

Um Simulador de Eventos Discretos Multiagentes Distribuído

Francisco Wallison Rocha¹, Emilio Francesquini², Daniel Cordeiro¹

¹Escola de Artes, Ciências e Humanidades — Universidade de São Paulo (USP)
São Paulo — SP — Brasil

{wallison.rocha, daniel.cordeiro}@usp.br

²Centro de Matemática, Computação e Cognição – Universidade Federal do ABC (UFABC)
Santo André — SP — Brasil

e.francesquini@ufabc.edu.br

Abstract. *This work propose a distributed and shared-memory multi-agent discrete event simulator capable of handling large-scale scenarios with efficient load balancing. Our solution executes scenarios with over 4 million agents in 3.5 hours (shared memory) and 9.5 hours (distributed memory).*

Resumo. *Este trabalho propõe um simulador de eventos discretos multiagentes, projetado para memória compartilhada e distribuída, focado em lidar com cenários de larga escala com balanceamento de carga eficiente. Resultados mostram a execução de cenários com mais de 4 milhões de agentes em 3,5 h (memória compartilhada) e 9,5 h (memória distribuída).*

1. Introdução

A sociedade enfrenta as graves consequências das mudanças climáticas, impulsionadas primordialmente pela ação humana [Shukla et al. 2022]. A queima de combustíveis fósseis é um dos principais agravantes, liberando grandes quantidades de CO₂. Segundo o relatório de 2022 do *Intergovernmental Panel on Climate Change* (IPCC) [Shukla et al. 2022], o CO₂ correspondeu a aproximadamente 60% dos gases de efeito estufa lançados na atmosfera entre 1990 e 2019, emissões que continuam crescendo devido ao consumo global.

Para avaliar e propor soluções urbanas e climáticas, pesquisadores utilizam Simulações de Eventos Discretos (DES — *Discrete Event Simulation*) [Babulak and Wang 2010]. A DES modela a dinâmica de sistemas reais avançando o tempo conforme os eventos ocorrem. Contudo, simuladores tradicionais processam eventos de forma sequencial, o que torna a simulação microscópica de sistemas complexos um gargalo computacional imenso em cenários de larga escala. Para mitigar esse problema, recorre-se à Simulação Paralela de Eventos Discretos (PDES). O grande desafio da PDES é dividir a carga de trabalho garantindo a causalidade (impedindo que eventos futuros alterem estados passados). Simuladores de tráfego conhecidos, como SUMO [Alvarez Lopez et al. 2018] e MATSim [W Axhausen et al. 2016], apresentam paralelismo limitado e não foram concebidos para funcionar nativamente com memória distribuída. Uma alternativa é o InterSCSimulator [Santana et al. 2018], que suporta execução distribuída; todavia, ele apresenta limitações de escalabilidade, uso ineficiente de CPU (limitado a 50%) e alta complexidade de implantação para usuários comuns.

Neste contexto, este trabalho propõe o Hyperbolic Time Chamber (HTC), um novo motor de simulação de eventos discretos com alto grau de paralelismo, projetado para execução escalável em arquiteturas de memória compartilhada e distribuída.

2. Trabalhos Relacionados

Em arquiteturas distribuídas e sistemas multiagentes, o desafio central para *Digital Twins* é evitar gargalos de comunicação e manter a sincronização de estados. Abordagens recentes utilizam tecnologias de mensageria assíncrona (Apache Flink e Kafka) [Van Den Berghe 2021] ou a plataforma MARS com o protocolo MQTT [Clemen et al. 2021] para garantir a readequação de agentes virtuais ao mundo físico. Observa-se também a transição para microsserviços [Bellini et al. 2025] e estratégias descentralizadas em memória RAM, como o *Inter Model Broker* (IMB) [Lohman et al. 2023], visando contornar a latência dos bancos de dados tradicionais.

Na simulação urbana microscópica, a exaustiva carga computacional impõe severos limites de escala [Yeon et al. 2023]. Como alternativas, a literatura adota arquiteturas baseadas em consultas (*query-based*) [Mavromatis et al. 2020] ou hibridização de granularidade [Schuhmann et al. 2024], reservando o detalhamento extremo apenas para zonas críticas. Apesar desses avanços, a integração fluida entre distribuição descentralizada, reatividade em nível de agente e controle de carga permanece um desafio aberto, lacuna que o HTC busca preencher por meio de sua arquitetura baseada em troca de mensagens.

