IX Simpósio de Engenharia Ferroviária
24 e 25 de junho de 2026 – Campinas/SP

UMA ALTERNATIVA EM CÓDIGO ABERTO PARA CRIAÇÃO E UTILIZAÇÃO DE MODELOS DE SIMULAÇÃO DE MALHAS FERROVIÁRIAS

Thales Augusto dos Santos[1,2]; Luiz Antônio Silveira Lopes[2]

[1] MRS Logística S.A. — Brasil
[2] Instituto Militar de Engenharia - IME — Brasil

RESUMO

Este artigo apresenta o desenvolvimento de uma aplicação em Python para simulação da circulação de trens em corredores ferroviários, com foco na avaliação de capacidade operacional. Dada a complexidade dos sistemas ferroviários, métodos analíticos mostram-se limitados, sendo a simulação de eventos discretos uma alternativa mais realista. O modelo proposto utiliza a biblioteca SimPy para representar a dinâmica de circulação, incluindo interações entre trens, pátios e eventos de falha na via permanente. A malha ferroviária é modelada por uma matriz que representa o estado das seções de bloqueio. A aplicação permite múltiplas simulações com variação estocástica dos tempos de viagem, gerando distribuições de capacidade. Em um estudo de caso, foi possível estimar a capacidade de um corredor ferroviário com maior precisão e baixo custo computacional. Os resultados demonstram que a ferramenta é uma alternativa eficiente e flexível em relação a softwares proprietários, permitindo análises mais robustas e integração com técnicas avançadas.

Palavras-chave: *Simulação; Python; Capacidade; Ferrovia; SimPy.*

1. INTRODUÇÃO

A operação ferroviária só pode ocorrer quando há a interação segura e eficiente de uma série de sistemas que permitem que trens transportem pessoas e cargas conforme os níveis de serviço esperados. Para que esses deslocamentos ocorram, vagões, locomotivas, a via permanente, a sinalização e infraestrutura – para mencionar os mais comuns – sofrem e exercem restrições que vão definir como uma ferrovia pode ser operada.

Devido a essas interações, sistemas ferroviários são essencialmente complexos e sua capacidade não pode ser tomada de forma trivial. Apesar de aparentemente intuitivo como o "número máximo de trens ou vagões que circulam em certo período", esta referência é normalmente inviável, teórica, pois desconsidera outros fatores que qualificam a capacidade, sendo necessário aplicar fatores de correção, que a colocam em níveis práticos (STOK, 2008).

De acordo com a União Internacional dos Caminhos de Ferro – UIC (2004), a capacidade prática é uma métrica subjetiva, porque depende do modo como a infraestrutura da malha ferroviária é utilizada sob a ótica do número de trens em circulação, da velocidade média, da heterogeneidade e da estabilidade do tráfego.

Ao longo das viagens, os trens também interagem com outros subsistemas importantes como pátios e, nessas instalações, ocorrem processos como carga e descarga de vagões, trocas de maquinistas, cruzamento e ultrapassagem de trens, manobras de vagões e serviços de manutenção. Mesmo aparentemente previsíveis, essas atividades sofrem oscilações que podem se propagar para além daqueles trens em atividade no pátio

(MARINOV et al., 2013). A forma como recursos estáticos como linhas e aparelhos de mudança dia via (AMV) interagem com os elementos dinâmicos define como a capacidade instalada é utilizada, ou seja, as regras e modelos operacionais influem diretamente na quantidade de trens e vagões que pátios e corredores ferroviários podem processar em um dado período (ASSAD, 1980; BOYSEN, 2012).

