# Simulador de Eventos Discretos Multiagente: Uma Proposta para Execução Distribuída em Larga Escala

**Francisco Wallison Rocha**[1], **Emilio Francesquini**[2], **Daniel Cordeiro**[1]

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

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

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

e.francesquini@ufabc.edu.br

***Abstract.*** *Decision-making in various complex contexts is challenging due to its high cost and implementation difficulties. Simulation has proven to be an approach that can assist professionals and researchers in testing their solutions before applying them. However, simulating such complex scenarios at scale remains a challenge, due to their size and the time required for execution. In this context, this work presents an initial proposal for a large-scale, multi-agent discrete event simulator that runs on shared and distributed memory. The initial results of this simulator are promising, with a simulation involving 1,501,000 agents and the exchange of 7.5 million messages executed in just 7 minutes.*

***Resumo.*** *A tomada de decisão em diversos contextos complexos é um desafio devido ao seu alto custo e à dificuldade de implementação. A simulação tem se mostrado uma abordagem que pode auxiliar profissionais e pesquisadores a testarem suas soluções antes de aplicálas. No entanto, simular esses contextos complexos em larga escala é um desafio, devido ao seu tamanho e ao tempo necessário para a execução. Diante disso, este trabalho apresenta uma proposta inicial de um simulador de eventos discretos multiagente em larga escala, que executa em memória compartilhada e distribuída. Os resultados iniciais deste simulador são promissores, com a execução de uma simulação envolvendo 1.501.000 atores e a troca de 7,5 milhões de mensagens em apenas 7 minutos.*

## 1. Introdução

A simulação de Sistemas Complexos de grande escala ainda é um desafio em aberto na Computação por causa de quão interligados, imprevisíveis e dinâmicos eles são. Essa complexidade é devida ao número de elementos envolvidos nesses contextos e às suas interações. Exemplos desses sistemas são sistemas físicos de partículas, sistemas de transporte e sistemas de saúde [Wang et al. 2024].

Em muitos desses cenários, existem problemas que requerem soluções robustas e não triviais. Essas soluções geralmente demandam uma quantidade significativa de recursos e também um alto custo para serem implementadas [Santana 2019].

Para apoiar profissionais e pesquisadores na resolução ou mitigação desses problemas, diversas abordagens e tecnologias têm sido desenvolvidas para modelar Sistemas

Complexos, sendo um exemplo as simulações. Um exemplo desses simuladores é o InterSCSimulator. O InterSCSimulator é um simulador de larga escala que simula um mapa de cidade ou região com viagens de pedestres e carros, assim como sistemas de metrô e ônibus [Santana et al. 2017]. Embora o InterSCSimulator seja um simulador de larga escala, ele apresenta algumas limitações ao modelar sistemas muito grandes, como o uso excessivo de memória (196 GB) e um tempo de execução de 95 minutos para um cenário de 4 milhões de agentes; além disso, um uso de até 50% da CPU [Santana et al. 2017].

Diante disso, este trabalho apresenta uma proposta inicial de um simulador de eventos discretos multiagente [Drogoul and Ferber 2018]. O simulador aqui proposto visa mitigar as limitações apresentadas pelo InterSCSimulator, permitindo que pesquisadores e profissionais possam executar grandes simulações, como a da cidade de São Paulo, com mais de 12 milhões de agentes envolvidos.

## 2. Trabalhos Relacionados

As simulações são muito úteis na resolução de diversos problemas. Através da modelagem de sistemas complexos, elas permitem testar e validar soluções antes de aplicá-las. Um exemplo é a plataforma de simuladores heterogêneos baseada em *High Level Architecture* (HLA), com um *middleware* para simulação de eventos discretos distribuídos (semelhante a este trabalho). O objetivo era criar um ambiente com execução de alto desempenho de sistemas embarcados em larga escala, heterogêneos e complexos [Brito et al. 2015].