3. Proposta e Resultados Experimentais

O *Hyperbolic Time Chamber*¹ (HTC) é um motor de simulação de eventos discretos, multiagente, nativamente paralelo, distribuído e de código aberto. Para garantir desempenho estrutural, o HTC foi construído integralmente sobre o *framework* Apache Pekko², um ecossistema concorrente baseado no modelo de atores (em Scala e Java).

A arquitetura do simulador orquestra diversos gerenciadores, com destaque para a gerência de tempo e para os atores de simulação (`SimulationBaseActor`). O controle temporal é organizado hierarquicamente: o `Global Time Manager` (GTM, instanciado como *cluster-singleton*) atua como controlador central da simulação. Para evitar que milhões de atores sobrecarreguem o nó global, o GTM delega o contato direto às instâncias do `Local Time Manager` (LTM). Os LTMs residem localmente nos nós, agregando requisições de escalonamento antes de sincronizar o avanço definitivo com o GTM. O tempo no HTC avança por `Ticks` discretos e dinâmicos (representando milissegundos, segundos ou minutos). O `SimulationBaseActor` acopla as regras de tempo, diferenciando “eventos de interação” (trocados diretamente entre agentes) de “eventos espontâneos” (o despertar autônomo do agente agendado via *Time Manager*). A classe integra dois componentes vitais do Pekko: o `Cluster Sharding`, que distribui e balanceia dinamicamente os atores entre os nós do *cluster*, e o `PersistenceActor`, que salva *snapshots* periódicos para recuperação de estado em caso de rebalanceamento.

Para validar o simulador, empregamos um modelo de mobilidade urbana utilizando a população sintética de Toulouse³ (4.807.663 agentes, 488.325 vértices e

¹<https://github.com/fwrock/hyperbolic-time-chamber>

²<https://pekko.apache.org/>

³<https://eqasim-org.github.io/eqasim-france/cases/toulouse.html>

1.081.827 arestas). Experimentos em *single-node* ocorreram em uma VM do Google Cloud (n2d-highmem-96, dois nós NUMA, 96 vCPUs AMD EPYC e 768 GB de RAM). Já os testes distribuídos foram conduzidos no Google Kubernetes Engine (GKE), particionados em 16 *Pods*. Foi provisionado para cada *pod* até 6 vCPUs e 20 GiB de RAM, totalizando 96 vCPUs e 320 GiB de RAM no *cluster*.

A Tabela 1 compara as execuções. A abordagem no GKE demandou $2,8\times$ mais tempo do que a execução *single-node*. Essa comparação não representa uma medida de speedup, mas evidencia o custo adicional introduzido pela execução distribuída, associado à latência de rede, à serialização/desserialização de mensagens e à maior frequência de ciclos do Garbage Collector, decorrente da menor disponibilidade de memória (320 GB contra 768 GB). O aumento do uso de CPU reflete a carga de rede inerente à distribuição. Ainda assim, isso demonstra a capacidade do HTC de executar cenários em larga escala, mesmo sob restrições severas de infraestrutura.

Tabela 1. Comparação de recursos entre VM (GCP) e GKE durante a simulação.

Métrica	VM (GCP)	GKE (16 pods)
Duração total	3,32 h	9,33 h
Pico de CPU (núcleos)	74,6	68,5
Pico de Heap JVM (GiB)	460,0	173,8
Pico da taxa de mensagens (msg/s)	242 317	89 558

A avaliação do balanceamento de carga entre os 16 *Pods* do GKE demonstrou uma distribuição homogênea dos recursos computacionais. A CPU registrou uma média de 3,00 núcleos por *pod* (variando de 2,73 a 4,23, com coeficiente de variação de 14,6%), enquanto a memória *Heap JVM* manteve média de 8,95 GiB (mínimo de 8,82, máximo de 9,53 e coeficiente de variação de apenas 9,5%). Por outro lado, as taxas de transferência de rede apresentaram maior variação: o RX médio foi de 102,1 GiB (coeficiente de variação de 49,1%) e o TX médio de 93,6 GiB (coeficiente de variação de 43,6%), somando um tráfego total de 3,13 TiB. Essa disparidade na rede é esperada e evidencia o impacto da comunicação associada à coordenação centralizada no GTM.

