INSTITUTO POLITÉCNICO DE BRAGANÇA
Escola Superior de Tecnologia e Gestão

UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

Simulação de um Sistema de Fabrico Usando Modelação Baseada em Agente

Leonardo Beckhauser de Moraes

Dissertação para obtenção do grau de Mestre em:

Engenharia Eletrotécnica e de Computadores

Trabalho realizado sob a orientação de

Prof. Dr. José Fernando Lopes Barbosa

Prof. Dr. Claudio Leones Bazzi

Prof. Dr. Paulo Jorge Pinto Leitão

Bragança

2025

Simulação de um Sistema de Fabrico Usando Modelação Baseada em Agente

Leonardo Beckhauser de Moraes

Dissertação para obtenção do grau de Mestre em:

Engenharia Eletrotécnica e de Computadores

Trabalho realizado sob a orientação de

Prof. Dr. José Fernando Lopes Barbosa

Prof. Dr. Claudio Leones Bazzi

Prof. Dr. Paulo Jorge Pinto Leitão

Bragança

2025

Dedicatória

Este trabalho é dedicado à minha família e amigos.

Agradecimentos

Gostaria de expressar minha profunda gratidão à minha família, em especial à minha mãe, por todo o amor, dedicação e apoio incondicional ao longo da minha vida. Sou imensamente grato por tudo o que ela fez e faz por mim, sendo o alicerce que tornou possível minha trajetória acadêmica, desde a graduação até o mestrado. Aos meus amigos, que estiveram presentes nos momentos mais desafiadores, oferecendo força, companheirismo e incentivo quando mais precisei, deixo também o meu sincero agradecimento. Ao meu orientador, José Barbosa, agradeço pela orientação atenciosa, pela confiança no meu trabalho e pelo suporte constante durante o desenvolvimento desta pesquisa. A todos que contribuíram, de forma direta ou indireta, para a realização deste trabalho, o meu muito obrigado.

Resumo

A evolução tecnológica tem impulsionado mudanças significativas nos processos produtivos, elevando-os a patamares cada vez mais automatizados, assim o presente trabalho propõe o desenvolvimento e a avaliação de um modelo de simulação baseado em agentes para representar o funcionamento de um sistema ciber-físico de manufatura inteligente, com foco na descentralização da tomada de decisão e na auto-organização dos processos produtivos. O modelo foi implementado na plataforma NetLogo e utiliza uma lógica de alocação inspirada em aprendizado por reforço, baseada em colônias de formigas, para otimizar a escolha de máquinas de acordo com desempenho histórico e nível de ocupação. A modelagem considera agentes de produto e de recurso, com diferentes habilidades, tempos de processamento e *buffers* limitados, permitindo simular fluxos produtivos complexos e avaliar seu desempenho sob diferentes condições operacionais. Foram realizados testes com cinco cenários distintos, incluindo variações na taxa de chegada de produtos, tamanho dos buffers e ocorrência de falhas em máquinas. Os resultados demonstram que o sistema é capaz de se adaptar a restrições e mudanças no ambiente, promovendo alocação mais eficiente das tarefas e redução de gargalos. Conclui-se que a abordagem baseada em agentes, aliada a mecanismos simples de reforço, permite criar modelos flexíveis, escaláveis e alinhados aos princípios da Indústria 4.0. A plataforma NetLogo mostrou-se adequada para prototipagem e experimentação iterativa, permitindo futuras extensões do modelo para sistemas mais complexos.

Palavras-chave: Modelagem Baseada em Agentes, Indústria 4.0, Aprendizado por Reforço, NetLogo

Abstract

Technological advancements have driven significant changes in production processes, making them increasingly automated. This study proposes the development and evaluation of an agent-based simulation model to represent the operation of a cyber-physical system for smart manufacturing, focusing on decentralized decision-making and self-organization of production processes. The model was implemented on the NetLogo platform and employs a reinforcement learning-inspired allocation logic, based on ant colony behavior, to optimize machine selection according to historical performance and occupancy level. The modeling includes product and resource agents with varying skills, processing times, and limited buffers, enabling the simulation of complex production flows and the evaluation of system performance under different operational conditions. Tests were conducted across five distinct scenarios, including variations in product arrival rates, buffer sizes, and machine failures. The results show that the system can adapt to constraints and environmental changes, promoting more efficient task allocation and reducing bottlenecks. It is concluded that the agent-based approach, combined with simple reinforcement mechanisms, enables the creation of flexible and scalable models aligned with the principles of Industry 4.0. The NetLogo platform proved suitable for prototyping and iterative experimentation, allowing future extensions of the model to more complex systems.

Keywords: Agent-Based Modeling, Industry 4.0, Reinforcement Learning, NetLogo

Conteúdo

Lista de Tabelas

Lista de Figuras

Siglas

ABM Modelagem Baseada em Agentes *(Agent Based Modeling)*. 1, 2, 16, 17, 20, 23, 59

AI Inteligência Artificial *(Artificial Intelligence)*. 1, 14

CPS Sistemas Ciber Físico *(Cyber Physical Systems)*. 1, 8, 9, 11, 14

ERP Sistema de Planejamento de Recursos *(Enterprise Resource Planning)*. 9

IoT Internet das Coisas *(Internet of Things)*. 1, 8, 9, 11, 12

IPB Instituto Politécnico de Bragança. 2, 24

ML Aprendizado de Máquina *(Machine Learning)*. 14, 15

PA Agentes de Produto *(Product Agents)*. 28–32, 38, 43

RA Agentes de Recurso *(Resource Agents)*. 24, 29, 30, 32, 37, 39, 43

RFID Identificação por Radiofrequência *(Radio-Frequency Identification)*. 24, 26, 49

RL Aprendizado por Reforço *(Reinforcement Learning)*. 15, 16, 33, 34

SMA Sistema Multiagente *(Multi-Agent System)*. 13, 14

Capítulo 1

Introdução

Os avanços tecnológicos têm provocado profundas transformações nos sistemas produtivos, tornando-os cada vez mais automatizados, interconectados e inteligentes. Neste contexto, surge a Indústria 4.0 como um novo paradigma que integra tecnologias digitais, Sistemas Ciber Físico *(Cyber Physical Systems)* (CPS), Internet das Coisas *(Internet of Things)* (IoT) e Inteligência Artificial *(Artificial Intelligence)* (AI) aos processos de fabricação tradicionais [1]–[3].

A implementação dos chamados CPS permite a convergência entre os mundos físico e digital, viabilizando processos produtivos adaptativos, descentralizados e autônomos [4]–[6]. Esses sistemas, ao combinarem sensores, atuadores, redes de comunicação e algoritmos de controle, são capazes de monitorar o ambiente, tomar decisões em tempo real e reconfigurar suas operações de acordo com as condições operacionais [7], [8].

No âmbito da Indústria 4.0, torna-se necessário compreender como sistemas inteligentes e distribuídos operam sob diferentes condições. Uma abordagem promissora para esta finalidade é a Modelagem Baseada em Agentes *(Agent Based Modeling)* (ABM), que permite representar elementos individuais, como máquinas e produtos, como agentes autônomos capazes de tomar decisões e interagir entre si [9], [10]. Essa técnica se mostra particularmente útil em ambientes industriais dinâmicos, nos quais o controle centralizado pode se tornar ineficiente ou inviável [11].

Este trabalho propõe o desenvolvimento de um modelo de simulação baseado em agentes para representar o funcionamento de um sistema ciber-físico de manufatura em escala reduzida, localizado no Instituto Politécnico de Bragança (IPB). O modelo foi implementado utilizando a plataforma NetLogo, uma das ferramentas mais consolidadas para simulações ABM, por sua simplicidade, visualização interativa e suporte à experimentação iterativa [12]–[14].

A lógica do modelo considera a descentralização da tomada de decisão, o uso de *buffers* para controle de filas e um mecanismo de reforço inspirado em sistemas naturais, como colônias de formigas, promovendo a auto-organização e a otimização do sistema produtivo [15], [16].

Através da simulação de diferentes cenários busca-se analisar o comportamento emergente do sistema e propor estratégias de alocação que aumentem sua eficiência e resiliência. Incluindo variações na taxa de chegada de produtos, tamanhos de *buffers* e falhas operacionais

1.1 Problema de Pesquisa

Em vista das necessidades crescentes de tornar os sistemas de manufatura mais eficientes, flexíveis e resilientes tem impulsionado a adoção de ferramentas computacionais avançadas para apoiar a tomada de decisão em ambientes industriais. Nesse contexto, destaca-se a ABM, uma abordagem que permite simular o comportamento individual dos elementos do sistema, como máquinas e produtos, possibilitando a observação de padrões globais emergentes a partir de interações locais.

Com os avanços da automação e da integração digital nas linhas de produção, torna-se fundamental compreender como sistemas descentralizados operam em cenários complexos e sujeitos a imprevistos. A ABM se mostra especialmente eficaz nesse cenário, pois facilita a modelagem de sistemas distribuídos, adaptativos e autônomos, alinhando-se aos princípios da Indústria 4.0.

Este trabalho propõe o desenvolvimento de um modelo de simulação computacional

que represente o comportamento de um sistema de produção inteligente, simulando sua performance sob diferentes condições operacionais. A proposta original consiste em modelar os recursos produtivos, máquinas, como agentes autônomos, capazes de tomar decisões com base em sua disponibilidade e histórico de desempenho. O objetivo é analisar como essas decisões afetam o fluxo produtivo, a ocupação dos recursos, os tempos de ciclo dos produtos e a resiliência do sistema diante de falhas.

A plataforma escolhida para o desenvolvimento foi o NetLogo, por sua interface acessível, bibliotecas de apoio, suporte a visualizações em tempo real e experimentações iterativas. Com ela, é possível simular diferentes configurações, como variações na taxa de chegada de produtos, tamanhos de *buffers* e falhas operacionais, permitindo explorar estratégias de alocação e realocação de tarefas em um ambiente controlado. Essa abordagem também contribui para o desenvolvimento de ambientes virtuais de treinamento, nos quais operadores e engenheiros podem avaliar soluções sem comprometer a operação real da planta.

1.2 Objetivos

O objetivo geral deste trabalho é desenvolver e avaliar um modelo de simulação baseado em agentes para um sistema ciber-físico de manufatura, utilizando a plataforma NetLogo.

Os objetivos específicos incluem:

- Representar os recursos (máquinas) e produtos como agentes autônomos com regras de comportamento próprias;
- Simular o sistema sob diferentes condições operacionais, incluindo falhas e variações na taxa de chegada de produtos;
- Medir e analisar métricas como tempo médio de produção, ocupação dos recursos e eficiência global;
- Propor estratégias de alocação e realocação de tarefas para mitigar falhas e gargalos;

- Comparar o desempenho do sistema em cenários distintos, com base em dados extraídos de simulações repetidas por meio da ferramenta BehaviorSpace.

1.3 Estrutura do Documento

Este trabalho está estruturado em seis capítulos principais, organizados de forma a proporcionar uma compreensão progressiva dos conceitos, metodologias e resultados obtidos com a simulação de um sistema de produção baseado em agentes:

- **Capítulo 1:** Apresenta o contexto da pesquisa, destacando a relevância da modelagem baseada em agentes na Indústria 4.0 e nos sistemas ciber-físicos. Inclui também a justificativa do estudo, a formulação do problema, os objetivos do trabalho e a presente estrutura do documento.

- **Capítulo 2:** Aborda os fundamentos teóricos relacionados à Indústria 4.0, com foco em sistemas ciber-físicos (CPS), Internet das Coisas (IoT) e Sistemas Multiagentes. Além disso, discute as principais plataformas de simulação baseadas em agentes, com destaque para o NetLogo, ferramenta utilizada neste projeto.

- **Capítulo 3:** Detalha a metodologia adotada para o desenvolvimento do modelo de simulação. São definidos os tipos de agentes (produto e recurso), suas interações, dinâmicas e lógica de decisão baseada em reforço, inspirada em colônias de formigas. Também são descritos os elementos do estudo de caso, como as máquinas modeladas e os cenários de simulação.

- **Capítulo 4:** Apresenta o processo de construção do modelo na plataforma NetLogo, incluindo a codificação dos comportamentos dos agentes, a implementação de falhas, o mecanismo de reforço e evaporação, além da configuração da interface gráfica e das métricas monitoradas durante as simulações.

- **Capítulo 5:** Analisa os resultados obtidos em diferentes cenários de simulação, com variações na taxa de chegada de produtos, tamanhos de *buffers* e ocorrência de falhas operacionais. As análises buscam avaliar a eficiência do sistema e sua capacidade de adaptação a condições adversas.

- **Capítulo 6:** Apresenta as conclusões do trabalho, destacando as contribuições do modelo proposto, suas limitações e sugestões para trabalhos futuros, incluindo possíveis melhorias na lógica de reforço e expansão da simulação para contextos mais complexos.

Capítulo 2

Estado da Arte

O presente capítulo apresenta os conceitos fundamentais de sistemas ciberfísicos e sistemas multiagentes, para a compreensão do trabalho. Em seguida, é aprofundada a discussão sobre a metodologia de simulação, com destaque para a abordagem baseada em agentes e a aplicação de *machine learning* na modelagem e otimização de comportamentos dinâmicos. Por fim, são apresentadas e comparadas diversas plataformas de simulação, sendo o NetLogo a ferramenta selecionada para a implementação das simulações do sistema fabril proposto neste estudo, integrando técnicas de *machine learning* para análise e tomada de decisão autônoma.

2.1 Industria 4.0

Conforme destacado por Lasi, Fettke, Kemper et al. [1], a indústria representa um dos setores fundamentais da economia, sendo responsável pela produção de bens materiais que, ao longo do tempo, passaram por processos de mecanização e automação. As evoluções tecnológicas trouxeram transformações significativas nos métodos de fabricação, caracterizando três grandes revoluções industriais: a primeira, impulsionada pela mecanização no século XVIII; a segunda, marcada pela utilização intensiva da eletricidade no século XIX; e a terceira, que consolidou a digitalização dos sistemas produtivos no século XX.

Os mesmos autores apontam que a interconectividade entre tecnologias digitais, a

Internet e dispositivos inteligentes está remodelando as fábricas e promovendo um novo paradigma produtivo, denominado Indústria 4.0. Esse conceito baseia-se na criação de sistemas de manufatura modulares, nos quais os produtos possuem capacidade de autogerenciamento ao longo do processo fabril, possibilitando produção personalizada em pequena escala, sem comprometer a eficiência típica da manufatura em massa [1].

Com o avanço tecnológico e o aumento da complexidade dos sistemas industriais, tornou-se essencial buscar soluções inteligentes e adaptativas. Nesse contexto, Os CPS emergem como uma alternativa viável, integrando componentes computacionais e físicos para formar redes capazes de monitorar, analisar e controlar processos produtivos em tempo real. A convergência entre *software*, *hardware* e redes de comunicação facilita uma interação fluida entre o ambiente físico e digital, promovendo decisões autônomas e otimizando a execução das tarefas [16] [15].

Com a ampliação do uso de CPS e a integração de cada vez mais dispositivos conectados pela IoT, a quantidade de informações processadas instantaneamente atinge níveis muito altos. Diante disso, a estrutura centralizada convencional se revela ineficiente para enfrentar os desafios da indústria moderna, demandando soluções distribuídas e inteligentes para o tratamento e gerenciamento de dados [7] [8].