Para avaliar a capacidade operacional de uma ferrovia, três metodologias podem ser consideradas (STOK, 2008):

1. Métodos Analíticos: Aplicam formulações matemáticas em que a variável de resposta (capacidade) é explicitamente obtida em função de outras variáveis conhecidas. No Brasil, o método mais comum é o de Colson, sendo inclusive usado para declaração de capacidade das concessões ferroviárias regulamentadas pela Agência Nacional de Transportes Terrestres (ANTT);

2. Métodos de Otimização: Baseiam-se em algoritmos e heurísticas para encontrar soluções ótimas ou satisfatórias para maximizar a capacidade segundo os recursos disponíveis. Uma aplicação conhecida é o método UIC de compressão da grade horária de trens (UIC 406 R) que visa o sequenciamento dos trens planejados que melhor utilize as horas disponíveis de operação;

3. Métodos de Simulação: Consistem na elaboração de modelos que emulam o sistema real incorporando as características mais relevantes para a avaliação. Representam a forma que permite maior aproximação da realidade operacional e, por isso também, podem ser os de implementação e validação mais difíceis conforme o nível de precisão exigido.

As ferramentas usadas para simular a circulação de trens têm que ser capazes de representar as oscilações do mundo real, conferindo aproximação com a realidade que é obtida por meio de algoritmos de licenciamento de trens que emulam regras (políticas) da operação ferroviária (SIEFER, 2008). A solução eficiente de conflitos de circulação, especialmente em linhas singelas possui um desafio crítico: eventos de *deadlock*, que são retratados na literatura e representam situações em que dois trens em sentidos opostos ficam em um impasse de ocupação da mesma seção de bloqueio.

Existem muitas opções de softwares com diferentes níveis de licença de usuário que são usados para elaboração de modelos de simulação por eventos discretos e contínua. Essas aplicações diferenciam-se também pela abrangência de sistemas que podem ser tratados; vão desde modelos generalistas adaptados ao tipo de problema estudado, até modelos especializados dedicados a operações ferroviárias. Estes são capazes de simular trens conforme o tipo de locomotivas e vagões, emular os efeitos do perfil do relevo, do tipo de sinalização e das restrições de via permanente. As ferramentas de simulação mais populares e as especializadas em ferrovia são softwares proprietários, requisitam investimentos constantes para sua atualização e, normalmente, têm alto custo de aquisição, o que explica em parte, o restrito número de desenvolvedores e usuários de modelos de simulação.

Em adição, por terem muitos parâmetros de configuração e especificidades, modelos elaborados em softwares de simulação proprietários comumente exigem maior prazo para desenvolvimento, o que muitas vezes é uma restrição crítica dependendo do estudo ou situação avaliada.

Posto este cenário e tendo em vista a popularização acelerada de linguagens de programação como Python, vislumbrou-se a possibilidade de desenvolver uma ferramenta de simulação de corredores ferroviários que usa recursos desenvolvidos em linguagem Python para simular eventos discretos. Desta forma, foi desenvolvida uma aplicação capaz de simular o processo de circulação de trens, aplicando regras de licenciamento considerando:

- Informações da infraestrutura de malha como a quantidade de pátios e linhas disponíveis;

- Quantidade, sentido e tempos de circulação dos trens;

- Frequência e duração de interrupções da circulação para manutenções na via permanente;

- Múltiplos cenários de simulação variando aleatoriamente os tempos de circulação dentro de limites pré-definidos e obter uma amostra de contagem de trens processados em cada rodada de simulação.

Assim, obteve-se uma alternativa ao uso de ferramentas mais custosas de mercado, sem todos os recursos de análise que oferecem, mas que ainda assim, permite análises e respostas mais aprofundadas do que métodos determinísticos ou outros que, pelo escopo, não abordam a dinâmica de circulação de trens.

A aplicação apresentada neste trabalho já foi utilizada em diversos estudos de ampliação de capacidade instalada de uma operadora ferroviária, possibilitando a comparação de alternativas de investimentos com resultados obtidos mais rapidamente do que quando usados os softwares de simulação convencionais. Além disso, dado o número de usuários de linguagem Python expressivamente maior, a aplicação será mais exposta a revisões e melhorias implementadas conforme necessidades específicas.

2. DESENVOLVIMENTO

Para o desenvolvimento da aplicação, foi utilizado um computador Intel Core i7, 16 GB de memória e sistema operacional Windows 11; para elaboração e execução de códigos em linguagem Python, utilizou-se o Visual Studio Code (VS Code).

No ambiente de desenvolvimento executado no VS Code, foram instaladas as bibliotecas: SimPy, Itertools, Random, CSV e Pandas.