[Lin and Yao 2015] apresentam uma técnica para o balanceamento de carga em simuladores de eventos discretos paralelos e *multithread*. Eles utilizam uma abordagem de *Q-learning* para migrar as cargas das *threads* entre os processos. Este trabalho adota um modelo diferente, *multithread* e distribuído, baseado nos atores do *Apache Pekko*[1]. Além disso, o balanceamento de carga das entidades simuladas fica a cargo do *Apache Pekko Cluster Sharding* [2].

Outro simulador em larga escala é o InterSCSimulator [Santana et al. 2017], um simulador de mobilidade urbana baseado em atores, semelhante a este trabalho. No entanto, ele se restringe à mobilidade e adota um gerenciador de tempo global, ao contrário deste trabalho, que utiliza uma abordagem híbrida com gerenciadores de tempo global e local, além dos Relógios de Lamport [Lamport 1978].

## 3. Proposta e Resultados Experimentais

Dadas as limitações apresentadas em trabalhos anteriores, este trabalho apresenta uma proposta de simulador de eventos discretos em larga escala que executa de maneira paralela e distribuída. Um modelo no qual cada ator representa uma thread executando, e a comunicação ocorre por meio de trocas de mensagens. Para o modelo de atores, este trabalho usa o arcabouço *Apache Pekko*, que é implementado nas linguagens de programação Scala[3]/Java[4] e executa na Java Virtual Machine (JVM). O *Apache Pekko* é arcabouço de código aberto voltado para a criação de aplicações que exigem concorrência, distribuição,

[1]https://pekko.apache.org

[2]https://pekko.apache.org/docs/pekko/current/typed/cluster-sharding-concepts.html

[3]https://www.scala-lang.org

[4]https://www.java.com

resiliência e elasticidade. Ele adota o Modelo de Atores como base, oferecendo abstrações de alto nível que auxiliam na criação de aplicações paralelas e distribuídas com uma maior facilidade.

Por ser um simulador de eventos discretos, o tempo avança de acordo com os eventos que devem ser executados em ordem cronológica. Para gerenciar o tempo de simulação e preservar a ordem dos eventos, foi usado um gerenciador de tempo global, no qual os atores envolvidos na simulação agendam seus eventos, e ele coordena os avanços das unidades de tempo (chamadas de *Tick*). Porém, esse gerenciador pode se tornar um gargalo na simulação. Para mitigar esse problema, este trabalho propõe uma abordagem que usa um pool de gerenciadores de tempo locais. Esses gerenciadores de tempo locais são responsáveis por se comunicar diretamente com os demais atores da simulação, deixando assim apenas a responsabilidade de sincronizar o tempo com o gerenciador global, diminuindo possíveis gargalos ocasionados por haver apenas um único gerenciador.

Além de garantir a ordem dos eventos em relação ao tempo global da simulação, por se tratar de um sistema multiagente, também é necessário garantir a ordem dos eventos que ocorrem dentro de uma unidade de tempo menor dentro de um *Tick* (chamada de *Subtick*), mediante a troca de mensagens entre os agentes envolvidos. Para evitar uma sobrecarga de mensagens entre os atores da simulação e os gerenciadores locais, este trabalho implementou o algoritmo do Relógio de Lamport [Lamport 1978] para viabilizar e garantir o controle da ordem desses eventos.

Além das abordagens voltadas à otimização do desempenho da simulação, o simulador foi projetado para possibilitar a modelagem de diversos sistemas complexos, incluindo, mas não se limitando a, sistemas de mobilidade urbana. Diferentemente de outros simuladores, que possuem um escopo restrito, este modelo de simulação busca oferecer mais flexibilidade para adaptações e expansões futuras.