2.1.1 Sistemas ciberfisicos (CPS)

Os CPS representam uma convergência sofisticada entre *hardware* e *software*, estabelecendo uma comunicação bidirecional constante entre ambientes físicos e virtuais. Essas plataformas integram dispositivos sensoriais, mecanismos de atuação, lógicas de controle inteligente e infraestruturas de rede para observar, processar e regular operações industriais com mínima latência. Sua competência para executar escolhas autônomas e adaptar-se a alterações contextuais os posiciona como elementos cruciais para sistemas automatizados inteligentes, particularmente no âmbito da Quarta Revolução Industrial. Cada unidade física, como equipamentos industriais, possui seu equivalente digital que registra dados operacionais e de estado, permitindo uma administração produtiva mais eficaz [2].

A conexão entre CPS e IoT apresenta forte interdependência, embora com características distintas. A IoT prioriza a interligação de dispositivos e aquisição de dados, enquanto os CPS implementam ciclos completos de retroalimentação que traduzem processamento digital em intervenções físicas. Num cenário fabril, por exemplo, redes IoT podem capturar métricas como temperatura e oscilações mecânicas, ao passo que os CPS processam essas informações para autorregular parâmetros de funcionamento ou ativar mecanismos de proteção, assegurando desempenho ótimo e prevenção de avarias [17].

No paradigma da Indústria 4.0, os CPS atuam como habilitadores essenciais para operações fabris com maior adaptabilidade, autonomia e rendimento. Eles possibilitam que sistemas produtivos se reajustem automaticamente conforme demandas variáveis, diminuindo significativamente a dependência de supervisão humana. Ademais, promovem o uso otimizado de recursos através de técnicas analíticas capazes de detectar anomalias e prever falhas, reduzindo assim custos de manutenção e períodos improdutivos. Sua natureza integradora ainda possibilita a combinação harmoniosa de tecnologias diversas, incluindo sistemas robóticos, plataformas Sistema de Planejamento de Recursos *(Enterprise Resource Planning)* (ERP) e infraestruturas em nuvem, formando ambientes industriais altamente conectados e inteligentes [18].

A arquitetura 5C, proposta por Lee, Bagheri e Kao [4], é uma das abordagens mais consolidadas para estruturar CPS no contexto da Indústria 4.0. Essa arquitetura organiza as funcionalidades de um CPS em cinco níveis hierárquicos: Conexão (*Connection*), Conversão (*Conversion*), Cibernético (*Cyber*), Cognição (*Cognition*) e Configuração (*Configuration*). Cada nível representa uma etapa de processamento e decisão que transforma dados brutos em ações inteligentes e autônomas.

A Figura 2.1 ilustra a pirâmide da arquitetura 5C, usada como referência conceitual para o modelo de simulação.

Figura 2.1: Arquitetura 5C para implementação de Sistemas Ciber-Físicos.

Fonte: Adaptado de Lee, Bagheri e Kao [4]

A arquitetura 5C, conforme apresentada por Trappey, Trappey, Govindarajan et al. [19], organiza-se em cinco níveis funcionais que estruturam a integração entre componentes físicos e digitais nos Sistemas Ciber-Físicos (CPS):

O **Nível I – Conexão** marca o ponto inicial da arquitetura, no qual sensores, atuadores e dispositivos de controle são integrados por meio de redes de comunicação. Essa etapa possibilita a coleta de dados diretamente do ambiente físico. Com os avanços tecnológicos, muitos sensores atuais já incorporam capacidades de processamento e conectividade, inclusive via redes sem fio, facilitando a integração direta com a Internet.

No **Nível II – Conversão**, os dados adquiridos são transformados em informações úteis por meio de processamento e análise. Essa etapa permite realizar inferências e ajustes operacionais e lida com grandes volumes de dados (*big data*), exigindo infraestrutura adequada de armazenamento e computação, como os serviços baseados em nuvem.

O **Nível III – Cibernético** envolve o uso de algoritmos e estruturas computacionais para monitorar o estado do sistema e antecipar comportamentos futuros. Esse nível inclui aspectos de controle lógico, segurança e integração dos processos computacionais.

No **Nível IV – Cognição**, o foco está no suporte à tomada de decisão. A partir do

conhecimento gerado nas etapas anteriores, este nível oferece diagnósticos e recomendações, estando diretamente associado ao planejamento da produção, manutenção preditiva e gestão operacional.

Finalmente, o **Nível V – Configuração** traduz a inteligência acumulada em ações efetivas no ambiente físico. Esse nível contempla capacidades de aprendizado, autoajuste e adaptação, sendo essencial para a implementação de inteligência artificial na Indústria 4.0, com foco na autonomia e eficiência dos sistemas [19].

Apesar das vantagens associadas à adoção de CPS, diversos obstáculos ainda precisam ser superados para garantir sua implementação eficaz. A crescente conectividade desses sistemas amplia os riscos relacionados à segurança cibernética, tornando as infraestruturas mais vulneráveis a ataques. Outro desafio significativo é a necessidade de integrar dispositivos com diferentes protocolos e características técnicas, o que demanda o uso de padrões abertos e arquiteturas modulares e escaláveis. Além disso, aplicações industriais impõem requisitos rigorosos, como alta confiabilidade e baixa latência, exigindo soluções robustas de processamento em tempo real [20].

A habilidade dos CPS de unir o mundo físico ao digital oferece oportunidades para melhorar a produtividade, permitir personalizações em larga escala e otimizar o uso de energia. No entanto, para que tais benefícios sejam plenamente realizados, é essencial enfrentar questões técnicas críticas, sobretudo no que diz respeito à segurança, interoperabilidade e desempenho em tempo real, assegurando que esses sistemas sejam, ao mesmo tempo, eficientes, seguros e confiáveis [21].

2.1.2 Internet das Coisas (IoT)

De acordo com Saha, Saha, Ghosh et al. [22], a IoT constitui uma rede sofisticada que permite a comunicação e o intercâmbio de dados entre dispositivos inteligentes, máquinas e objetos físicos. Essa conectividade é essencial para o desenvolvimento de sistemas autônomos e inteligentes, capazes de reagir de maneira adaptativa às condições do ambiente.

No setor industrial, a IoT desempenha um papel fundamental ao promover a integração digital entre equipamentos de produção, sensores e sistemas de controle. Conforme apontado por Atzori, Iera e Morabito [23], essa estrutura descentralizada proporciona três funcionalidades principais: monitoramento constante do desempenho operacional, processamento em tempo real de grandes quantidades de dados e decisões automatizadas com base em análises locais. Tais capacidades são especialmente relevantes em cenários de manufatura, onde a aplicação da IoT permite:

- **Maior Visibilidade das Operações**: A coleta contínua de dados por sensores embarcados em equipamentos, como informações sobre vibração, temperatura e consumo de energia e viabiliza a criação de réplicas digitais (*digital twins*) dos processos produtivos [3].
- **Manutenção Antecipada**: O uso de algoritmos de aprendizado de máquina possibilita a análise de fluxos de dados em tempo real para prever falhas e ajustar os planos de manutenção de forma proativa [24].
- **Ajustes Dinâmicos em Tempo Real**: Sistemas ciber-físicos integrados são capazes de modificar automaticamente os parâmetros de produção conforme mudanças na demanda ou nas condições do ambiente [6].

Apesar de seus benefícios, a adoção da IoT na indústria enfrenta barreiras importantes. Conforme destacado por Lee [5], os principais desafios envolvem: a integração eficiente entre dispositivos com diferentes padrões de comunicação, a proteção das redes contra ameaças cibernéticas e o gerenciamento adequado do elevado volume de dados gerados continuamente.

2.2 Sistemas Multiagentes

À medida que os sistemas industriais se tornam mais dinâmicos e complexos, cresce a necessidade por soluções que ofereçam maior flexibilidade e capacidade de adaptação. Os

Sistema Multiagente *(Multi-Agent System)* (SMA) apresentam-se como uma alternativa viável, pois permitem que múltiplos agentes autônomos operem de forma distribuída, cooperando para gerenciar processos e responder de maneira eficiente a mudanças nos ambientes produtivos [9], [11]. Neste estudo, utilizou-se um SMA para modelar um sistema de manufatura flexível, em que cada estação de trabalho é representada por um agente capaz de tomar decisões de forma autônoma e se comunicar com os demais agentes para coordenar suas ações. A modelagem permitiu testar diversos cenários e a identificação de melhorias, evidenciando os benefícios dos SMA na otimização da produção e na redução de custos operacionais.

O termo *agente* pode assumir significados distintos conforme o contexto e a área de aplicação, como inteligência artificial ou ciência da computação. Conforme definido por Wooldridge [25], um agente é um sistema computacional com autonomia, capaz de interagir com seu ambiente por meio de sensores e atuadores, atuando de forma deliberada para alcançar objetivos específicos. Essa capacidade de agir com independência e orientação a metas o diferencia de outras formas de sistemas computacionais.

De acordo com Wooldridge e Jennings [10], os agentes geralmente apresentam quatro características fundamentais:

- **Autonomia** - O agente opera sem intervenção humana direta ou de outros sistemas, controlando suas próprias ações e estado interno de maneira independente.
- **Capacidade de interação social** - São aptos a comunicar-se com outros agentes por meio de linguagens apropriadas, colaborando quando necessário para resolver tarefas complexas.
- **Reatividade** - Monitoram o ambiente ao seu redor e reagem de forma eficaz a mudanças contextuais.
- **Pró-atividade** - Não apenas respondem a estímulos externos, mas também são capazes de agir com base em objetivos internos, antecipando ações necessárias para atingir suas metas.

No desenvolvimento de sistemas inteligentes para a indústria, uma questão relevante diz respeito ao modelo de controle adotado. Abordagens centralizadas concentram a tomada de decisão em um único ponto, o que pode gerar gargalos, dificultar a escalabilidade e comprometer a adaptabilidade em situações imprevistas. Em contrapartida, o controle distribuído, como o implementado por meio de SMA, descentraliza as decisões, permitindo que cada agente atue com base em informações locais e em regras de cooperação com os demais. Esse paradigma favorece sistemas mais resilientes, escaláveis e adaptáveis, alinhando-se aos princípios da Indústria 4.0 e às arquiteturas de CPS [10].

2.2.1 Machine learning

Aprendizado de Máquina *(Machine Learning)* (ML) é um subcampo da AI que capacita sistemas computacionais a aprenderem padrões a partir de dados, sem serem explicitamente programados. Em seu trabalho, Saraswat e Raj [26] abordam a definição de ML e seus três principais modelos de aprendizado: supervisionado, não supervisionado e por reforço. O ML permite que sistemas computacionais melhorem seu desempenho por meio da experiência, analisando grandes volumes de dados estruturados ou não estruturados, como textos, imagens e sinais de sensores. Essa capacidade de aprendizado automático tem impulsionado avanços em diversas áreas, desde diagnósticos médicos até a otimização de processos industriais. [26]

O aprendizado supervisionado é uma das técnicas mais comuns em ML. Nesse método, os algoritmos aprendem a partir de dados rotulados, nos quais cada exemplo de entrada está associado a um resultado conhecido. Assim, o modelo desenvolve a capacidade de relacionar entradas a saídas, tornando-se apto a realizar previsões ou classificações quando aplicado a novos dados. [26]. Técnicas como árvores de decisão e redes neurais são comumente aplicadas em problemas como detecção de fraudes, filtragem de spam e diagnósticos médicos. Por exemplo, um modelo de árvore de decisão pode ser usado para classificar um e-mail é legítimo ou spam com base em características como o conteúdo e o remetente [27].

Diferentemente da aprendizagem supervisionada, a abordagem não supervisionada opera sem a necessidade de dados previamente rotulados, baseando-se exclusivamente na análise das características intrínsecas dos dados para identificar padrões e estruturas subjacentes. Esse método se destaca por sua capacidade de extrair conhecimento de forma autônoma, sem a dependência de anotações manuais, o que elimina não apenas o custo associado à rotulagem, mas também potenciais vieses introduzidos durante esse processo [28]. No entanto, a ausência de rótulos pré-definidos representa um desafio significativo: a avaliação da qualidade do aprendizado torna-se complexa, já que não há referências explícitas para validar os resultados. Apesar dessa limitação, a aprendizagem não supervisionada desempenha um papel fundamental em diversas aplicações, especialmente na análise exploratória de dados multidimensionais [29].

O Aprendizado por Reforço *(Reinforcement Learning)* (RL) constitui um paradigma fundamental no âmbito da aprendizagem automática ML, caracterizando-se por um processo de tomada de decisão sequencial em que um agente inteligente interage com um ambiente dinâmico, o objetivo central do RL é determinar a sequência de ações mais eficazes para otimizar um resultado desejado em um contexto particular. Diversos *softwares* e sistemas computacionais empregam essa técnica para identificar a melhor estratégia ou direção a seguir diante de uma dada situação [26]. Uma distinção fundamental entre a aprendizagem por reforço e a aprendizagem supervisionada reside na natureza dos dados de treinamento. No RL, o agente desenvolve seu conhecimento por meio da interação contínua com o ambiente, obtendo *feedback* positivo (recompensas) ou negativo (penalidades) conforme executa ações. Ao contrário do aprendizado supervisionado, que utiliza dados previamente rotulados com entradas e saídas corretas, o RL não parte de exemplos pré-definidos, mas sim de tentativa e erro, ajustando seu comportamento com base nas consequências de suas decisões. O agente, portanto, deve descobrir por conta própria a melhor forma de atingir o objetivo, aprendendo com a resposta do ambiente. Essa capacidade de aprender sem a necessidade de dados de treinamento explicitamente rotulados é uma característica distintiva da RL [26].

Embora o presente trabalho não implemente algoritmos formais de aprendizagem supervisionada ou não supervisionada, a lógica de decisão dos agentes incorpora um comportamento inspirado em RL. No modelo proposto, essa ideia é representada por meio de um vetor de `reforço` associado a cada recurso (máquina), o qual é atualizado positivamente após execuções bem-sucedidas de tarefas, e gradualmente decresce com o tempo, simulando um mecanismo de esquecimento. Esse reforço influencia a seleção das máquinas pelos agentes de produto, de forma que aquelas com melhor desempenho passado tornam-se preferidas para novas operações. Essa abordagem representa uma simplificação prática do RL, capturando seus princípios fundamentais sem necessidade de técnicas complexas ou aprendizado supervisionado formal. Com isso, é possível observar um comportamento adaptativo emergente, característico de sistemas auto-organizados.

2.3 Plataformas de Simulação

O crescimento da ABM deve-se, em grande parte, ao desenvolvimento de plataformas especializadas que, nos últimos anos, simplificaram e tornaram mais acessível esta abordagem para diversas áreas de pesquisa. Contudo, é importante destacar que existem diferenças significativas no suporte e recursos oferecidos por cada uma dessas ferramentas. A seguir, são apresentados os principais simuladores de ABM: AnyLogic, Repast e NetLogo [13].