O SimPy, uma biblioteca Python sob licença do MIT (Massachusetts Institute of Technology), é o cerne da aplicação, pois permite a criação de um ambiente de simulação de eventos discretos, oferecendo recursos compartilhados para modelar pontos de congestão de capacidade (TEAM SIMPY, 2023).

Uma vez definidos os meios de desenvolvimento da aplicação, foi elaborado o modelo conceitual do problema, explicado a seguir.

2.1. Escopo

A aplicação foi proposta para simular o processo de circulação de trens em um dado corredor em linha singela ou dupla, com pátios de origem, intermediários e de destino, sendo possível escolher o modelo operacional (fluxo bidirecional ou unidirecional de “A” para “B” ou unidirecional de “B” para “A”) e inserir eventos de falhas de via que causassem a interrupção de tráfego por um dado intervalo.

Como não foram construídos módulos para calcular a velocidade dos trens em função do perfil da via, do peso das próprias composições ferroviárias e do esforço trator disponível, ou seja, pela diferença entre forças de tração e resistências normais e acidentais, foram definidos parâmetros de tempo de circulação entre pátios e seções de bloqueio conforme o sentido de circulação. Tais referências podem ser obtidas em registros de tempos de viagens realizadas ou de simuladores de condução de trens. Ressalta-se que para estes parâmetros, devem ser considerados tempos sem eventos de espera por qualquer razão, pois estes efeitos são estimados a partir dos eventos de licenciamento de trens do modelo de simulação proposto.

Além disso, devem ser informados os pátios e seções de linha singela ou dupla a partir de um vetor que represente a malha ferroviária em que cada linha seja o local da malha e cada coluna seja a quantidade de seções de bloqueio que existem naquela posição.

Seja a matriz M ∈ $\mathbb{Z}^{n\times k}$ na qual cada elemento $m_{i,j}$ indica o estado da seção de bloqueio j na posição i, podendo assumir valores que representam disponibilidade ou ocupação por trens em diferentes sentidos.

$$M = \begin{bmatrix} m_{1,1} & m_{1,2} & \cdots & m_{1,j} \\ m_{2,1} & m_{2,2} & & m_{2,j} \\ \vdots & & \ddots & \vdots \\ m_{i,1} & m_{i,2} & \cdots & m_{i,j} \end{bmatrix} \quad (1)$$

Em que,

$$m_{i,j} = \begin{cases} 0, se\ a\ seção\ estiver\ livre \\ 1, seção\ ocupada\ no\ sentido\ A \rightarrow B \\ -1, ocupada\ no\ sentido\ B \rightarrow A \end{cases}$$

2.2. Algoritmo de Licenciamento (Despacho) de Trens

O algoritmo concebido deveria ser capaz de resolver o seguinte problema: fazer um trem viajando de “A” para “B” ou de “B” para “A” ocupar as seções de bloqueio à frente, esperando quando necessário, em um pátio de cruzamento, a passagem de um trem no sentido oposto. A Figura 1 apresenta o pseudocódigo e a Figura 2 o fluxograma de funcionamento.

```
ALGORITMO: Simulação de Circulação de Trem (Eventos Discretos)

ENTRADAS:
MODEL_OP, MALHA, SB_cap, T_RUN_IMP, T_RUN_EXP

INÍCIO

SE MODEL_OP = 2 ENTÃO
   inicio ← aleatório(0,1)
SENÃO
   inicio ← MODEL_OP
FIM SE

SE inicio = 0 ENTÃO
   inicio ← 0
   sentido ← +1
   destino ← tamanho(MALHA)
SENÃO
   inicio ← tamanho(MALHA) - 1
   sentido ← -1
   destino ← -1
FIM SE

lista_ocupacao ← ∅
tempo_inicio ← tempo atual
posicao ← inicio

PARA i ← inicio ATÉ destino PASSO sentido FAÇA

   SE i = inicio ENTÃO
      OCUPAR SB_cap[i]
      MALHA[i] ← sentido
      SE sentido = +1 ENTÃO
         AGUARDAR T_RUN_IMP[posicao]
      SENÃO
         AGUARDAR T_RUN_EXP[posicao]
      FIM SE
   FIM SE

   SE seção singela E não inicio ENTÃO
      CONTINUAR
   FIM SE

   SE seção pátio E não inicio ENTÃO
      ENQUANTO ocupado FAÇA
         AGUARDAR 0.1
      FIM ENQUANTO

      OCUPAR pátio e seção anterior
      AGUARDAR 0.1
      LIBERAR seção anterior

      SE sentido = +1 ENTÃO
         AGUARDAR T_RUN_IMP[posicao]
      SENÃO
         AGUARDAR T_RUN_EXP[posicao]
      FIM SE

      posicao ← posicao + sentido
      LIBERAR linha singela
   FIM SE

   SE i = final ENTÃO
      CALCULAR tempo total
      REGISTRAR resultados
   FIM SE
```