Para avaliar a proposta inicial, foram realizados experimentos em uma máquina com processador Intel® Core™ i7-150U ×12, com 32GB de memória RAM. Embora seja uma aplicação distríbuída, a execução foi realizada utilizando apenas uma máquina. O sistema simulado foi um sistema de filas clássico de caixas de supermercado. Nesse sistema, o cliente chega ao caixa; se estiver livre, ele é atendido, senão, é colocado em uma fila e aguarda sua vez. É definida uma duração para cada etapa, e também é agendado um evento no gerenciador para que essa etapa possa acontecer. Na simulação, estavam envolvidos cerca de 1.501.000 atores, sendo 1,5 milhão de clientes e 1 mil caixas de supermercado. O tempo da simulação foi de 43.202 *ticks* (cada *tick* correspondendo a um segundo no mundo real). Como resultado deste experimento, foi obtida uma taxa de troca de mensagens de aproximadamente $17.857 msg/s$, em que $msg$ representa o número de mensagens transmitidas em $s$ segundos. Os eventos da simulação foram gerados de forma aleatória, com o objetivo de testar o sistema de filas e o desempenho do simulador proposto neste trabalho.

## 4. Conclusões e Trabalhos Futuros

Este trabalho apresentou uma proposta inicial de um simulador de eventos discretos com execução paralela e distribuída. Destaca-se a abordagem híbrida de gerenciamento de tempo, que utiliza um gerenciador global com um pool de gerenciadores locais, além do uso do Relógio de Lamport para garantir a ordem na comunicação entre os agentes

envolvidos na simulação.

Os resultados obtidos nos experimentos realizados mostram que a proposta inicial de um novo simulador é promissora, executando aproximadamente 1,5 milhão de atores, com uma taxa de troca de mensagens de aproximadamente $17.857 msg/s$.

Como trabalhos futuros, podemos destacar a realização de experimentos com uma quantidade maior de agentes envolvidos, na casa das dezenas de milhões. Outro trabalho a ser realizado é a implementação de uma simulação de tráfego urbano e a realização de comparações de desempenho com o InterSCSimulator e outros simuladores. Um outro trabalho a ser relaizado, será testar a proposta com outras configuações de máquina, com a finalidade de avaliar a sua escalabilidade. Além disso, comparar o gerenciador de tempo global com outros modelos de gerenciar o tempo com a finalidade de melhorar o tempo de execução da simulação.

## 5. Agradecimentos

Este trabalho foi realizado com apoio da Fundação de Amparo à Pesquisa do Estado de São Paulo (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 (Techno-Human Systems of the Future) do Centro Internacional de Pesquisa (IRC) 'Transitions' (CNRS, USP e FAPESP CIP #2025/01171-0).

## Referências

Brito, A. V., Costa, L. F. S., Bucher, H., Sander, O., Becker, J., Oliveira, H., and Melcher, E. U. (2015). A distributed simulation platform using hla for complex embedded systems design. In *Proceedings of the 19th International Symposium on Distributed Simulation and Real Time Applications*, pages 195–202, Chengdu, China. IEEE Press.

Drogoul, A. and Ferber, J. (2018). Multi-agent simulation as a tool for studying emergent processes in societies. In *Simulating societies*, pages 127–142. Routledge.

Lamport, L. (1978). Time, clocks, and the ordering of events in a distributed system. *Commun. ACM*, 21(7):558–565.

Lin, Z. and Yao, Y. (2015). Load balancing for parallel discrete event simulation of stochastic reaction and diffusion. In *2015 IEEE International Conference on Smart City/SocialCom/SustainCom (SmartCity)*, pages 609–614, Chengdu, China. IEEE.

Santana, E. F. Z. (2019). *InterSCSimulator: a scalable, open source, smart city simulator*. PhD thesis, Universidade de São Paulo, Instituto de Matemática e Estatística.

Santana, E. F. Z., Lago, N., Kon, F., and Milojicic, D. S. (2017). InterSCSimulator: Large-scale traffic simulation in smart cities using erlang. In *International Workshop on Multi-Agent Systems and Agent-Based Simulation*, pages 211–227. Springer.

Wang, H., Yan, H., Rong, C., Yuan, Y., Jiang, F., Han, Z., Sui, H., Jin, D., and Li, Y. (2024). Multi-scale simulation of complex systems: a perspective of integrating knowledge and data. *ACM Computing Surveys*, 56(12):1–38.