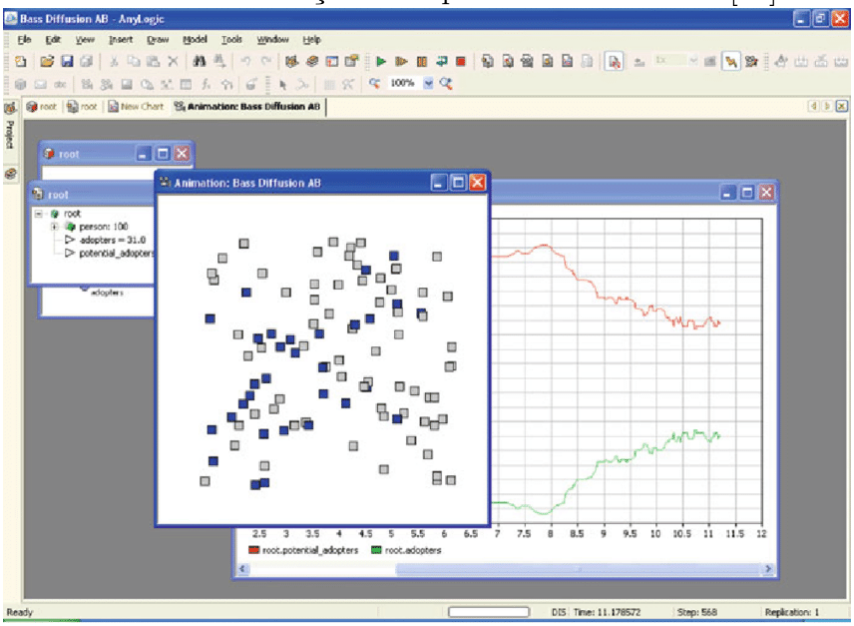
2.3.1 AnyLogic

O AnyLogic 6.5 é um *software* voltado à modelagem e simulação, que se destaca por integrar, em uma única plataforma, diferentes abordagens metodológicas, como a ABM, a simulação de eventos discretos e a dinâmica de sistemas [12]. Essa integração permite a criação de modelos complexos onde agentes interagem e influenciam o sistema como um todo. O *software* oferece funcionalidades avançadas como a leitura e escrita dinâmica de dados em arquivos e bancos de dados, além da geração de gráficos em tempo real durante a simulação. Adicionalmente, possibilita a comunicação com outros programas

externos para troca de informações. Embora o desenvolvimento de modelos seja restrito ao sistema operacional Microsoft Windows, as simulações compiladas podem ser executadas em qualquer plataforma com suporte Java. A ampla variedade de casos apresentados no site oficial da AnyLogic evidencia sua aplicação em múltiplas áreas, abrangendo desde estudos em sistemas sociais e ambientais até simulações voltadas ao planejamento de serviços e ao comportamento de fluxos humanos. Apesar da disponibilidade de tutoriais em vídeo, o código fonte e a documentação detalhada desses exemplos não são acessíveis ao público [12]

Em sua pesquisa Clarke [30] utilizou o ambiente AnyLogic como ferramenta principal para desenvolver seus modelos de simulação ABM. Um exemplo da utilização em seus experimentos é visto na Figura 2.2.

Figura 2.2: Interface de simulação do experimento de Clarke [30] no AnyLogic



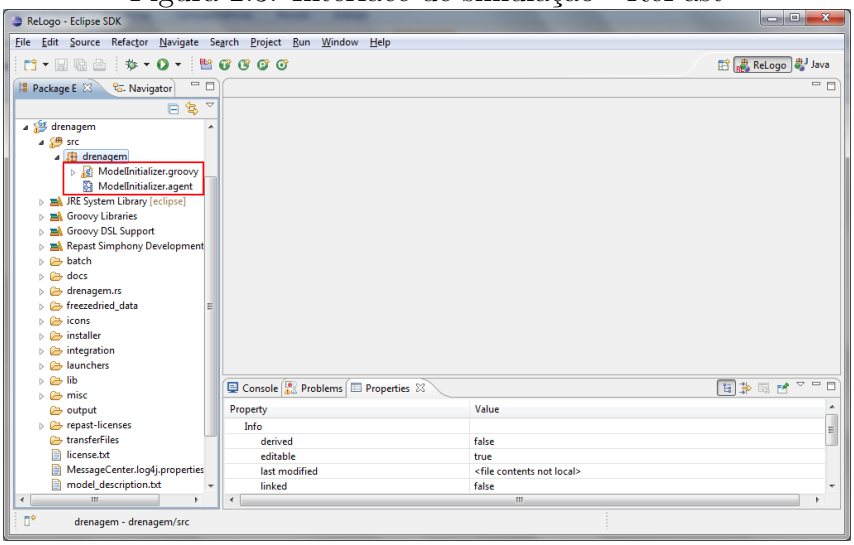
Fonte: Clarke [30]

2.3.2 RePast

O RePast foi criado no Laboratório de Computação para Pesquisa em Ciências Sociais da Universidade de Chicago com o propósito específico de desenvolver simulações baseadas em agentes para as ciências sociais[12]. Além disso, a plataforma foi projetada para atender a um domínio particular, especialmente as ciências sociais, incorporando ferramentas especializadas para essa área.

A Figura 2.3 exemplifica a interface de simulação do RePast, integrada ao ambiente de desenvolvimento Eclipse IDE.

Figura 2.3: Interface de simulação - RePast



Fonte: Adaptado do Allan [12]

Outro propósito do RePast consiste em tornar o desenvolvimento de modelos acessível a usuários com pouca ou nenhuma experiência prévia. Para isso, foram implementadas diversas estratégias, como a disponibilização de um modelo integrado simplificado e a utilização de interfaces interativas, que permitem a criação de modelos por meio de menus e scripts em Python [12]. O RePast oferece suporte à construção de modelos computacionais utilizando diferentes linguagens de programação, como Java, C#, C++, Groovy, ReLogo e Python, oferecendo flexibilidade para adaptação a diferentes contextos e preferências de desenvolvimento [14]

2.3.3 Netlogo

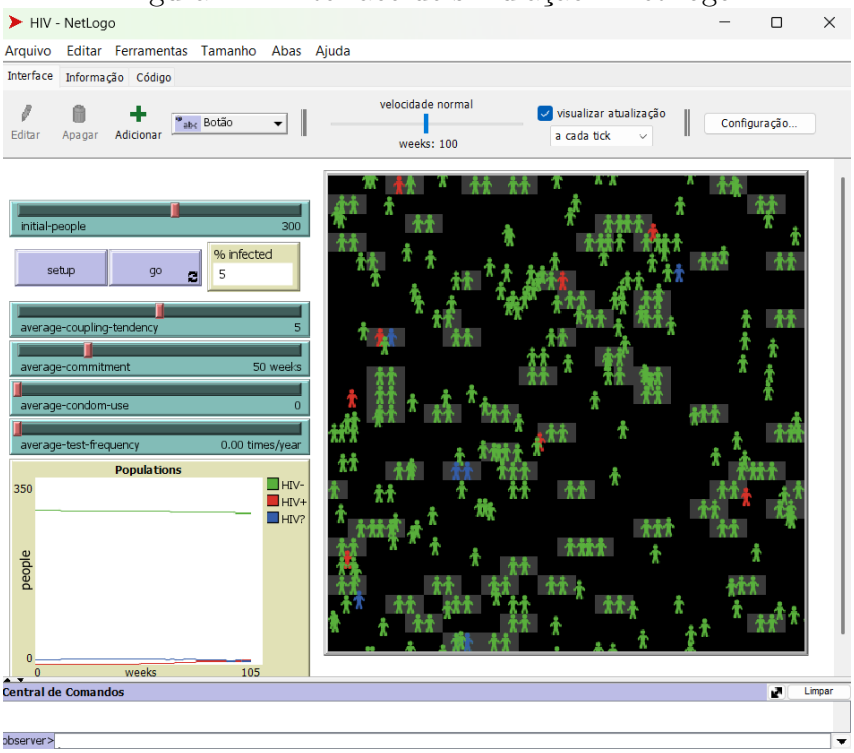
Desenvolvido inicialmente como StarLogoT, o NetLogo é um ambiente computacional de alto nível que combina uma linguagem de programação acessível e eficiente, com ferramentas gráficas integradas e ampla documentação. Sua estrutura é especialmente projetada para a simulação de sistemas complexos dinâmicos, permitindo a implementação de modelos em ambientes web. Um dos recursos mais importantes da plataforma é sua capacidade para coordenar a execução simultânea de centenas ou até milhares de agentes autônomos, o que viabiliza a análise das relações entre comportamentos individuais no nível micro e no nível macro que surgem da interação de muitos indivíduos [12].

A facilidade de uso, que era o objetivo primordial em seu *design*, demonstra a clara influência do StarLogo no NetLogo enquanto ferramenta pedagógica. Sua linguagem de programação incorpora diversas estruturas e elementos fundamentais de alto nível, o que minimiza significativamente o trabalho de programação. Derivada da linguagem Logo, ela integra várias funções de organização e controle, mas não possui todos os elementos presentes em linguagens de programação convencionais [12].

O NetLogo é amplamente reconhecido por sua interface bem elaborada e pela qualidade de sua documentação, destacando-se entre as plataformas de modelagem baseadas em agentes. A ferramenta oferece amplo suporte ao usuário, com manuais detalhados, tutoriais práticos e uma rica coleção de modelos prontos para uso e personalização. Esses modelos abrangem diversas áreas das ciências naturais e humanas, como biologia, medicina, física, química, matemática, computação, economia e psicologia social [12].

Na Figura 2.4, observa-se a interface do NetLogo, exibindo um modelo de simulação epidemiológica para casos de HIV. Este modelo específico está disponível na biblioteca padrão do *software*, na seção dedicada a aplicações biológicas.

Figura 2.4: Interface de simulação - NetLogo



Fonte: Adaptado de Allan [12]

O NetLogo é uma ferramenta amplamente adotada para simulações ABM, oferecendo uma curva de aprendizado acessível, ambiente visual interativo e excelente suporte para a experimentos de comportamentos descentralizados e dinâmicos [31].

Ao contrário de outras plataformas mais especializadas em modelagem de eventos discretos (como o Simulink) ou focadas em aplicações industriais avançadas (como o AnyLogic), o NetLogo permite rápida prototipagem, personalização da interface gráfica e controle direto sobre as regras de comportamento dos agentes [32]. Além disso, sua biblioteca nativa oferece modelos clássicos como o *Ants*, que inspirou a lógica de reforço adotada neste trabalho, e funcionalidades úteis como o *BehaviorSpace*, que automatiza testes em larga escala.

Portanto, considerando a natureza exploratória do projeto, a necessidade de testar múltiplos cenários e a relevância do comportamento emergente dos agentes, O NetLogo

se destaca por equilibrar de forma ideal simplicidade, flexibilidade e poder de representação. Essa escolha reforça o alinhamento metodológico entre a proposta da simulação e a ferramenta utilizada [33].

Capítulo 3

Metodologia

Este capítulo apresenta a abordagem metodológica utilizada no desenvolvimento do modelo de simulação. Primeiramente, são introduzidos os conceitos fundamentais da ABM, destacando sua adequação para representar sistemas produtivos descentralizados e dinâmicos. Em seguida, descreve-se a arquitetura geral do sistema proposto, com foco na caracterização dos agentes de produto e de recurso, suas interações e regras de comportamento. Também são apresentados os algoritmos que regem a lógica de alocação e execução de tarefas, bem como os parâmetros utilizados para compor os diferentes cenários de simulação que serão testados nos capítulos seguintes.

3.1 Modelagem Orientada a Agentes

No contexto deste trabalho, a ABM foi utilizada para modelar um sistema de manufatura em pequena escala, onde os produtos e os recursos do sistema são representados por agentes distintos. O objetivo é simular o comportamento do sistema sob diferentes condições operacionais, avaliar seu desempenho e observar a emergência de padrões de organização, tais como gargalos, filas e realocação de tarefas.

A escolha por essa abordagem se justifica pelas seguintes vantagens:

- **Descentralização do controle**: permite simular sistemas distribuídos sem necessidade de um agente centralizador;

- **Flexibilidade e escalabilidade**: facilita a adição de novos agentes ou regras com mínima reconfiguração do modelo;
- **Capacidade de lidar com comportamentos não lineares**: como falhas, congestionamentos e mudanças de prioridade;
- **Aderência aos princípios da Indústria 4.0**: modela estruturas inteligentes e auto-organizáveis, compatíveis com os conceitos de sistemas ciber-físicos.

A plataforma escolhida para desenvolvimento do modelo foi o *NetLogo*, por sua facilidade de uso, suporte visual nativo e ampla documentação. A ferramenta permite programar e executar simulações interativas com múltiplos agentes, além de exportar resultados estatísticos para análises mais aprofundadas. Seu ambiente gráfico favorece a experimentação iterativa, onde o usuário pode ajustar parâmetros como tempo de chegada, número de máquinas, capacidade dos *buffers* e probabilidade de falha, observando os impactos imediatos na produção.

3.2 Caracterização das Máquinas

A caracterização das máquinas no modelo segue diretamente a definição dos agentes. Cada máquina é representada como um agente autônomo com habilidades específicas, capacidade de processamento, *buffer* e lógica interna para controle de estado. Essa abordagem permite simular, de forma precisa, o comportamento individual dos recursos produtivos presentes em uma célula de manufatura real, mantendo coerência com os princípios da modelagem baseada em agentes.

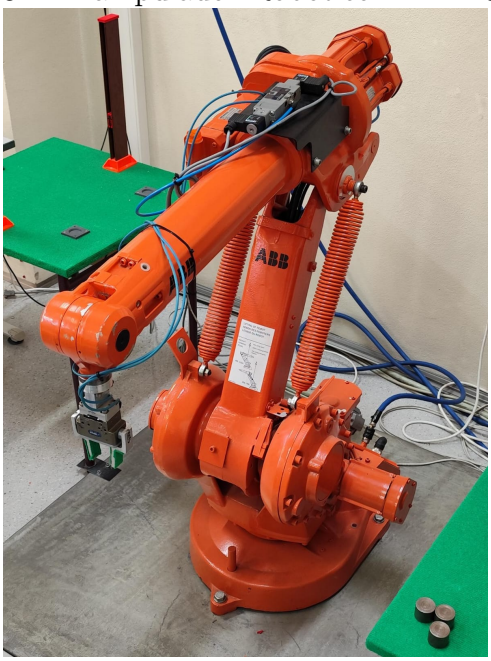
A estrutura adotada busca refletir as características de um sistema físico instalado no Laboratório da Escola Superior de Tecnologia e Gestão do IPB. A célula utilizada como base do estudo contém equipamentos diversos, como manipuladores robóticos, máquinas de perfuração e leitor de Identificação por Radiofrequência *(Radio-Frequency Identification)* (RFID), que foram abstraídos no modelo computacional como Agentes de Recurso *(Resource Agents)* (RA) com funções bem definidas e parâmetros ajustáveis..

A primeira máquina do sistema é o manipulador robótico (ABB IRB 1400), que desempenha um papel fundamental na logística interna da linha de produção, garantindo a transferência eficiente de itens entre as estações de trabalho. Este agente de recurso possui a habilidade transfer, sendo responsável por deslocar os produtos entre diferentes máquinas de maneira precisa e sincronizada. Com um tempo de operação constante, seu funcionamento é essencial para assegurar um fluxo produtivo contínuo, garantindo uma execução otimizada das tarefas.