Figura 1. Pseudocódigo da Classe Trem Usada no Ambiente de Simulação.

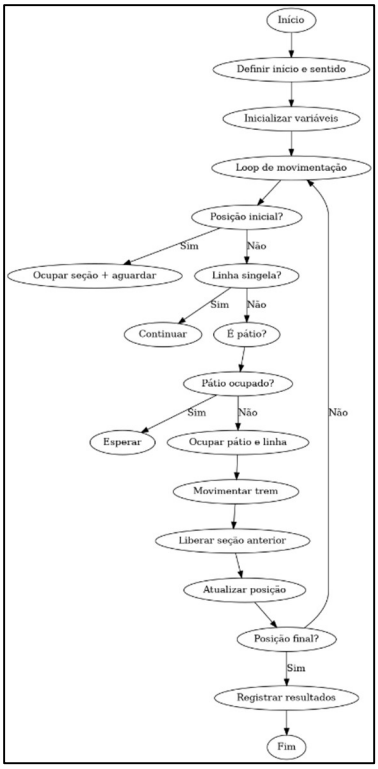

Figura 2. Fluxograma das Etapas de Funcionamento do Algoritmo de Licenciamento de Trens

2.3. Criação dos Eventos de Manutenção da Via Permanente

Dentro do ambiente de simulação por eventos discretos, além da circulação dos trens, existem os eventos de manutenções da via permanente que, ao ocorrerem, bloqueiam uma

da dada seção de bloqueio. No modelo proposto, existe a previsão que esses eventos possam ocorrer de forma regular (como se fossem manutenções programadas) ou aleatoriamente.

Para isso, foi criado um método que age dentro do ambiente de simulação gerando eventos de manutenção conforme parâmetros definidos pelo usuário. Os eventos de tempo entre falhas e tempo para reparo podem ser configurados com valores constantes ou a partir de distribuições de probabilidade geradas pela biblioteca Random da linguagem Python. Além disso, podem ser definidos de forma fixa ou aleatória os locais onde as falhas ocorrem.

Isso permite que sejam representados eventos que podem ocorrer em toda malha ou condições específicas, em que um serviço relevante de manutenção ocorre em um dado trecho da ferrovia. O processo é demonstrado na Figura 3 a seguir.

```
ALGORITMO: GERAÇÃO DE BLACKOUTS (FALHAS NA MALHA)

INÍCIO

i ← 0

ENQUANTO VERDADEIRO FAÇA

    mtbf ← MTBF
    mttr ← MTTR

    loc_blackout ← aleatório(0, tamanho(MALHA) - 1)

    OCUPAR SB_cap[loc_blackout]

    linha ← aleatório(0,1)

    SE MALHA[loc_blackout] for singela ENTÃO
        linha ← 0
    FIM SE

    SE linha = 0 ENTÃO
        MALHA[loc_blackout][linha] ← 1
    SENÃO
        MALHA[loc_blackout][linha] ← -1
    FIM SE

    REGISTRAR início do blackout

    AGUARDAR mttr

    LIBERAR SB_cap[loc_blackout]
    MALHA[loc_blackout][linha] ← 0

    REGISTRAR fim do blackout

    AGUARDAR (mtbf - mttr)

    i ← i + 1

FIM ENQUANTO

FIM
```

Figura 3. Pseudocódigo do Método de Criação de Falhas da Via Permanente

2.4. Execução do Modelo em Múltiplas Simulações

Esta funcionalidade foi elaborada para que, a partir de um número de iterações (replicações) definidas pelo usuário, o modelo rodasse combinações aleatórias dos dados de entrada para obter uma amostra da contagem de trens processados em cada simulação. Assim, a aplicação permitiu que a capacidade fosse observada a partir de uma distribuição de resultados, mostrando os valores mais prováveis, tornando a análise mais realista.