4. Conclusões e Trabalhos Futuros

Este trabalho apresentou o HTC, um simulador paralelo de eventos discretos baseado em um modelo de distribuição orientado a atores. A abordagem de *Time Managers* hierárquicos combinada com *Relógios de Lamport* foi projetada para preservar a causalidade em larga escala, minimizando gargalos centrais. Adicionalmente, o *Cluster Sharding* proporcionou um balanceamento homogêneo de CPU e de memória. Os experimentos confirmaram a viabilidade de simular cidades com quase 5 milhões de agentes em tempo hábil (3,5 h centralizado; 9,5 h distribuído), mesmo operando sob limites rígidos de recursos computacionais em contêineres.

Como trabalhos futuros, planejamos avaliar novas estratégias de roteamento e algoritmos de particionamento personalizados para mitigar o tráfego de rede no nó central. Exploraremos também a escalabilidade (*speedup*) variando sistematicamente o número de nós no *cluster* e compararemos o desempenho com o de outros simuladores distribuídos de estado da arte.

5. Agradecimentos

Este trabalho foi realizado com apoio da FAPESP (procs. #2019/26702-8, #2021/06867-2 e #2023/00811-0), do CCD Cidades Carbono Neutro (FAPESP #2024/01115-0), do CNPq (proc. #444766/2024-3) e do grupo de pesquisa THUS do Centro Internacional de Pesquisa (IRC) ‘Transitions’ (CNRS, USP e FAPESP CIP #2025/01171-0).

Referências

- Alvarez Lopez, P., Behrisch, M., Bieker-Walz, L., Erdmann, J., Flötteröd, Y.-P., Hilbrich, R., Lücken, L., Rummel, J., Wagner, P., and Wießner, E. (2018). Microscopic traffic simulation using sumo. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 2575–2582. IEEE.
- Babulak, E. and Wang, M. (2010). Discrete event simulation. *Aitor Goti (Hg.): Discrete Event Simulations. Rijeka, Kroatien: Sciyo*, page 1.
- Bellini, P., Palesi, L. A. I., Mereu, F., and Nesi, P. (2025). Scalable framework for behavior execution of mobility and transport digital twins. In *2025 IEEE 11th International Conference on Big Data Computing Service and Machine Learning Applications (Big-DataService)*, pages 1–9. IEEE.
- Clemen, T., Ahmady-Moghaddam, N., Lenfers, U. A., Ocker, F., Osterholz, D., Ströbele, J., and Glake, D. (2021). Multi-agent systems and digital twins for smarter cities. In *Proceedings of the 2021 ACM SIGSIM conference on principles of advanced discrete simulation*, pages 45–55.
- Lohman, W., Cornelissen, H., Borst, J., Klerkx, R., Araghi, Y., and Walraven, E. (2023). Building digital twins of cities using the inter model broker framework. *Future Generation Computer Systems*, 148:501–513.
- Mavromatis, I., Piechocki, R. J., Sooriyabandara, M., and Parekh, A. (2020). Drive: A digital network oracle for cooperative intelligent transportation systems. In *2020 IEEE Symposium on Computers and Communications (ISCC)*, pages 1–7. IEEE.
- Santana, E. F. Z., Lago, N., Kon, F., and Milojevic, D. S. (2018). Interscsimulator: Large-scale traffic simulation in smart cities using erlang. In *Multi-Agent Based Simulation XVIII: International Workshop, MABS 2017, São Paulo, Brazil, May 8-12, 2017, Revised Selected Papers 18*, pages 211–227. Springer.
- Schuhmann, F., Nguyen, N. A., Schweizer, J., Huang, W.-C., and Lienkamp, M. (2024). Creating and validating hybrid large-scale, multi-modal traffic simulations for efficient transport planning. *Smart Cities*, 8(1):2.
- Shukla, P. R. et al. (2022). Climate change 2022: Mitigation of climate change. *Contribution of working group III to the sixth assessment report of the Intergovernmental Panel on Climate Change*, 10:9781009157926.
- Van Den Berghe, S. (2021). A processing architecture for real-time predictive smart city digital twins. In *2021 IEEE International Conference on Big Data (Big Data)*, pages 2867–2874. IEEE.
- W Axhausen, K., Horni, A., and Nagel, K. (2016). *The multi-agent transport simulation MATSim*. Ubiquity Press.
- Yeon, H., Eom, T., Jang, K., and Yeo, J. (2023). Dtumos, digital twin for large-scale urban mobility operating system. *Scientific Reports*, 13(1):5154.