O ABB IRB 1400 é um robô industrial amplamente utilizado em aplicações de manipulação, sendo conhecido por sua alta precisão, repetibilidade e confiabilidade em operações de transporte de materiais. Ele integra uma estrutura robusta e um sistema de controle eficiente, permitindo ajustes inteligentes e adaptação dinâmica às demandas da produção.

A Figura 3.1 ilustra o ABB IRB 1400 utilizado no sistema para executar a tarefa de transferência de produtos entre as estações de trabalho.

Figura 3.1: Manipulador Robótico ABB IRB 1400



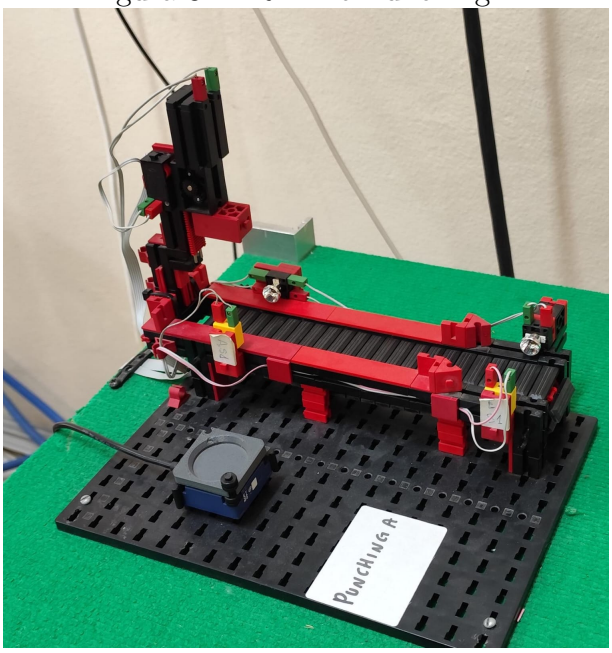
Fonte: Elaboração Própria.

A segunda máquina do sistema é o leitor RFID, responsável por identificar o tipo de produto logo na entrada do processo produtivo. Essa máquina realiza a leitura das informações de cada item e determina qual sequência de operações deverá ser seguida. No modelo, essa função é representada pela habilidade leitura e possui um tempo de execução curto, garantindo agilidade no início do ciclo de produção.

O processamento principal dos produtos ocorre nas máquinas *Punching machine A* e *Punching machine B*, ambas habilitadas para realizar a operação de puncionamento. A *Machine A* apresenta um tempo de execução mais eficiente (5 unidades de tempo), enquanto a *Machine B* executa a mesma tarefa em 7 unidades de tempo. A presença de duas máquinas com a mesma habilidade permite que o sistema se torne mais flexível, permitindo que os agentes de produto escolham dinamicamente a máquina mais vantajosa, com base em critérios como disponibilidade e desempenho passado.

A Figura 3.2 apresenta uma representação do leitor RFID e da Punching Machine A.

Figura 3.2: RFID e Punching A

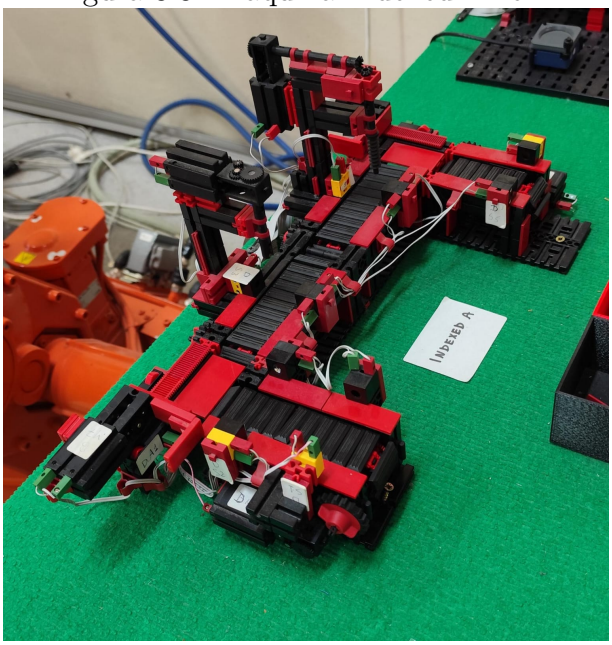


Fonte: Elaboração Própria.

As operações de perfuração são realizadas pelas máquinas Indexed Line A e Indexed Line B, consideradas mais complexas por possuírem duas habilidades distintas. A Indexed Line A executa a primeira operação em 7 unidades de tempo e a segunda em 6, enquanto a Indexed Line B realiza as mesmas tarefas em 5 e 9 unidades de tempo, respectivamente. Essas diferenças nos tempos de execução influenciam diretamente a tomada de decisão dos agentes de produto, que tendem a priorizar máquinas com menor carga ou com melhor histórico de desempenho, conforme determinado pela lógica de reforço implementada no modelo.

A Figura 3.3 ilustra a Indexed Line A, responsável pelas tarefas de perfuração descritas anteriormente.

Figura 3.3: Maquina Indexed Line A



Fonte: Elaboração Própria.

Por fim, após todas as operações de manufatura, os produtos são encaminhados ao inspetor, que realiza a inspeção final de qualidade. Esse agente possui a habilidade de inspeção e um tempo fixo de execução de 3 unidades de tempo. A inspeção representa a etapa final do processo produtivo e assegura que cada item tenha passado por todas as etapas previstas no plano de produção, garantindo a conformidade dos produtos com os requisitos estabelecidos.

A Tabela 3.1 apresenta os agentes modelados, suas habilidades e respectivos tempos de execução (em *ticks*).

Tabela 3.1: Representação dos recursos, habilidades e tempo

Resource	**(Skill, time)**
Manipulator robot	{transfer, 3}
Punching machine A	{punch_1, 5}
Punching machine B	{punch_1, 7}
Indexed line A	{drill_1, 7}, {drill_2, 6}
Indexed line B	{drill_1, 5}, {drill_2, 9}
RFID reader	{read, 2}
Inspector	{inspection, 3}

3.2.1 Definição dos Agentes

O modelo desenvolvido considera dois tipos principais de agentes, cada um com papéis distintos e complementares no funcionamento do sistema:

- **Agentes de Produto (Product Agents) (PA)**: representam os produtos que percorrem a linha de produção.
- **Agentes de Recurso (Resource Agents) (RA)**: representam os recursos produtivos, como máquinas, estações de trabalho e robôs.

Agentes de Produto (PA)

Os Agentes de Produto *(Product Agents)* (PA) atuam como entidades inteligentes responsáveis por coordenar o ciclo de vida completo dos itens em produção. Cada PA é

associado a um tipo de produto (*star* ou *triangle*) e carrega consigo um *plano de processo* (`process-plan`), que define a sequência de tarefas a ser executada.

Durante a simulação, os PA realizam as seguintes ações:

- Solicitam recursos com base na tarefa atual do seu `process-plan`;
- Coordenam a execução das tarefas conforme a ordem definida;
- Aguardam a finalização da tarefa antes de prosseguir;
- Atualizam métricas de desempenho, como tempo de processamento e desvios operacionais;
- Adaptam-se dinamicamente às mudanças no ambiente, como falhas ou sobrecarga dos recursos.

A Tabela 3.2 apresenta os planos de processo definidos para os produtos *star* e *triangle*:

Tabela 3.2: Sequência de Operações para os produtos Star e Triangle

Sequência	**Star**	**Triangle**
#1	read	read
#2	punch_1	drill_1
#3	drill_1	drill_2
#4	drill_2	punch_1
#5	inspection	inspection

Agentes de Recurso (RA)

Os RA modelam os equipamentos do sistema produtivo. Cada RA possui:

- Um conjunto de habilidades (`skills`) que definem as operações que pode realizar (por exemplo, `punch_1`, `drill_2`, `transfer`);
- Um estado operacional, controlado por variáveis como `available` (ligado/desligado) e `busy` (ocupado/livre);

- Um *buffer* com capacidade limitada, onde os PAs aguardam sua vez para serem processados;
- Tempos de execução específicos para cada operação, definidos conforme a máquina simulada;
- Um vetor de reforço (`reinforcement`), que representa o desempenho histórico da máquina em cada habilidade.

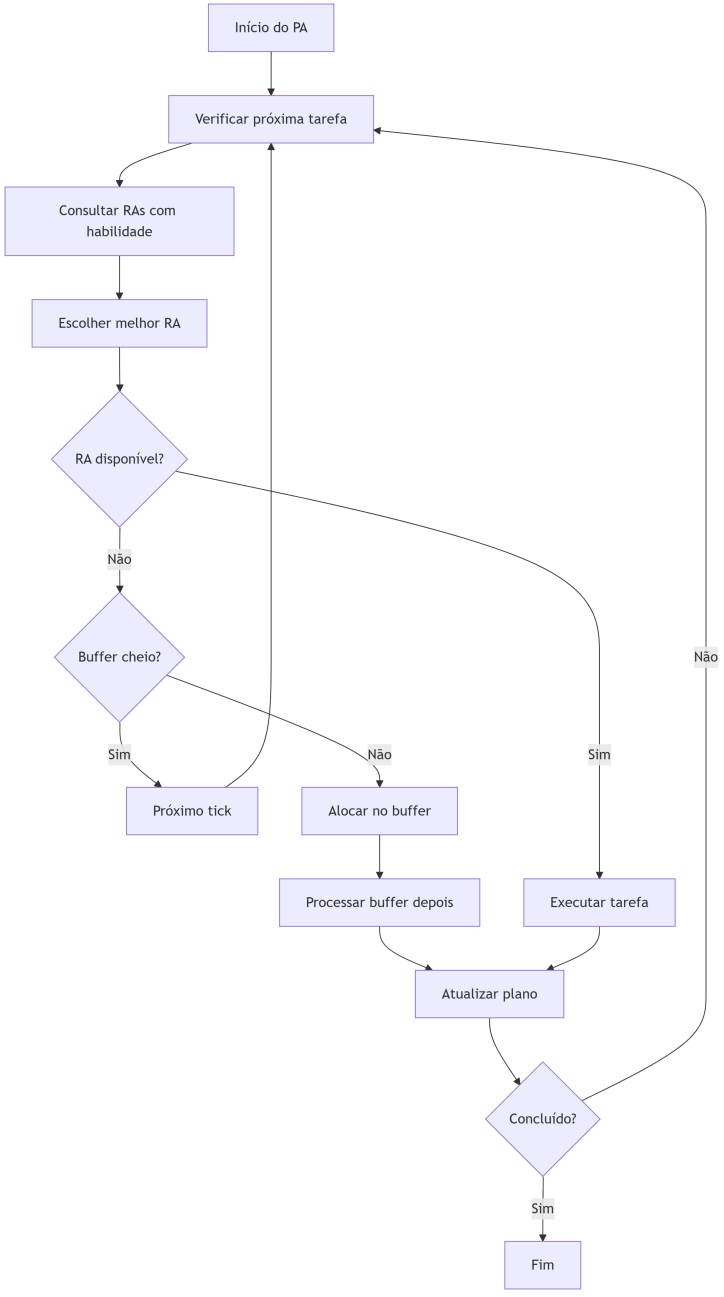
O vetor de reforço é atualizado dinamicamente com base no sucesso da execução de tarefas. Quanto melhor o desempenho da máquina, maior será seu reforço para determinada operação, influenciando a escolha futura pelos PA. O modelo ainda contempla falhas operacionais programadas, permitindo a simulação de indisponibilidades temporárias dos RA.

Essas características permitem que os agentes interajam de forma descentralizada, reproduzindo dinâmicas típicas de sistemas produtivos inteligentes e auto-organizados.

3.2.2 Interações e Dinâmicas

A dinâmica de funcionamento do sistema de produção simulado baseia-se na interação contínua entre os PA e os RA, a negociação ocorre conforme ilustrado na Figura 3.4.

Figura 3.4: Diagrama de interações



Fonte: Elaboração Própria.

Um PA, que representa um item a ser processado, precisa passar por uma série de tarefas definidas em seu plano *(process-plan)*. Para iniciar, ele entra em contato com o Sistema de Controle e informa qual a primeira tarefa que precisa ser realizada e qual a

habilidade *(skill)* necessária para essa tarefa.

O Sistema de Controle, como um gerente de produção, consulta quais RA (as máquinas ou operadores) possuem essa habilidade, em seguida o sistema considera quais máquinas são mais eficientes e quais não estão muito ocupadas no momento, no cálculo do reforço, evitando assim sobrecarregar um único recurso.

Uma vez que um RA é selecionado, a máquina pode executar a tarefa imediatamente, caso esteja disponível; se estiver ocupada, o produto precisa esperar a sua vez para ser processado. Ele é colocado no *buffer* da Máquina desse RA, que funciona como uma fila de espera com um certo limite de espaço. Se houver espaço, o produto entra na fila, caso estiver cheio, o produto não pode ser alocado naquele momento e o PA precisa esperar até que uma vaga se libere.

Quando chega a sua vez, o RA retira o primeiro produto do seu *buffer* e executa a tarefa necessária, levando um tempo específico para isso, tempo desse definido na Tabela 3.1

Após a conclusão da tarefa no RA, o produto precisa seguir para a próxima etapa do seu *process-plan*, que será realizada por outro RA. A tarefa que acabou de ser realizada é removida do seu *process-plan* e então ele faz uma nova solicitação para os RA e o ciclo inicia novamente, busca uma maquina com a habilidade, confere o reforço das máquinas com a habilidade para realizar aquela tarefa, escolhe a com maior pontuação, se ocupada aloca no *buffer*, se disponível procesa, caso não tenha máquina disponível fica buscando a cada *tick* um agente até concluir todas as tarefas e finalizar o produto e marcar ele como concluido quando não houver mais tarefas em seu process-plan.

A Tabela 3.3 apresenta a lógica de alocação do PA. Conforme mencionado anteriormente, esse comportamento é cíclico. O algoritmo abaixo representa, em pseudocódigo, essa interação com os RA.

Tabela 3.3: Algoritmo – Lógica de Alocação do Agente de Produto (PA)

Se `process-plan` não está vazio **então**
    `tarefa` ← próxima tarefa no `process-plan`
    `RAs_disponíveis` ← RAs com skill compatível
    Se `RAs_disponíveis` ≠ vazio **então**
        `RA_escolhido` ← RA com maior `reinforcement` e *buffer* disponível
        Se `RA_escolhido` está disponível **então**
            Enviar produto para `RA_escolhido`
        Senão se *buffer* do `RA_escolhido` tem espaço **então**
            Inserir produto no *buffer*
        Senão
            Aguardar próximo `tick`
    Senão
        Aguardar próximo `tick`
Senão
    Marcar produto como concluído

3.3 Mecanismo de Reforço e Inspiração Biológica

A lógica de reforço implementada neste trabalho foi inspirada em sistemas naturais de comportamento coletivo, em especial no modelo *Ants*, disponível na biblioteca *Biology* do NetLogo. Nesse modelo, formigas são representadas como agentes simples que buscam alimento e depositam feromônio no ambiente ao retornarem ao ninho. Esse feromônio atua como um mecanismo de reforço ambiental: quanto mais vezes um caminho é percorrido com sucesso, maior sua atratividade para as demais formigas. A decisão de qual direção seguir é baseada na concentração do feromônio, resultando na convergência espontânea para rotas otimizadas.