Em linhas gerais, a aplicação executa um loop dentro do número de replicações definido pelo usuário, cria instâncias da classe trem e do método de falhas e executa simulações dentro do ambiente SimPy.

```
MODELO COMPLETO: SIMULAÇÃO FERROVIÁRIA COM BLACKOUTS

INÍCIO

1. INICIALIZAÇÃO
- GHT ← []
- BLKT ← []
- EXPERIMENT_LOGS ← []
- MODEL_OP definido
- NUMBER_RUNS definido
- NUM_REPS ← 0

2. LOOP DE REPLICAÇÕES
PARA m ← 1 ATÉ NUMBER_RUNS FAÇA

   2.1 FATOR ALEATÓRIO
   - FATOR_ALEATORIO ← triangular(0.9, 1.0, 1.2)
   - T_RUN_EXP ← T_RUN_EXP_ORIG × FATOR_ALEATORIO
   - T_RUN_IMP ← T_RUN_IMP_ORIG × FATOR_ALEATORIO

   2.2 CONTROLE
   - NUM_REPS ← NUM_REPS + 1
   - Registrar replicação
   - Resetar MALHA

   2.3 AMBIENTE SIMPY
   - Criar env
   - Inicializar SB_cap como containers

   2.4 PROCESSOS PARALELOS

      PROCESSO TREM:
      - Definir início e sentido
      - Inicializar variáveis
      - LOOP:
         * Verificar seção
         * Ocupar/liberar recursos
         * Aguardar tempos
         * Atualizar posição
         * Registrar logs
      - Finalizar trem

      PROCESSO BLACKOUT:
      - LOOP INFINITO:
         * Sortear localização
         * Ocupar seção
         * Definir linha
         * Registrar início
         * Aguardar MTTR
         * Liberar seção
         * Registrar fim
         * Aguardar MTBF - MTTR

   2.5 EXECUÇÃO
   - env.run(até tempo final)

FIM PARA

FIM
```

Figura 4. Pseudocódigo da Integração de Métodos e Execução Completa da Aplicação

Desta forma, podem ser configurados:

- O corredor ferroviário objeto de estudo;

- Os tempos de circulação a partir de dados reais ou simulados;

- A quantidade de pátios ferroviários existentes ou novos (testes de expansão de capacidade);

- A frequência de eventos de manutenção;

- A demanda de transporte em termos da frequência de trens a ser testada;

- A quantidade e duração dos experimentos.

3. RESULTADOS E DISCUSSÕES

Nesta seção, serão apresentados e discutidos os resultados da utilização da aplicação para estudo de capacidade de um corredor ferroviário operado por uma ferrovia de cargas do Brasil.

O objetivo era estimar a capacidade a partir da implantação de novos pátios de cruzamento. Como dados de entrada, foram usados tempos reais de circulação de trens de carga vazios e carregados entre os pátios e seções de bloqueio da malha.

Figura 5. Fragmento de Script Python em VS Code dos Parâmetros do Modelo

Mesmo sendo um corredor de mais de 100 km de extensão, havia poucos pátios de cruzamento, o que restringe a capacidade de utilização da malha, ou seja, a quantidade de trens que podem ser processados ao longo de um dia de operação. Neste estudo de caso, foram considerados 1440 minutos para cada rodada de simulação, descartando-se os resultados durante o processo de carregamento da malha (*warm-up*) e finalização da simulação (*cooldown*).

