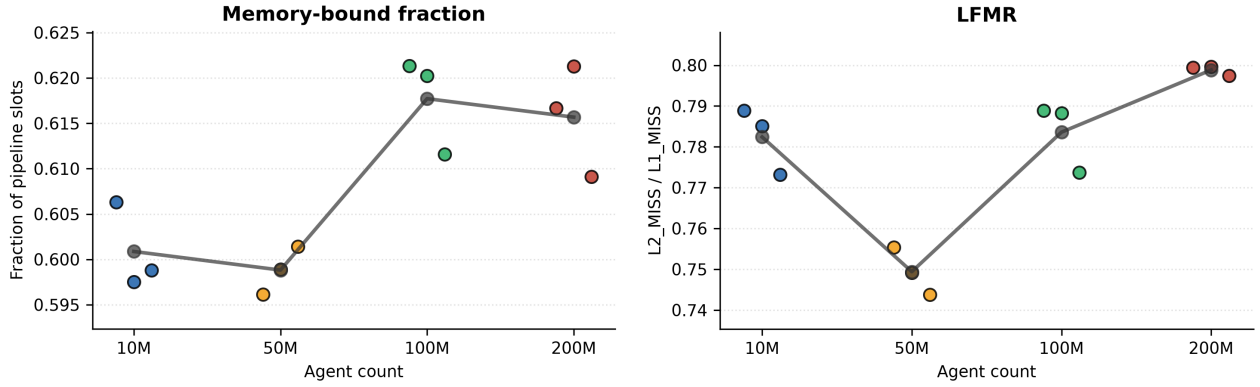

Figure 2: Memory-bound fraction and LFMR across agent counts. and the memory-bound fraction was measured via `TOPDOWN.MEMORY_BOUND_SLOTS / TOPDOWN.SLOTS_P`

[3]Simulations of 225M+ agents exhaust memory capacity and are terminated by the Linux OOM killer in all VM instances. The 200M agent simulation consumed approximately 58GB at peak, corresponding to ≈ 290 bytes per agent. Thus, 200M agents is the practical upper bound for this VM configuration.

Memory-bound fraction and LFMR. We observe that 1) memory-bound fraction shows that the workload is memory-bound for over 60% of pipeline slots across all agent counts and 2) LFMR remains high across all agent counts, indicating that the majority of L1 misses also miss L2. From Fig. 2, we conclude that despite the moderate LLC MPKI, performance is bottlenecked by memory accesses and L2 cache is largely ineffective as most of L1 misses escalate to main memory. Taken together, these metrics place this workload in the *memory-bound* region of the DAMOV classification tree[5]. A precise classification requires additional measurements, including temporal and spatial locality, which we plan for future work.

4 Conclusion and Future Work

We characterize BioDynaMo's epidemiology use case using the DAMOV methodology across agent counts from 10M to 200M on GCP C4 VMs. From our observations of LLC MPKI, memory-bound fraction, IPC, and LFMR, we conclude that memory latency is the dominant performance bottleneck of this workload, and that L2 cache is largely ineffective as a catch for L1 misses. These findings suggest that naively increasing parallelism may not improve performance and could worsen memory bottlenecks. We further plan to characterize temporal and spatial locality for a precise DAMOV classification, validate our methodology against a simulation-based setup, and extend experiments to GPU execution, where we expect parallelism to be embraced and memory bottlenecks to become more pronounced.

References

[1] L. Breitwieser et al. High-performance and scalable agent-based simulation with biodynamo. In *Proceedings of the 28th ACM SIGPLAN Annual Symposium on Principles and Practice of Parallel Programming*, PPoPP '23, page 174â188, New York, NY, USA, 2023. Association for Computing Machinery.

[2] L. Breitwieser, A. Hesam, J. de Montigny, V. Vavourakis, A. Iosif, J. Jennings, M. Kaiser, M. Manca, A. Di Meglio, Z. Al-Ars, F. Rademakers, O. Mutlu, and R. Bauer. BioDynaMo: a modular platform for high-performance agent-based simulation. *Bioinformatics*, 38(2):453–460, 09 2021.

[3] J. de Montigny et al. An in silico hybrid continuum-/agent-based procedure to modelling cancer development: Interrogating the interplay amongst glioma invasion, vascularity and necrosis. *Methods*, 185:94–104, 2021. Methods on simulation in biomedicine.

[4] T. Duswald et al. Bridging scales: A hybrid model to simulate vascular tumor growth and treatment response. *Computer Methods in Applied Mechanics and Engineering*, 418:116566, 2024.

[5] G. F. Oliveira et al. DAMOV: A New Methodology and Benchmark Suite for Evaluating Data Movement Bottlenecks. *IEEE Access*, 9:134457–134502, 2021. Conference Name: IEEE Access.