No modelo de produção proposto, esse princípio é adaptado para o domínio industrial por meio da implementação de um sistema RL simplificado. Cada recurso, máquina, mantém internamente um vetor de reforço, representando a qualidade percebida de execução de tarefas específicas. Quando uma máquina executa uma operação com sucesso e seu tempo é considerado eficiente, seu valor de reforço para aquela habilidade é aumentado. Essa lógica pode ser interpretada como uma forma de recompensa local, alinhada ao conceito central do RL, no qual agentes aprendem com base em interações e recompensas.

Além disso, para evitar uma convergência excessiva para decisões passadas, o modelo incorpora uma taxa de evaporação dos reforços ao longo do tempo. Essa característica é análoga ao *fator de desconto* usado em RL para mitigar o domínio absoluto de experiências antigas. A decisão de alocação dos produtos às máquinas é feita com base em uma função de `score`, que combina o valor de reforço da máquina com sua taxa de ocupação, incentivando a escolha de recursos eficientes e menos sobrecarregados.

A Tabela 3.4 apresenta uma comparação entre os elementos do modelo *Ants* e o sistema de produção baseado em reforço desenvolvido neste trabalho.

Tabela 3.4: Comparação entre o modelo Ants e o sistema de produção baseado em reforço

Elemento	**Descrição**
Agente	Ants: Formigas Sistema: Produtos (agentes)
Reforço	Ants: Feromônio no ambiente (patches) Sistema: Valor de reforço na máquina (por habilidade)
Localização do reforço	Ants: Ambiente (patches) Sistema: Lista interna na máquina (`reinforcement`)
Gatilho de reforço	Ants: Ao retornar ao ninho com comida Sistema: Ao concluir tarefa com desempenho satisfatório
Evaporação	Ants: Redução periódica no feromônio Sistema: Redução contínua dos reforços a cada ciclo
Lógica de decisão	Ants: Seguir caminho com maior feromônio Sistema: Escolher máquina com maior score (reforço + ociosidade)
Efeito emergente	Ants: Rotas otimizadas para alimento Sistema: Distribuição inteligente de tarefas entre máquinas
Objetivo	Ants: Reduzir o tempo de retorno ao ninho Sistema: Aumentar a eficiência produtiva e reduzir tempo total

Esse paralelismo entre o comportamento de formigas e o sistema de manufatura ilustra como regras simples de reforço local podem gerar aprendizado e adaptação coletiva. Ao integrar o conceito de aprendizado por reforço, mesmo que de forma simplificada, o modelo implementado reproduz dinâmicas de auto-organização e otimização contínua, típicas de sistemas inteligentes e descentralizados.

3.4 Parametrização e Cenários de simulação

Para a avaliação do desempenho do sistema de manufatura, foram definidos cinco cenários de simulação, cada um com um objetivo específico, com o intuito de analisar diferentes variáveis e como elas afetam o funcionamento geral do modelo. Esses cenários são fundamentais para identificar a eficiência do sistema em condições ideais e sob diferentes situações de operação, como variação na taxa de chegada de produtos, tamanho do *buffer*, e falhas operacionais.

O primeiro cenário estabelece a performance padrão do sistema em condições ideais, sem a ocorrência de falhas operacionais. Para esse cenário, o *buffer* foi configurado com o tamanho mínimo de 1, e a taxa de chegada de produtos foi fixada em zero, o que simula uma situação onde todos os produtos estão disponíveis para processamento simultaneamente desde o início da simulação. Essa configuração permite observar o comportamento dinâmico do sistema de produção, funcionando como uma linha de base para comparações com outros cenários. As métricas coletadas incluem a taxa de ocupação das máquinas, a eficiência global do processamento dos produtos, e o tempo médio de processamento por produto.

O segundo cenário foi projetado para avaliar o impacto da taxa de chegada de produtos no desempenho do sistema. Neste cenário, o *buffer* manteve o tamanho fixo de 1, enquanto a taxa de chegada foi ajustada para valores crescentes, simulando intervalos maiores entre a entrada de novos produtos no sistema. Espera-se que, sob essas condições, a eficiência do sistema aumente, pois as máquinas terão mais tempo para processar cada produto individualmente, resultando em uma ocupação das máquinas mais equilibrada e com menor sobrecarga, também pressupõe que o tempo ocioso do produto, ou seja o tempo em que ele esta fora das maquinas e fora do *buffer*, seja menor devido a uma melhor distribuição dos produtos nas maquinas.

O terceiro cenário teve como propósito analisar o efeito do tamanho do *buffer* no sistema. A taxa de chegada de produtos foi novamente fixada em zero, com todos os produtos já disponíveis para processamento. O tamanho do *buffer* apresentou diferentes

valores, como 1, 3, 5 e 7, a fim de investigar como tamanhos maiores de *buffer* afetam a eficiência do sistema. Esse cenário é importante para verificar se *buffers* maiores causam congestionamento devido ao tempo de espera aumentado ou se, por outro lado, ajudam a melhorar a eficiência do sistema ao reduzir tempos de espera e otimizar o fluxo de produção, ou até mesmo a possibilidade de não haver diferença significativa na eficiência do produto.

No quarto cenário, o foco foi testar a resiliência do sistema a falhas operacionais. Para isso, simularam-se falhas temporárias em algumas máquinas, como a *"Indexed line A"*, a *"Indexed line B"*, a *"Punching machine A"* e a *"Punching machine B"*. Essas máquinas ficaram inoperantes por intervalos de tempo específicos, definidos por dois sliders na interface gráfica, um controlando o momento de desligamento e outro ajustando o tempo de recuperação das máquinas. Esse cenário representa situações de manutenção ou avaria, com o objetivo de avaliar a capacidade do sistema de reorganizar a produção diante da indisponibilidade temporária de um recurso crítico.

Finalmente, o quinto cenário consistiu na comparação integrada dos resultados obtidos nos cenários anteriores, permitindo uma análise mais abrangente do desempenho do sistema sob diferentes condições operacionais. Foram comparadas as condições de operação normal, operação com falhas, e os ajustes realizados nos *buffers* e taxas de chegada. O objetivo deste cenário foi identificar quais configurações do sistema maximizaram a eficiência global, considerando tanto as condições ideais quanto as situações de falha e ajustes no *buffer* e na taxa de chegada de produtos. Esse cenário forneceu dados importantes para extrair conclusões sobre a robustez do sistema e propor melhorias para otimizar seu desempenho em diferentes condições operacionais.

Capítulo 4

Desenvolvimento e Implementação

Neste capítulo, é detalhado o processo de desenvolvimento e implementação do modelo proposto na plataforma NetLogo. São descritas as principais variáveis e funções associadas aos agentes, além da lógica de execução interna do sistema simulado. A capítulo aborda também a simulação de falhas nos recursos, a estrutura da interface gráfica utilizada para interação com o modelo, e a configuração dos parâmetros que influenciam o comportamento do sistema. Por fim, são apresentadas as métricas de desempenho que foram monitoradas ao longo das simulações e que servirão de base para a avaliação dos resultados nos experimentos realizados.

4.1 Aplicação no NetLogo

A aplicação no ambiente NetLogo inicia-se com a definição das características essenciais dos RA, que representam as máquinas responsáveis pela execução das tarefas no sistema produtivo. Dentro deste conjunto de variáveis, destacam-se:

- **Disponibilidade (*available*)**: Esta variável booleana indica o estado operacional da máquina na linha de produção. Um valor verdadeiro significa que a máquina está apta a operar, ou seja ligada, enquanto um valor falso pode ser interpretado como a máquina estando inativa devido a uma falha, manutenção ou indisponibilidade por

outros motivos.

- **Ocupação (*busy*)**: Esta variável lógica sinaliza se a máquina está atualmente processando um produto. Um valor verdadeiro indica que a máquina está ocupada e, portanto, não pode receber um novo item para processamento imediato naquele momento. Essa informação é fundamental para definir e distribuir as tarefas.

- **Buffer (*buffer*)**: O *buffer* representa a capacidade de armazenamento da máquina, funcionando como uma fila de espera ou um espaço físico adjacente onde os produtos aguardam para serem processados. A máquina, ao executar uma tarefa, busca o próximo produto disponível em seu *buffer*. A gestão eficiente dos *buffers* influencia diretamente o fluxo de materiais e a prevenção de gargalos na produção.

- **Valor de Reforço (*reinforcement*)**: Este valor numérico associado a cada habilidade que a maquina possui, reflete seu desempenho ou adequação para realizar as tarefas. Inicialmente definido com um valor base, ele é dinamicamente atualizado ao longo da simulação com base no sucesso ou eficiência da máquina na execução das tarefas. Esse valor desempenha um papel fundamental nos mecanismos de seleção da "melhor"máquina para uma determinada operação.

- **Habilidades (*skills*)**: Esta variável consiste em um vetor ou lista que define as competências de cada máquina. Cada elemento do vetor corresponde a um tipo de tarefa possível no sistema. Um valor diferente de zero na posição correspondente a uma determinada tarefa indica que a máquina possui a habilidade necessária para executá-la.

Para a modelagem dos PA em NetLogo, definimos algumas variáveis cruciais que governam seu fluxo e processamento ao longo do sistema produtivo:

- **Tipo de Produto (*product-type*)**: Esta variável categórica armazena a identificação do produto criado, podendo assumir os valores *"star"* ou *"triangle"*. Essa distinção é fundamental, pois determina a sequência específica de tarefas que cada tipo de produto deve seguir.

- **Plano de Processamento (*process-plan*)**: Esta variável do tipo lista (ou vetor) define a rota de processamento do produto, ou seja, a sequência ordenada de tarefas que ele deve executar. Essa sequência é determinada com base no *product-type*, conforme especificado na Tabela 3.2.
- **Em Processamento (*in-process*)**: Esta variável booleana indica o estado atual do produto em relação às máquinas de processamento. Seu valor é true enquanto o produto está sendo processado ativamente por uma máquina. Nos períodos em que o produto se encontra em um *buffer*, aguardando processamento, ou desde sua criação até ser alocado a uma máquina, o valor desta variável é falsa.

4.2 Mecanismo de Reforço e Evaporação

O modelo implementado no NetLogo utiliza um mecanismo de reforço para guiar a tomada de decisão dos agentes de produto quanto à escolha dos recursos para execução de tarefas. Essa lógica é incorporada diretamente nas estruturas e procedimentos do código, com base em uma lista de reforços mantida individualmente por cada agente de recurso RA.

Cada máquina possui duas listas principais:

- `skills`: vetor binário indicando quais tarefas a máquina é capaz de executar.
- `reinforcement`: vetor numérico com o valor de reforço associado a cada habilidade.

Por exemplo:

```
set skills [0 0 1 0 0 0] ;; máquina que realiza "punch_1"
set reinforcement [0 0 10 0 0 0]
```

4.2.1 Reforço após execução bem-sucedida

Ao final de cada tarefa, se o tempo de execução (`duration`) for considerado adequado, o valor de reforço da máquina é aumentado. Isso ocorre dentro do procedimento `execute-task`, conforme o trecho a seguir:

```
let base-reinforcement 10
let time-factor (1 / duration)
let new-reinforcement (base-reinforcement * time-factor)

if duration <= max-wait-time [
    let current-reinforcement item current-skill-index reinforcement
    set current-reinforcement current-reinforcement + new-reinforcement

    set reinforcement replace-item current-skill-index ...
    reinforcement current-reinforcement
]
```

Esse código calcula um reforço proporcional ao desempenho (menor tempo = maior reforço) e atualiza o valor armazenado para a respectiva habilidade.

4.2.2 Evaporação periódica dos reforços

Para evitar que reforços antigos dominem o processo decisório indefinidamente, o modelo aplica uma evaporação constante ao longo do tempo. Esse processo é executado a cada ciclo (`tick`) no bloco principal `go`, como segue:

```
ask physical-agents [
    let evaporation-rate 0.95
    set reinforcement map [r -> r * evaporation-rate] reinforcement
    set reinforcement map [r -> max (list r 1)] reinforcement  ;; valor mínimo
]
```

A taxa de evaporação garante que os reforços diminuam gradualmente, permitindo que novas máquinas ou mudanças no desempenho afetem o processo de escolha dos agentes de forma dinâmica.

4.2.3 Cálculo do score para seleção da máquina

A função `calculate-score` combina o valor de reforço da máquina com sua taxa de ocupação, permitindo balancear a escolha entre eficiência histórica e disponibilidade atual:

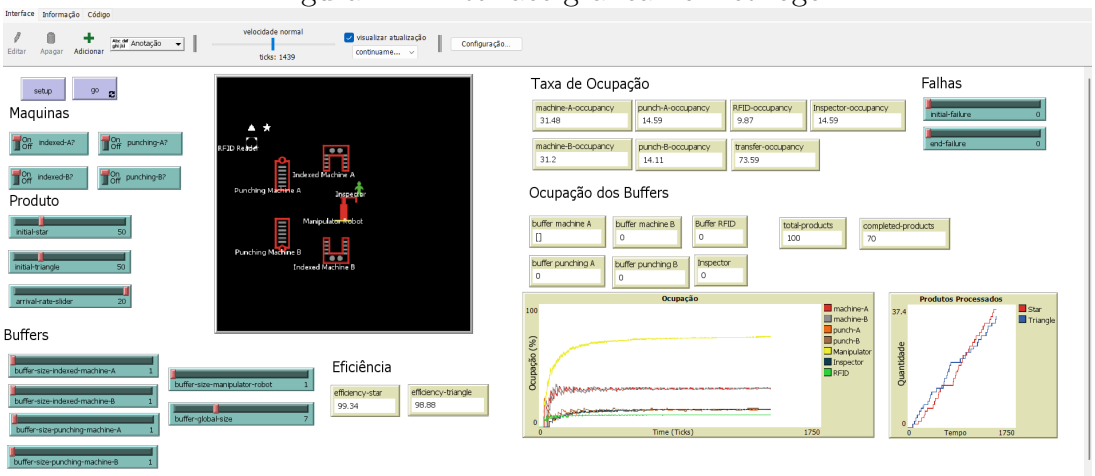
```
report ((current-reinforcement * 0.5) + (100 - occupancy) * 0.5)
```

Com isso, o sistema implementa uma forma aproximada de aprendizado por reforço, permitindo que os recursos mais eficientes e menos sobrecarregados sejam preferidos pelos agentes de produto ao longo do tempo.

4.3 Parametrização na Interface

A interface gráfica desenvolvida no NetLogo foi estruturada para permitir a configuração dinâmica dos parâmetros da simulação, bem como a observação em tempo real do comportamento dos agentes e dos resultados do sistema fabril. A Figura 4.1 apresenta a interface completa da simulação, destacando seus principais componentes.

Figura 4.1: Interface gráfica no NetLogo



Fonte: Elaboração Própria.

4.3.1 Controle de Máquinas e Produtos

Na parte esquerda da interface, vista na Figura 4.1 os botões de controle para ligar ou desligar as máquinas *Indexed A/B* e *Punching A/B*. A quantidade inicial de produtos inseridos na simulação é controlada por meio dos *sliders* `initial-star` e `initial-triangle`, que definem, respectivamente, o número de produtos do tipo *Star* e *Triangle*. A taxa de chegada de novos produtos é parametrizada por meio do slider `arrival-rate-slider`, simulando diferentes fluxos de entrada de produtos no sistema.