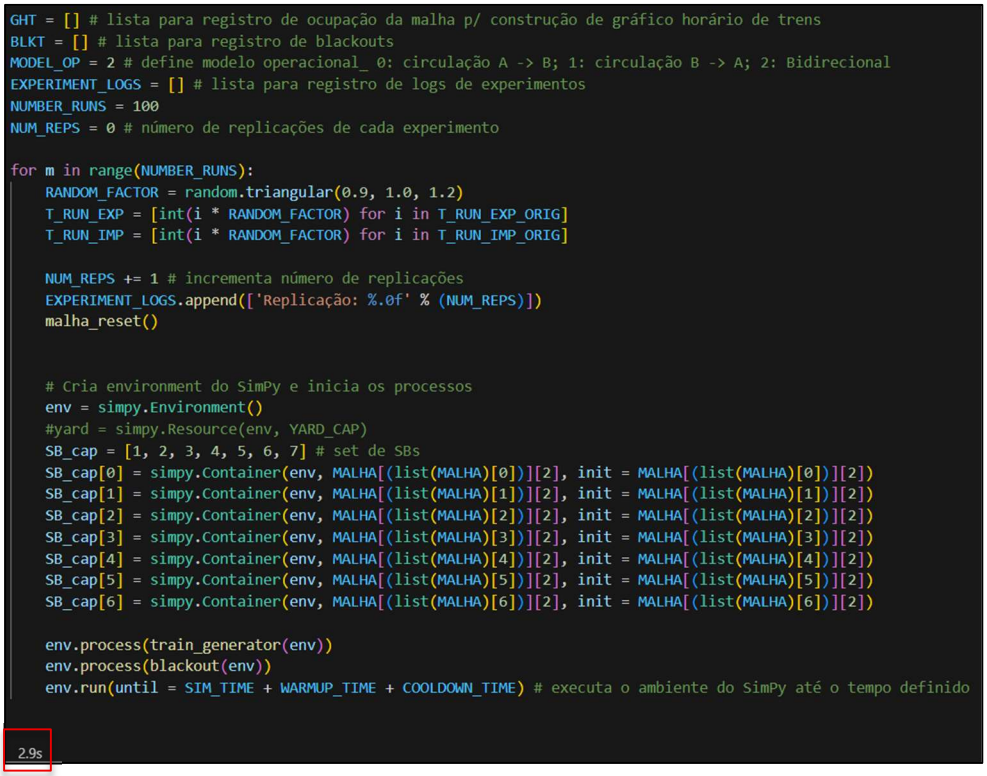

Figura 6. Fragmento de Script Python em VS Code que Controla a Execução do Modelo. São executadas "m" simulações considerando 7 seções de bloqueio da malha que são tratadas como recursos (*containers*) do SimPy. A cada simulação, os tempos de circulação T_RUN_EXP e T_RUN_IMP são multiplicados por um fator aleatório que oscila dentro de limites conforme a base de dados reais.

Conforme mostrado na Figura 6, foram executadas 100 rodadas de simulação em aproximadamente 2,9 segundos de tempo computacional. A possibilidade de obter análises de capacidade de um corredor ferroviário, retratando as interações dinâmicas entre trens considerando as características reais de circulação é uma grande vantagem em

relação a métodos mais tradicionais, analíticos por exemplo; com tempo de execução sensivelmente baixo.

Para a visualização dos resultados foi utilizada a biblioteca Plotly, que é um *open source* que permite a utilização de gráficos interativos (PLOTLY TECHNOLOGIES INC., 2026). A Figura 7 permite a visualização de como os trens foram licenciados durante a replicação 91, indicando a quantidade de trens (quantidade de linhas projetadas) e onde e como os cruzamentos ocorreram. Com isso, pode-se avaliar como a infraestrutura de malha é utilizada e onde ocorrem as maiores esperas por licenciamento. É possível também identificar onde as falhas de via ocorreram (indicadas pelo marcador "blackout") e os trens que foram impactados.