4.3.2 Configuração de Buffers

Cada máquina possui um *slider* associado para determinar o tamanho de seu *buffer*, ou seja, a capacidade de armazenar produtos aguardando processamento. Os *sliders* individuais, permitem ajustar esse valor para cada estação. Adicionalmente, há um *slider* que pode ser utilizado para alterar a capacidade de todos os *buffers* de forma centralizada.

4.3.3 Indicadores e Monitoramento

Diversos monitores exibem métricas quantitativas em tempo real, incluindo:

- Taxa de ocupação de cada máquina
- Número total de produtos processados
- Eficiência média de processamento
- Eficiência média de processamento
- Estado atual dos *buffers* individuais, com quantos produtos estão nessa espaço.

4.3.4 Visualização Gráfica

Dois gráficos principais, localizados na parte inferior direita, complementam a análise:

- **Ocupação (%)**: gráfico temporal que mostra a taxa de uso de cada máquina ao longo da simulação, permitindo identificar períodos de ociosidade ou sobrecarga;
- **Produtos Processados**: gráfico que mostra o número de produtos do tipo *Star* e *Triangle* concluídos ao longo do tempo.

4.4 Implementação de Falhas

A implementação de falhas no sistema simulado tem como alvo representar situações realistas de interrupção temporária de funcionamento das máquinas, como ocorre frequentemente em ambientes industriais devido a falhas operacionais ou manutenções corretivas. Essa funcionalidade é fundamental para avaliar a resiliência do sistema produtivo diante da indisponibilidade de recursos críticos, permitindo observar como os agentes se reorganizam de forma autônoma para garantir a continuidade da produção.

No modelo desenvolvido, cada RA possui uma variável booleana denominada *available*, a qual indica se o recurso está operacional (`true`) ou inativo (`false`). A lógica de falhas é controlada por dois parâmetros definidos na interface do NetLogo:

- **initial-failure**: momento (em *ticks*) no qual o RA deixa de estar disponível.
- **end-failure**: intervalo de tempo (em *ticks*) em que o RA permanece inoperante.

Quando o tempo de simulação (*ticks*) atinge o valor definido como initial-failure, a máquina é desativada, tendo sua variável `available` atribuída como `false`. Durante esse período, o agente deixa de aceitar novos produtos para processamento e interrompe quaisquer tarefas em andamento. Ao final do intervalo estabelecido por end-failure, o recurso é automaticamente reabilitado e volta ao seu estado de operação normal.

Durante o período de falha, os PA que necessitam da habilidade daquele RA devem buscar outros recursos alternativos com a mesma competência. Caso não encontrem nenhuma opção disponível ou com espaço livre no *buffer*, permanecem em espera até que o recurso desejado volte a operar ou que outro se torne acessível. Essa lógica permite

observar efeitos emergentes como acúmulo de produtos nos *buffers*, aumento do tempo de espera e redirecionamento da carga de trabalho entre os recursos disponíveis.

Essa funcionalidade foi testada em diferentes cenários, variando-se o tempo de início e a duração das falhas, bem como o tipo de máquina afetada (por exemplo, falhas nas prensas ou nas linhas indexadas). Esses testes possibilitaram uma análise detalhada sobre o impacto das falhas na eficiência global do sistema, no tempo médio de processamento dos produtos e na taxa de ocupação dos recursos alternativos.

4.5 Métricas de Monitoramento

Para avaliar o desempenho do sistema de manufatura modelado, foram definidas métricas quantitativas que capturam aspectos relevantes da operação, como ocupação dos recursos, eficiência produtiva e tempo de ciclo dos produtos. A coleta e análise desses dados foram fundamentais para comparar os diferentes cenários de simulação propostos.

Devido à natureza estocástica do sistema, ou seja, à sua dependência de eventos aleatórios como a ordem de chegada dos produtos, a alocação dinâmica nos *buffers* e a ocorrência de falhas — uma única simulação não é suficiente para capturar com precisão o comportamento global do modelo. Para mitigar essa variabilidade, foi utilizada a ferramenta **BehaviorSpace** do NetLogo, que permite a execução automatizada de múltiplas simulações com os mesmos parâmetros e coleta sistemática de dados para posterior análise estatística.

Cada experimento foi executado com um número pré-definido de parâmetros, foram feitas 50 repetições para cada conjunto de parâmetros, e os valores médios das métricas foram utilizados para análise comparativa.

Durante a execução das simulações no *BehaviorSpace*, foram coletadas automaticamente as seguintes métricas, permitindo uma análise quantitativa do desempenho do sistema em diferentes cenários:

- **Completed-products**: quantidade total de produtos concluídos ao final da simulação, considerando ambos os tipos (Star e Triangle).

- **Machine-A-occupancy**: taxa média de ocupação da máquina *Indexed A*, expressa em percentual, indicando o tempo em que esteve ativamente processando produtos.
- **Machine-B-occupancy**: taxa média de ocupação da máquina *Indexed B*.
- **Punch-A-occupancy**: percentual de tempo em que a máquina *Punching A* esteve em operação.
- **Punch-B-occupancy**: percentual de ocupação da máquina *Punching B* ao longo da simulação.
- **Transfer-occupancy**: taxa de utilização do robô manipulador responsável pelo transporte dos produtos entre as máquinas.
- **RFID-occupancy**: ocupação média do leitor RFID, utilizado na etapa inicial do processo para identificação dos produtos.
- **Inspector-occupancy**: tempo médio de ocupação do agente responsável pela inspeção final dos produtos.
- **Efficiency-star**: eficiência média na produção de itens do tipo *Star*, medida pela relação entre tempo de processamento ativo e tempo total de ciclo.
- **Efficiency-triangle**: eficiência média na produção de itens do tipo *Triangle*.
- **Tempo-star**: tempo médio total (em *ticks*) necessário para processar um produto do tipo *Star*, incluindo espera, processamento e movimentação.
- **Tempo-triangle**: tempo médio total (em *ticks*) necessário para completar um produto do tipo *Triangle*.

Essas métricas serviram de base para a análise estatística e comparativa entre os diferentes cenários de simulação, como variações no tamanho dos *buffers*, frequência de falhas e taxa de chegada de produtos. As métricas foram exibidas tanto na interface gráfica, por meio de monitores e gráficos em tempo real, quanto exportadas em arquivos

.csv através do BehaviorSpace, permitindo a consolidação e o tratamento dos dados em *softwares* externos de análise, como Excel.

Esse processo de monitoramento estatístico garantiu maior confiabilidade aos resultados obtidos, tornando possível identificar tendências, comparar configurações e validar o impacto cada parâmetros, dessa forma, o uso integrado do BehaviorSpace com a interface gráfica interativa proporcionou uma base robusta para avaliar, comparar e compreender o comportamento do sistema simulado sob múltiplas perspectivas operacionais.

Capítulo 5

Resultados e Discussão

Este capítulo apresenta os resultados dos testes realizados com o modelo implementado, com a finalidade de avaliar seu desempenho sob diferentes condições operacionais. Cada cenário de simulação é descrito com suas configurações específicas, justificando os parâmetros utilizados e os resultados esperados. Em seguida, são analisados os dados obtidos, com foco em métricas como eficiência, tempo de processamento e taxa de ocupação dos recursos. A discussão dos resultados permite identificar padrões de comportamento do sistema, validar hipóteses levantadas durante o projeto e propor interpretações relevantes a partir das observações empíricas.

5.1 Cenário 1 - Avaliação Padrão do Sistema

Este cenário visa estabelecer um ponto de referência para análise comparativa nos cenários subsequentes. A configuração simula uma situação de carga total inicial, sem perturbações externas, permitindo avaliar o comportamento intrínseco do sistema sob severas restrições de fluxo. Os parâmetros utilizados foram:

- **Tamanho do buffer:** 1 (mínimo)
- **Taxa de chegada:** 0 (todos os produtos inseridos simultaneamente no início)
- **Falhas:** desativadas (recursos sempre disponíveis)

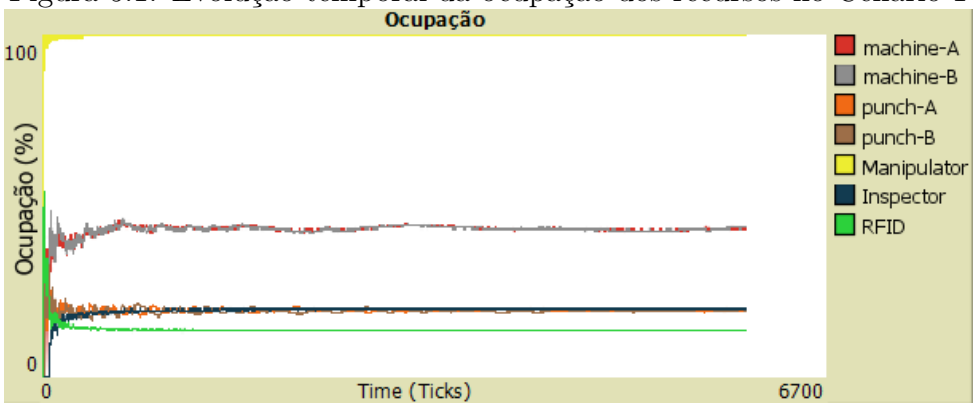
- **Quantidade de produtos:** 200 do tipo *Star* e 200 do tipo *Triangle*

Essa configuração representa uma situação de estresse operacional, com alta concentração de produtos desde o primeiro *tick*, o que favorece a identificação de gargalos e limitações estruturais no modelo.

Os resultados, obtidos a partir da média de 20 execuções independentes, indicam um desempenho global insatisfatório. A eficiência média de processamento foi de apenas 3,32% para os produtos do tipo *Star* e 3,22% para os do tipo *Triangle*, o que reflete altos tempos de espera fora das estações, tempo de produto ocioso. O tempo médio de conclusão dos produtos foi de 3.219,28 *ticks* para o tipo *Star* e 2.847,51 *ticks* para o tipo *Triangle* valores significativamente elevados, mesmo na ausência de falhas operacionais.

A Figura 5.1 apresenta o gráfico de ocupação percentual dos recursos ao longo do tempo de simulação no Cenário 1. Observa-se que, apesar da baixa eficiência global do sistema, há uma distribuição relativamente equilibrada da carga de trabalho entre os recursos com funções equivalentes.

Figura 5.1: Evolução temporal da ocupação dos recursos no Cenário 1



Fonte: Elaboração Própria.

As máquinas *Machine A* e *Machine B*, responsáveis pelas operações de perfuração (*drill_1* e *drill_2*), mantêm níveis de ocupação similares ao longo da simulação, assim como as máquinas *Punch A* e *Punch B*, que realizam as tarefas de puncionamento. Esse equilíbrio indica que o mecanismo de alocação descentralizada baseado em reforço foi capaz de distribuir os produtos de forma estável entre as alternativas disponíveis, evitando a sobrecarga de qualquer unidade específica.

O recurso mais crítico é o robô Manipulador (linha amarela), cuja ocupação atinge e se mantém próxima de 100% durante toda a simulação. Essa saturação contínua revela a existência de um gargalo operacional no transporte entre estações, o que impacta diretamente o fluxo de produção e contribui para os elevados tempos de ciclo registrados.

Por outro lado, os agentes RFID (verde claro) e Inspector (azul claro) apresentam taxas de ocupação significativamente inferiores - comportamento esperado, uma vez que esses agentes executam operações pontuais e de curta duração nos extremos da cadeia produtiva: o RFID realiza a leitura de identificação no início do processo, enquanto o Inspector apenas valida a conclusão da peça no final. O tempo reduzido dessas tarefas, aliado ao fato de que cada produto as utiliza apenas uma vez, explica a baixa demanda por esses recursos ao longo do tempo.

Os resultados evidenciam que a combinação de *buffers* extremamente reduzidos com carga total inicial simultânea leva à saturação precoce dos recursos logísticos (em especial do robô) e limita significativamente a fluidez do sistema. Como consequência, os produtos enfrentam longos períodos de espera para alocação, resultando em baixa eficiência global e elevado tempo de ciclo. Esse cenário demonstra que, mesmo em um sistema funcional (sem falhas ou paradas), a configuração inadequada de entrada e *buffers* pode comprometer drasticamente o desempenho de toda a linha de produção.

É possível observar que mesmo em um ambiente sem perturbações externas, os gargalos internos do sistema (principalmente o robô transferidor) limitam significativamente a eficiência global. Os dados aqui apresentados servirão como referência para avaliar melhorias nos próximos cenários, que considerarão alterações na taxa de chegada e na capacidade dos *buffers*.

5.2 Cenário 2 - Impacto da Variação na Taxa de Chegada de Produtos

O segundo cenário teve como objetivo avaliar a influência da taxa de chegada de produtos sobre o desempenho do sistema de produção. Para isso, foram realizados experimentos com diferentes intervalos entre a chegada de novos produtos: 0, 3, 5, 10, 15, 18 e 20 *ticks*. As demais configurações foram: *buffers* de tamanho 1 para todas as máquinas e um total de 400 produtos processados (200 do tipo *Star* e 200 do tipo *Triangle*).

A Tabela 5.1 apresenta as principais métricas coletadas durante as simulações.

Tabela 5.1: Resultados do cenário com variação da taxa de chegada

Taxa de Chegada	**Ef. Star (%)**	**Ef. Triangle (%)**	**Tempo Star (ticks)**	**Tempo Triangle (ticks)**	**Ocup. Transfer (%)**
0	3,32	3,22	3219,28	2847,51	99,57
3	5,82	6,27	2450,33	2418,14	99,78
5	8,45	7,20	1909,27	2167,20	99,67
10	12,72	12,59	1097,66	995,64	99,47
15	66,16	64,79	58,37	59,46	99,17
18	82,90	80,38	47,09	47,95	82,84
20	85,90	85,87	45,17	44,63	74,62

Os resultados apresentados na Tabela 5.1 demonstram de forma clara a influência da taxa de chegada de produtos no desempenho do sistema produtivo. Quando a taxa de chegada é elevada (valores baixos de intervalo, como 0, 3 e 5 *ticks*), a eficiência do sistema é extremamente baixa, com índices inferiores a 9% para ambos os tipos de produtos. Nessas configurações, o tempo médio de ciclo ultrapassa 2000 ticks, chegando a mais de 3000 *ticks* na simulação com taxa 0. Esse tempo elevado se deve, majoritariamente, ao tempo ocioso dos produtos aguardando acesso aos recursos, principalmente ao agente manipulador.

Observa-se que a ocupação do manipulador permanece próxima de 100% até a taxa de chegada de 10 *ticks*, o que evidencia um gargalo crítico na movimentação entre estações. Isso impacta diretamente a fluidez do sistema e acarreta congestionamento de produtos

fora dos *buffers*, limitando a capacidade de resposta das máquinas.

A partir de uma taxa de chegada de 15 *ticks*, há uma mudança significativa no comportamento do sistema: a eficiência dos produtos *Star* e *Triangle* sobe para mais de 60%, e os tempos médios de processamento caem drasticamente, ficando abaixo de 100 ticks. Com taxas ainda mais espaçadas (18 e 20), o sistema atinge um nível satisfatório de eficiência, superando 80% de aproveitamento com tempos médios inferiores a 50 *ticks*. Nesses casos, a ocupação do manipulador também diminui consideravelmente, indicando que o sistema opera com maior equilíbrio e menor saturação dos recursos.

Este cenário comprova que a taxa de chegada é um dos fatores mais sensíveis para a performance do sistema. A introdução de produtos em intervalos menores provoca sobrecarga e saturação dos recursos, resultando em menor eficiência e maior tempo de ciclo. Por outro lado, uma taxa moderada (entre 18 e 20 *ticks*) favorece a auto-organização do sistema, melhora a eficiência e reduz gargalos, mesmo com *buffers* limitados.

Com base nos dados da Tabela 5.1, observam-se os seguintes comportamentos:

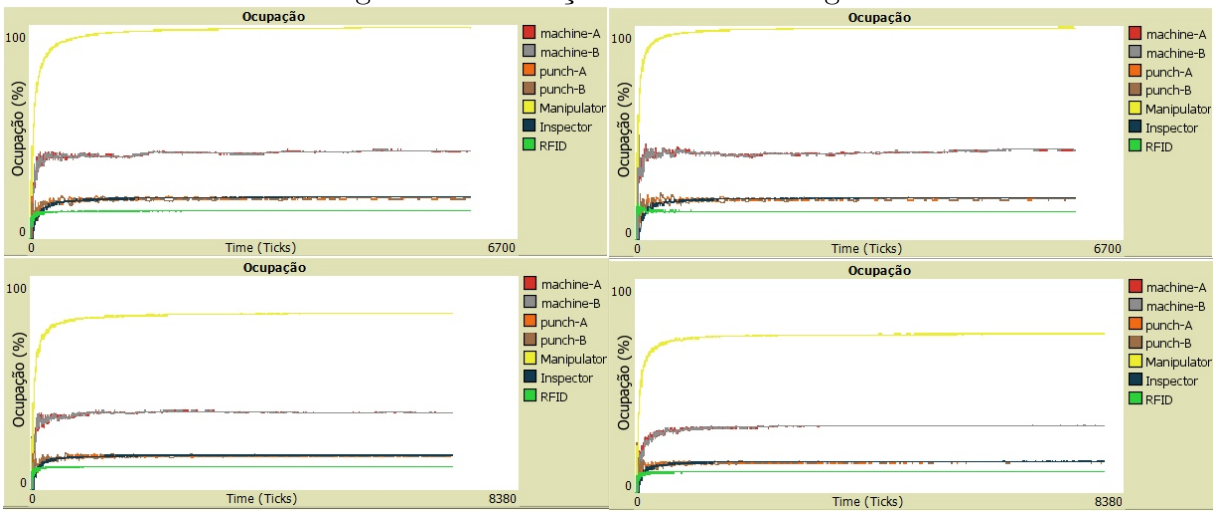
- **Eficiência**: Aumenta progressivamente conforme a taxa de chegada é elevada. Em taxa 0, a eficiência é muito baixa devido à sobrecarga inicial. Em taxa 20, os produtos fluem com regularidade, resultando em um uso muito mais eficiente dos recursos.

- **Tempo de produção**: Os tempos médios de processamento caem drasticamente. Produtos do tipo *Star* reduzem seu tempo de 3219,28 para apenas 45,17 *ticks*, enquanto os *Triangle* caem de 2847,51 para 44,63 *ticks*.

- **Robô manipulador (*Transfer*)**: Começa com uso extremo 99,57%, sendo claramente um gargalo quando todos os produtos chegam ao mesmo tempo. Com taxa 20, sua ocupação é reduzida para 74,62%, demonstrando alívio na carga operacional.

A Figura 5.2 mostra a taxa de ocupação das máquinas ao longo da simulação em quatro cenários distintos, cada um com uma taxa de chegada de produtos diferente: 10, 15, 18 e 20, respectivamente.

Figura 5.2: Variação da Taxa de chegada



Fonte: Elaboração Própria.

Nos dois primeiros gráficos, com taxas de chegada de 10 e 15 produtos por intervalo, a ocupação do Manipulator permanece alta, próximo a 99%, indicando que ele opera continuamente, sem períodos de ociosidade. Essa demanda elevada resulta em tempos de espera mais longos para os produtos, conforme mostrado na Tabela 5.1.

Nos dois últimos gráficos, com taxas de chegada de 18 e 20, há uma redução significativa na ocupação do Manipulator, chegando a aproximadamente 70% no último caso. Essa diminuição indica que o fluxo de produtos foi distribuído de forma mais equilibrada ao longo do tempo, reduzindo o congestionamento do agente responsável pelo transporte entre as estações. Como resultado, o sistema pode ter operado de maneira mais eficiente, evitando acúmulos excessivos nos *buffers* e melhorando o tempo médio de produção.

Esses resultados confirmam que o controle da taxa de entrada dos produtos é essencial em sistemas com *buffers* limitados. A inserção gradual dos produtos reduz os conflitos por recursos, evita filas extensas e melhora drasticamente o desempenho geral do sistema. A taxa 20 foi a mais eficiente, servindo como referência para comparações futuras.

5.3 Cenário 3 - Influência do Tamanho dos Buffers

Neste cenário, o objetivo foi avaliar como o tamanho dos *buffers* influencia o desempenho do sistema de produção. A taxa de chegada de produtos foi fixada em 20 produtos por intervalo, a fim de manter um fluxo contínuo sem tempos ociosos excessivos. A carga inicial foi de 200 produtos do tipo *star* e 200 do tipo *triangle*. Foram testadas três configurações distintas de *buffers*, com capacidades iguais a 3, 5 e 7 unidades para cada máquina. Para cada configuração, foram realizadas 10 simulações independentes, e os resultados médios estão apresentados na Tabela 5.2.

Tabela 5.2: Resultados do cenário 3 com variação do tamanho dos *buffers* (taxa de chegada = 20)

Tamanho do *buffer*	**Ef. Star (%)**	**Ef. Triangle (%)**	**Tempo Star (ticks)**	**Tempo Triangle (ticks)**	**Ocup. Transfer (%)**
3	86,42	85,17	45,27	45,44	74,59
5	86,08	85,67	45,38	45,13	74,61
7	86,58	85,66	44,95	44,79	74,60

Os resultados indicam que a variação do tamanho do *buffer* entre os valores testados teve impacto marginal sobre as métricas principais do sistema. A eficiência média dos produtos permaneceu praticamente estável, variando entre 86,08% e 86,58% para o tipo *star* e entre 85,17% e 85,67% para o tipo *triangle*. O tempo médio de produção apresentou leve tendência de redução com o aumento do *buffer:* de 45,27 *ticks* (*buffer* 3) para 44,95 *ticks* (*buffer* 7) para o tipo *star*, e de 45,44 *ticks* para 44,79 *ticks* para o tipo *triangle*.

As ocupações dos recursos produtivos (máquinas e transportador) também mantiveram estabilidade. O robô manipulador registrou ocupações próximas a 74,6% em todos os testes, enquanto as máquinas Indexed e Punch variaram pouco em torno de 31% e 14%, respectivamente. RFID e o agente *Inspector* permaneceram com ocupações baixas, como esperado, dado seu uso pontual e de curta duração.

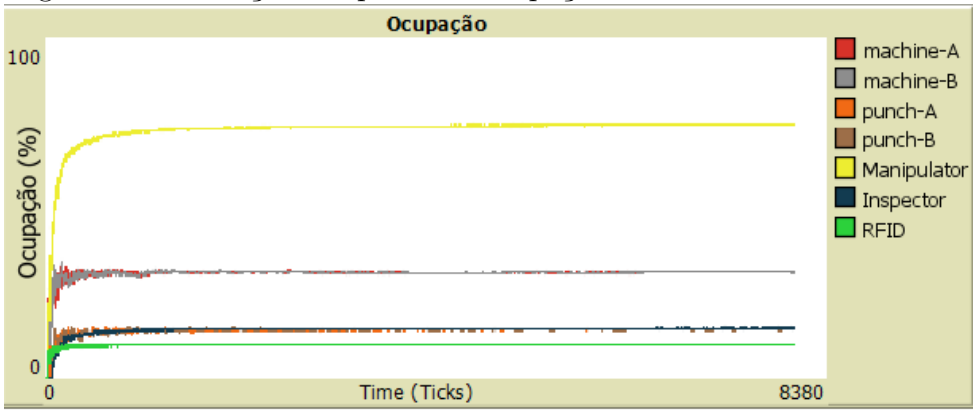
Esses resultados sugerem que, dentro da faixa testada, o aumento do tamanho dos *buffers* não comprometeu a eficiência do sistema, tampouco gerou gargalos adicionais. Ao

contrário, *buffers* ligeiramente maiores podem contribuir para uma leve melhoria nos tempos de produção, possivelmente pela redução de bloqueios temporários entre operações. No entanto, dado o impacto pequeno observado, não se justificam aumentos excessivos que possam elevar o custo de armazenamento ou complexidade física do sistema.

Observa-se que a eficiência dos produtos manteve-se acima de 85% em todas as configurações, com variações mínimas. O tempo médio de produção para ambos os tipos de produto apresentou ligeira redução com o aumento do *buffer*, indicando uma pequena melhoria na fluidez do sistema à medida que se reduz o bloqueio de operações por falta de espaço disponível.

A Figura 5.3 apresenta o gráfico de ocupação dos recursos ao longo do tempo para a configuração com *buffer* igual a 5. Nota-se uma distribuição equilibrada da carga entre os recursos com funções semelhantes: *Machine A* e *Machine B* apresentam ocupações médias semelhantes, assim como *Punch A* e *Punch B*. Isso demonstra que o mecanismo de alocação baseado em reforço continua eficiente mesmo com *buffers* maiores, sem induzir sobrecargas localizadas.

Figura 5.3: Evolução temporal da ocupação dos recursos no Cenário 4



Fonte: Elaboração Própria.

O robô manipulador (linha amarela) manteve um padrão de ocupação alto, próximo a 75%, porém sem atingir a saturação observada no Cenário 1. Isso reforça que a combinação de *buffers* adequados com uma taxa de chegada controlada permite um melhor aproveitamento dos recursos logísticos, resultando em maior fluidez e tempos de ciclo reduzidos.

5.4 Cenário 4 - Análise de Resiliência frente a Falhas Operacionais

Neste cenário, buscou-se avaliar a capacidade do sistema de reorganizar sua operação diante da falha temporária de uma máquina crítica. Para isso, foi simulada a interrupção da *Machine A* no *tick* 850, com duração de 2000 *ticks*. Durante esse período, a máquina ficou indisponível (`available = false`) e não pôde receber ou processar novos produtos. A reativação automática ocorreu ao final do período de falha, restaurando a máquina à operação normal.

A configuração do sistema incluiu uma taxa de chegada de produtos de 20 *ticks* e *buffers* de tamanho 5 para todas as máquinas. O total de produtos processados foi de 400, divididos igualmente entre os tipos *Star* e *Triangle*.

A Figura 5.4 apresenta a evolução da taxa de ocupação dos recursos ao longo do tempo. É possível observar, de forma clara, os seguintes comportamentos:

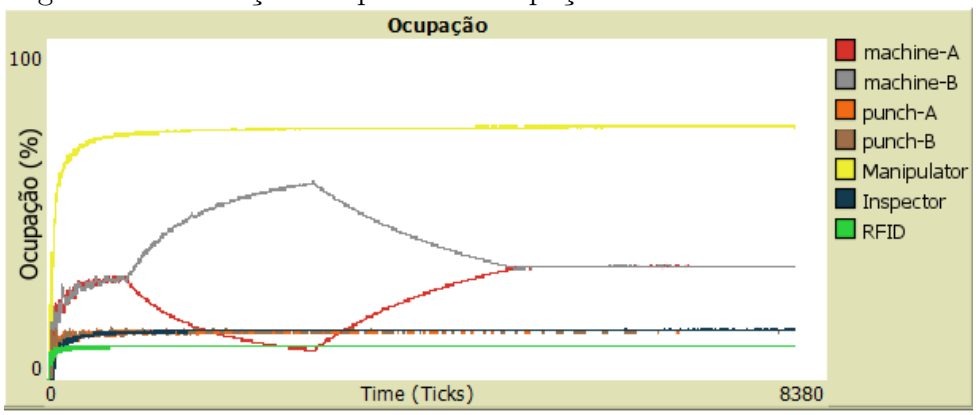
- A *Machine B* (linha cinza) assumiu gradualmente a carga que era originalmente compartilhada com a *Machine A* (linha vermelha), elevando sua ocupação continuamente após o *tick* 850. Isso indica que o sistema de alocação foi capaz de redirecionar as tarefas automaticamente.

- Após o retorno da *Machine A*, sua taxa de ocupação voltou a subir até alcançar um equilíbrio com a *Machine B*, demonstrando que o mecanismo de reforço foi recuperado ao longo do tempo, readaptando as decisões dos produtos.

- O *manipulador* (linha amarela) manteve ocupação próxima de 75% durante toda a simulação, reforçando seu papel como principal gargalo estrutural do sistema, mesmo em cenários com falhas.

- As máquinas de *punch* e o *inspector* apresentaram comportamento estável, indicando que o impacto da falha foi limitado à etapa de furação, sem comprometer severamente as etapas subsequentes.

Figura 5.4: Evolução temporal da ocupação dos recursos no Cenário 4



Fonte: Elaboração Própria.

Este experimento demonstra que o sistema é resiliente frente a falhas temporárias, principalmente devido à lógica de reforço adaptativo implementada. Apesar do aumento momentâneo da ocupação de recursos alternativos, a redistribuição de tarefas evitou bloqueios prolongados ou perda significativa de desempenho.

A introdução de *buffers* com capacidade intermediária (5 unidades) foi essencial para amortecer os efeitos da indisponibilidade temporária da *machine-A*, permitindo o acúmulo e posterior processamento dos produtos. No entanto, o tempo de ciclo médio dos produtos foi impactado, sugerindo que o tempo de recuperação do sistema está diretamente relacionado à duração da falha e à carga acumulada.

A Tabela 5.3 evidencia a resiliência do sistema frente a falhas operacionais. A eficiência dos produtos sofreu uma leve redução com a falha da *machine-A*, especialmente para os

produtos *Triangle*, cuja eficiência caiu de 86,08% para 83,99%. Além disso, o tempo médio de processamento aumentou para ambos os tipos de produto, passando de 45,22 para 46,38 *ticks* no caso do *Star*, e de 44,97 para 46,72 *ticks* no caso do *Triangle*.

Tabela 5.3: Comparação da eficiência e tempo médio dos produtos com e sem falha operacional (Cenário 4)

Condição	**Ef. Star (%)**	**Ef. Triangle (%)**	**Tempo Star (ticks)**	**Tempo Triangle (ticks)**
Sem falhas	86,52	86,08	45,22	44,97
Com falha (850 até 2000)	85,13	83,99	46,38	46,72

Esse acréscimo sugere que, embora o sistema tenha evitado bloqueios completos, houve maior acúmulo de produtos nos *buffers* e redistribuição da carga de trabalho entre os recursos disponíveis. Assim, o tempo de recuperação operacional está diretamente ligado à duração da falha, ao nível de ocupação dos recursos alternativos e à configuração dos *buffers*. Ainda assim, o sistema manteve o número total de produtos processados (400), reforçando sua capacidade de adaptação com base na lógica de reforço implementada.