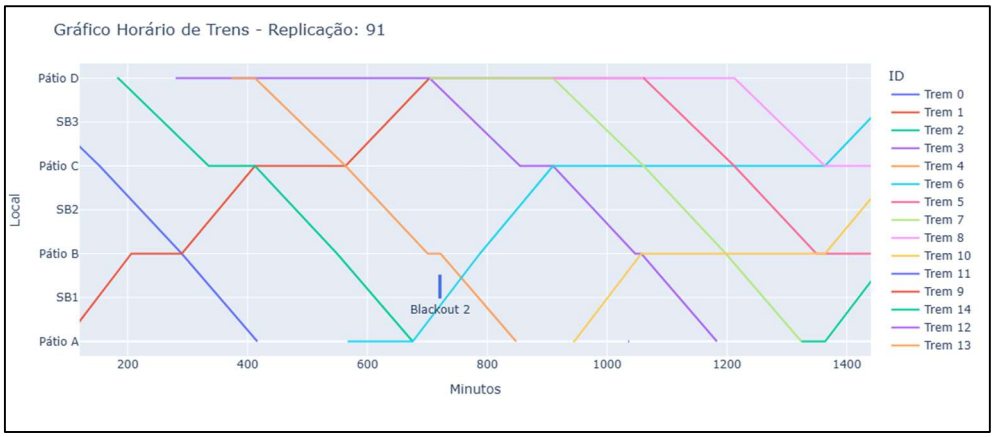

Figura 7. Gráfico Horário de Trens – GHT considerando a implantação dos pátios "B" e "C". O GHT é obtido a partir dos registros de simulação considerando os horários de entrada e partida dos trens em cada seção de bloqueio (Gerada no módulo plotly.express).

A Figura 8 mostra a distribuição dos resultados de simulação em termos da contagem de trens processados ao longo de cada simulação de vinte e quatro horas. Fica evidente que 74 de um total de 100 cenários, apresentam contagem entre 3,5 e 4,0 pares de trens/dia, ou seja, é muito provável que este corredor ferroviário, com esta quantidade de pátios de cruzamento e com o padrão de falhas típico, tem capacidade próxima a 4 pares de trens/dia, sendo a média de 3,6 pares de trens/dia.

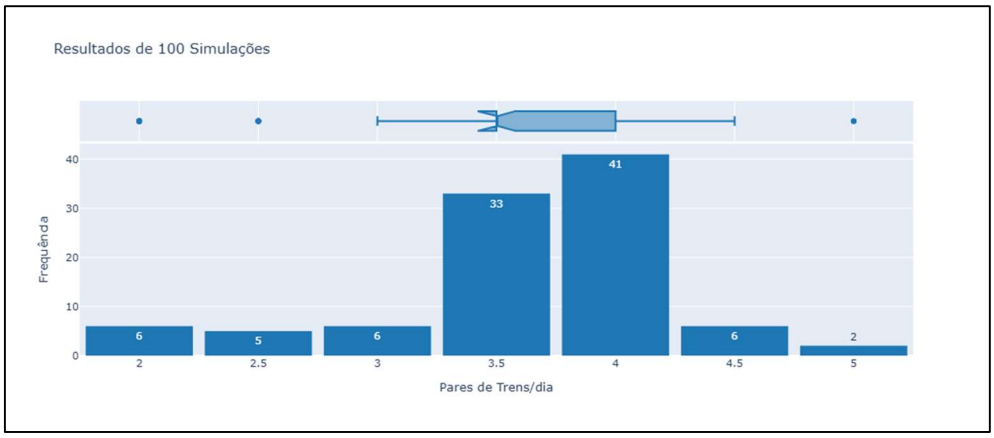

Figura 8. Histograma da Contagem de Trens ao Longo de 100 Rodadas de Simulação (Gerada no módulo plotly.express).

Desta forma, foi possível aplicar um método mais robusto do que métodos analíticos com estimativas mais precisas da capacidade esperada com a implantação de novos pátios de cruzamento.

4. CONCLUSÃO

O objetivo de construir uma aplicação para elaboração de modelos de simulação em código-fonte aberto, independente de ferramentas tradicionais de mercado foi atendido. O mais importante é que a solução se como uma alternativa satisfatória e adaptável em relação aos softwares tradicionais de simulação.