5.5 Cenário 5 - Comparação entre os Cenários Simulados

O Cenário 5 tem como finalidade comparar os desempenhos obtidos nos quatro experimentos anteriores, permitindo avaliar o impacto de diferentes variáveis operacionais sobre a eficiência e o tempo médio de produção dos produtos *Star* e *Triangle*. Os resultados são apresentados na Tabela 5.4.

Tabela 5.4: Resumo comparativo dos cenários simulados (Cenário 5)

	Cenário	**Ef. Star (%)**	**Ef. Triangle (%)**	**Tempo Star (ticks)**	**Tempo Triangle (ticks)**
1	Sem falhas, *buffer = 1, chegada = 0*	3,32	3,22	3219,28	2847,51
2	Chegada = 20, *buffer = 1*	85,90	85,87	45,17	44,63
3	Chegada = 20, *buffer = 5*	86,08	85,67	45,38	45,13
4	Falha machine-A (tick 850 a 2000)	85,13	83,99	46,38	46,72

No Cenário 1, a combinação de *buffers* mínimos e inserção instantânea de todos os produtos resultou em um gargalo severo no sistema, com baixíssimos níveis de eficiência (em torno de 3%) e tempos de ciclo extremamente elevados para ambos os tipos de produto.

No Cenário 2, ao ajustar a taxa de chegada para 20 *ticks*, mesmo mantendo os *buffers* no mínimo, houve uma melhoria drástica no desempenho. As eficiências superaram 85%, com tempos médios de produção próximos a 45 *ticks*, indicando que o simples controle do fluxo de entrada pode aliviar a pressão sobre os recursos logísticos.

O Cenário 3 avaliou o impacto do aumento da capacidade dos *buffers*, mantendo a mesma taxa de chegada do cenário anterior. Os resultados mostraram variações mínimas nas métricas, sugerindo que o efeito de *buffers* maiores é marginal quando a taxa de chegada já está ajustada. A eficiência se manteve estável, e os tempos médios de produção foram praticamente iguais aos do Cenário 2.

O Cenário 4 introduziu falhas temporárias na *Punching machine A*, simulando uma situação de avaria. Observou-se uma pequena queda na eficiência e um aumento moderado nos tempos de produção, revelando a resiliência parcial do sistema. O mecanismo de alocação por reforço contribuiu para redistribuir as tarefas para a *Punching machine B*, evitando colapsos operacionais, embora com impacto no tempo de recuperação.

Capítulo 6

Conclusões

Este trabalho apresentou o desenvolvimento e a avaliação de um modelo de simulação baseado em agentes (ABM) aplicado a um sistema ciber-físico de manufatura inteligente, com implementação na plataforma NetLogo. O modelo foi estruturado para representar, de forma descentralizada, os elementos produtivos de uma célula fabril, incorporando lógica de decisão baseada em reforço inspirado em colônias de formigas. Essa abordagem permitiu simular o comportamento emergente de agentes autônomos diante de diferentes condições operacionais, alinhando-se aos princípios fundamentais da Indústria 4.0, como flexibilidade, descentralização, adaptabilidade e auto-organização.

Através da construção de quatro cenários de simulação distintos que variaram taxa de chegada de produtos, capacidade dos *buffers* e ocorrência de falhas operacionais, foi possível observar como pequenas alterações nos parâmetros impactam diretamente o desempenho do sistema. Os principais resultados obtidos incluem:

- No **Cenário 1**, a inserção simultânea de todos os produtos com *buffers* mínimos evidenciou gargalos severos, especialmente no recurso logístico (robô manipulador), resultando em baixíssima eficiência (<4%) e longos tempos de ciclo.
- O **Cenário 2** mostrou que o simples controle da taxa de chegada já melhora significativamente a fluidez do sistema, reduzindo os tempos médios de produção em mais de 50% e equilibrando a ocupação dos recursos.

- No **Cenário 3**, a ampliação dos *buffers* teve efeito limitado quando a entrada de produtos era controlada, indicando que, sob certas condições, o *buffer* não é o principal gargalo, mas pode funcionar como mecanismo de absorção de variações.
- O **Cenário 4**, com introdução de falhas temporárias, demonstrou a capacidade do sistema de reagir autonomamente, redirecionando produtos para máquinas alternativas. Entretanto, essa resiliência depende da existência de redundância funcional entre os recursos.
- Finalmente, o **Cenário 5** integrou todos os dados anteriores e permitiu concluir que o melhor desempenho geral foi obtido com taxa de chegada moderada e *buffers* ajustados, mesmo diante de falhas pontuais.

A lógica de reforço demonstrou eficácia na adaptação dinâmica dos agentes, permitindo o aprendizado coletivo sobre os recursos mais vantajosos ao longo do tempo. Além disso, a implementação da evaporação periódica dos reforços evitou o viés de longo prazo, mantendo o sistema sensível a mudanças de desempenho e disponibilidade.

Do ponto de vista metodológico, a plataforma NetLogo mostrou-se altamente adequada para este tipo de simulação, proporcionando fácil visualização, controle de variáveis e execução de testes em larga escala. Sua capacidade de representar interações micro a micro com impacto macro torna-a uma ferramenta eficaz para estudos de sistemas produtivos inteligentes e complexos.

Como continuidade deste projeto para trabalhos futuros, recomenda-se: testar outras configurações para a lógica de refoço, inserir critérios de priorização por tipo de produto ou urgência, explorar contextos com múltiplos manipuladores logísticos ou linhas paralelas, realizar comparações com abordagens centralizadas para quantificar os ganhos da descentralização.

Bibliografia

[1] H. Lasi, P. Fettke, H.-G. Kemper, T. Feld e M. Hoffmann, «Industry 4.0,» *Business & Information Systems Engineering*, vol. 6, pp. 239–242, ago. de 2014. DOI: 10.1007/s12599-014-0334-4.

[2] R. Drath e A. Horch, «Industrie 4.0: Hit or Hype? [Industry Forum],» *Industrial Electronics Magazine, IEEE*, vol. 8, pp. 56–58, jun. de 2014. DOI: 10.1109/MIE.2014.2312079.

[3] F. Tao, Q. Qi, L. Wang e A. Nee, «Digital Twins and Cyber–Physical Systems toward Smart Manufacturing and Industry 4.0: Correlation and Comparison,» *Engineering*, vol. 5, pp. 653–661, ago. de 2019. DOI: 10.1016/j.eng.2019.01.014.

[4] J. Lee, B. Bagheri e H.-A. Kao, «A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems,» *Manufacturing Letters*, vol. 3, pp. 18–23, 2015, ISSN: 2213-8463. DOI: https://doi.org/10.1016/j.mfglet.2014.12.001. URL: https://www.sciencedirect.com/science/article/pii/S221384631400025X.

[5] E. Lee, «Cyber Physical Systems: Design Challenges,» *Electrical Engineering and Computer Sciences*, pp. 363–369, jun. de 2008. DOI: 10.1109/ISORC.2008.25.

[6] O. Moerth-Teo, C. Emmanouilidis, N. Hafner e M. Schadler, «Cyber-Physical Systems for Performance Monitoring in Production Intralogistics,» *Computers & Industrial Engineering*, vol. 142, p. 106 333, fev. de 2020. DOI: 10.1016/j.cie.2020.106333.

[7] Y. Liu, Y. Peng, B. Wang, S. Yao e Z. Liu, «Review on cyber-physical systems,» *IEEE/CAA Journal of Automatica Sinica*, vol. 4, n.º 1, pp. 27–40, 2017. DOI: 10.1109/JAS.2017.7510349.

[8] D. G. Pivoto, L. F. de Almeida, R. da Rosa Righi, J. J. Rodrigues, A. B. Lugli e A. M. Alberti, «Cyber-physical systems architectures for industrial internet of things applications in Industry 4.0: A literature review,» *Journal of Manufacturing Systems*, vol. 58, pp. 176–192, 2021, ISSN: 0278-6125. DOI: https://doi.org/10.1016/j.jmsy.2020.11.017. URL: https://www.sciencedirect.com/science/article/pii/S0278612520302119.

[9] P. Leitão, «Agent-based distributed manufacturing control: A state-of-the-art survey,» *Engineering Applications of Artificial Intelligence*, vol. 22, n.º 7, pp. 979–991, 2009, Distributed Control of Production Systems, ISSN: 0952-1976. DOI: https://doi.org/10.1016/j.engappai.2008.09.005. URL: https://www.sciencedirect.com/science/article/pii/S0952197608001437.

[10] M. Wooldridge e N. R. Jennings, «Intelligent agents: theory and practice,» *The Knowledge Engineering Review*, vol. 10, n.º 2, pp. 115–152, 1995. DOI: 10.1017/S0269888900008122.

[11] J. Barbosa e P. Leitão, «Simulation of multi-agent manufacturing systems using Agent-Based Modelling platforms,» em *2011 9th IEEE International Conference on Industrial Informatics*, 2011, pp. 477–482. DOI: 10.1109/INDIN.2011.6034926.

[12] R. Allan, «Survey of Agent Based Modelling and Simulation Tools,» jan. de 2009.

[13] S. Abar, G. Theodoropoulos, P. Lemarinier e G. M. P. O'Hare, «Agent Based Modelling and Simulation tools: A review of the state-of-art software,» *Comput. Sci. Rev.*, vol. 24, pp. 13–33, 2017. DOI: 10.1016/J.COSREV.2017.03.001.

[14] J. Ozik, N. T. Collier, J. T. Murphy e M. North, «The ReLogo agent-based modeling language,» *2013 Winter Simulations Conference (WSC)*, pp. 1560–1568, 2013. DOI: 10.1109/WSC.2013.6721539.

[15] J. Barbosa e P. Leitão, «Modelling and simulating self-organizing agent-based manufacturing systems,» em *IECON 2010 - 36th Annual Conference on IEEE Industrial Electronics Society*, 2010, pp. 2702–2707. DOI: 10.1109/IECON.2010.5675113.

[16] E. A. Lee, «Cyber Physical Systems: Design Challenges,» em *2008 11th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC)*, 2008, pp. 363–369. DOI: 10.1109/ISORC.2008.25.

[17] A. Bessani, M. Correia, B. Quaresma, F. André e P. Sousa, «DepSky: dependable and secure storage in a cloud-of-clouds,» em *Proceedings of the Sixth Conference on Computer Systems*, sér. EuroSys '11, Salzburg, Austria: Association for Computing Machinery, 2011, pp. 31–46, ISBN: 9781450306348. DOI: 10.1145/1966445.1966449. URL: https://doi.org/10.1145/1966445.1966449.

[18] E. Ferraris, J. Vleugels, Y. Guo, D. Bourell, J. P. Kruth e B. Lauwers, «Shaping of engineering ceramics by electro, chemical and physical processes,» *CIRP Annals*, vol. 65, n.º 2, pp. 761–784, 2016, ISSN: 0007-8506. DOI: https://doi.org/10.1016/j.cirp.2016.06.001. URL: https://www.sciencedirect.com/science/article/pii/S0007850616301937.

[19] A. J. C. Trappey, C. V. Trappey, U. H. Govindarajan, J. J. Sun e A. C. Chuang, «A Review of Technology Standards and Patent Portfolios for Enabling Cyber-Physical Systems in Advanced Manufacturing,» *IEEE Access*, vol. 4, pp. 7356–7382, 2016. DOI: 10.1109/ACCESS.2016.2619360.

[20] J. Zhou, Z. Cao, X. Dong e A. V. Vasilakos, «Security and Privacy for Cloud-Based IoT: Challenges,» *IEEE Communications Magazine*, vol. 55, n.º 1, pp. 26–33, 2017. DOI: 10.1109/MCOM.2017.1600363CM.

[21] R. (Rajkumar, I. Lee, L. Sha e J. Stankovic, «Cyber-physical systems: the next computing revolution,» em *Proceedings of the 47th Design Automation Conference*, sér. DAC '10, Anaheim, California: Association for Computing Machinery, 2010, pp. 731–736, ISBN: 9781450300025. DOI: 10.1145/1837274.1837461. URL: https://doi.org/10.1145/1837274.1837461.

[22] H. Saha, N. Saha, R. Ghosh e S. Roychoudhury, «Recent trends in implementation of Internet of Things — A review,» out. de 2016, pp. 1–6. DOI: 10.1109/IEMCON.2016.7746345.

[23] L. Atzori, A. Iera e G. Morabito, «The Internet of Things: A Survey,» *Computer Networks*, pp. 2787–2805, out. de 2010. DOI: 10.1016/j.comnet.2010.05.010.

[24] Y. Lei, N. Li, L. Guo, N. Li, T. Yan e J. Lin, «Machinery health prognostics: A systematic review from data acquisition to RUL prediction,» *Mechanical Systems and Signal Processing*, vol. 104, pp. 799–836, mai. de 2018. DOI: 10.1016/j.ymssp.2017.11.016.

[25] M. Wooldridge, *An Introduction to MultiAgent Systems*. John wiley & sons, jun. de 2002, p. 348.

[26] P. Saraswat e S. Raj, «A Brief Review on Machine Learning and Its Various Techniques,» *International Journal of Innovative Research in Computer Science & Technology*, 2021. DOI: 10.55524/ijircst.2021.9.6.25.

[27] D. Stonko, L. Wang, M. Strother e L. Chambless, «Machine learning analyses can differentiate meningioma grade by features on magnetic resonance imaging,» *Neurosurgical Focus*, vol. 45, nov. de 2018. DOI: 10.3171/2018.8.FOCUS18191.

[28] J. Jovel e R. Greiner, «An Introduction to Machine Learning Approaches for Biomedical Research,» *Frontiers in Medicine*, vol. 8, 2021. DOI: 10.3389/fmed.2021.771607.

[29] K. Meena e S. Suriya, «A Survey on Supervised and Unsupervised Learning Techniques,» *Springer International Publishing*, pp. 627–644, mar. de 2020. DOI: 10.1007/978-3-030-24051-6_58.

[30] K. Clarke, «Cellular Automata and Agent-Based Models,» *Handbook of Regional Science*, pp. 1217–1233, jan. de 2014. DOI: 10.1007/978-3-642-23430-9_63.

[31] J. C. Thiele e V. Grimm, «NetLogo meets R: Linking agent-based models with a toolbox for their analysis,» *Environ. Model. Softw.*, vol. 25, pp. 972–974, 2010. DOI: 10.1016/j.envsoft.2010.02.008.

[32] I. Sakellariou, P. Kefalas e I. Stamatopoulou, «Enhancing NetLogo to Simulate BDI Communicating Agents,» em *Artificial Intelligence: Theories, Models and Applications*, J. Darzentas, G. A. Vouros, S. Vosinakis e A. Arnellos, eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 263–275, ISBN: 978-3-540-87881-0. DOI: 10.1007/978-3-540-87881-0_24.

[33] E. Sklar, «NetLogo, a Multi-agent Simulation Environment,» *Artificial Life*, vol. 13, pp. 303–311, 2007. DOI: 10.1162/artl.2007.13.3.303.