Dada a larga adoção da linguagem Python nas mais diferentes áreas de conhecimento, a aplicação se coloca em um ponto de vantagem por se beneficiar das possibilidades de integração com bibliotecas *open source,* além de melhorias implementadas pelos próprios usuários conforme necessidade. Por exemplo, o modelo de simulação pode ser usado como módulo de um script mais amplo que integre métodos de Otimização e Aprendizado de Máquina. Tendo em vista o acelerado crescimento de modelos de Aprendizado por Reforço, por exemplo, esta aplicação pode ser adotada em modelos que busquem treinar agentes na busca por melhores estratégias de licenciamento de trens.

Este tipo de integração e flexibilidade é uma grande vantagem em relação aos softwares de simulação disponíveis que normalmente têm lançamento de atualizações mais demorados, rigidez para customizações e integrações com linguagens de programação mais comuns e acessíveis.

Para oportunidades futuras, além das mencionadas acima, espera-se a elaboração de módulos para cálculo das velocidades dos trens em função do peso das composições, perfil da via e esforço trator das locomotivas, permitindo que a aplicação se integre com soluções que contenham calculadoras de performance de trens.

5. AGRADECIMENTOS

Gostaria de manifestar os sinceros agradecimentos à MRS Logística por proporcionar os meios e oportunidades de assimilar, gerar e aplicar conhecimentos em projetos de relevância com a colaboração de colegas ferroviários, cuja experiência é destacada nos cenários nacional e internacional. Agradecimentos também ao Instituto Militar de Engenharia, na figura de seu corpo docente, em especial ao Professor D.Sc. Luiz Antônio Silveira Lopes, que tem dispendido tempo em orientações valiosas por mais de uma década.

Por fim, agradeço à comissão de professores e profissionais que organizam o Simpósio de Engenharia, que já é uma referência na promoção e desenvolvimento do conhecimento ferroviário, pela oportunidade de compartilhar este humilde trabalho.

6. REFERÊNCIAS

ASSAD, A. A. Models for rail transportation. *Transportation Research Part A*, v. 14A, p. 205–220, 1980.

BOYSEN, N. *et al.* Shunting yard operations: theoretical aspects and applications. *European Journal of Operational Research*, v. 220, n. 1, p. 1–14, 2012.

MARINOV, M.; ŞAHIN, I.; RICCI, S.; FRANKLIN, G. Railway operations, timetabling and control. *Research in Transportation Economics*, v. 41, 2013, p. 59–75.

MICROSOFT. *Visual Studio Code*. 2024. Disponível em: https://code.visualstudio.com/. Acesso em: 22 mar. 2026.

PLOTLY TECHNOLOGIES INC. *Plotly: open-source graphing library*. 2024. Disponível em: https://plotly.com/. Acesso em: 22 mar. 2026.

SIEFER, T. Simulation. In: HANSEN, I. A.; PACHL, J. *Railway timetable and traffic*. Hamburg: Eurailpress, 2008. p. 155–169.

STOK, R. *Estimation of railway capacity consumption using stochastic differential equations*. Trieste: Università di Trieste, 2008.

TEAM SIMPY. *SimPy documentation*. 2023. Disponível em: https://simpy.readthedocs.io/. Acesso em: 22 mar. 2026.

UIC – UNIÃO INTERNACIONAL DOS CAMINHOS DE FERRO. *UIC leaflet 406 R: capacity*. Paris: UIC, 2004b.

AN OPEN-SOURCE ALTERNATIVE FOR CREATING APPLIED RAILWAY NETWORK SIMULATION MODELS

ABSTRACT

This paper presents the development of a Python-based application for simulating train operations in railway corridors, with a focus on evaluating operational capacity. Given the complexity of railway systems, analytical methods prove to be limited, making discrete-event simulation a more realistic alternative. The proposed model uses the SimPy library to represent traffic dynamics, including interactions among trains, yards, and track failure events. The railway network is modeled as a matrix representing the state of block sections. The application enables multiple simulations with stochastic variation in travel times, generating capacity distributions. In a case study, it was possible to estimate the capacity of a railway corridor with greater accuracy and low computational cost. The results demonstrate that the tool is an efficient and flexible alternative to proprietary software, enabling more robust analyses and integration with advanced techniques.

Keywords: *Simulation; Python; Capacity; Railway; SimPy.*