VINÍCIUS LEOBET BREGOLI

DESENVOLVIMENTO DE GERADOR DE MODELOS DE SIMULAÇÃO PARA TOMADA DE DECISÃO NO CURTO PRAZO USANDO PROCESS MINING

Curitiba

2025

VINÍCIUS LEOBET BREGOLI

DESENVOLVIMENTO DE GERADOR DE MODELOS DE SIMULAÇÃO PARA TOMADA DE DECISÃO NO CURTO PRAZO USANDO PROCESS MINING

Trabalho de Conclusão de Curso apresentado ao Curso de Engenharia de Computação da Pontifícia Universidade Católica do Paraná como requisito parcial para obtenção do grau de Bacharel em Engenharia de Computação.

PONTIFÍCIA UNIVERSIDADE CATÓLICA DO PARANÁ – PUCPR

ESCOLA POLITÉCNICA

CURSO DE ENGENHARIA DE COMPUTAÇÃO

Orientador: Prof. Dr. Edson Emílio Scalabrin

Curitiba

2025

Resumo

A tomada de decisão no curto prazo em ambientes complexos, como centros cirúrgicos, exige ferramentas capazes de integrar dados históricos e informações em tempo real. Nesse contexto, a mineração de processos (Process Mining - PM) e a simulação computacional emergem como tecnologias para apoiar gestores na alocação eficiente de recursos, detecção de desvios e previsão de cenários. Este trabalho propõe o desenvolvimento de um gerador de modelos de simulação baseado em PM, com foco em reduzir o esforço humano e o tempo necessário para a construção de modelos. O gerador utiliza logs de eventos como fonte de dados para criar automaticamente modelos de simulação, permitindo avaliar alternativas de curto prazo e apoiar a tomada de decisão operacional. O estudo se apoia no framework PM4SOS, estendido e adaptado para o domínio hospitalar, e integra métodos multicritério e técnicas de otimização. O resultado esperado é a disponibilização de um protótipo que auxilie gestores a analisar filas, prever ocupação de salas, otimizar agendas e reduzir gargalos, contribuindo para maior eficiência operacional, redução de custos e melhor qualidade no atendimento.

Palavras-chave: Mineração de Processos, Simulação Computacional, Tomada de Decisão, Otimização, Agendamento.

Abstract

Short-term decision-making in complex environments, such as surgical centers, requires tools capable of integrating historical data and real-time information. In this context, Process Mining (PM) and computer simulation emerge as key technologies to support managers in efficient resource allocation, deviation detection, and scenario prediction. This work proposes the development of a simulation model generator based on PM, focusing on reducing human effort and the time required to build models. The generator uses event logs as a data source to automatically create simulation models, allowing the evaluation of short-term alternatives and supporting operational decision-making. The study is based on the PM4SOS framework, extended and adapted to the hospital domain, and integrates multicriteria methods and optimization techniques. The expected outcome is a prototype that helps managers analyze queues, predict room occupancy, optimize schedules, and reduce bottlenecks, contributing to greater operational efficiency, cost reduction, and improved service quality.

Keywords: Process Mining, Computer Simulation, Decision-Making, Optimization, Scheduling.

Lista de ilustrações

Lista de tabelas

Sumário

1 Introdução

Em organizações modernas, a complexidade operacional dos processos exigem decisões cada vez mais rápidas baseado em dados. Setores como saúde, manufatura, logística, mineração e serviços compartilham desafios semelhantes: alocação eficiente de recursos, detecção de gargalos, redução de custos e melhoria contínua de desempenho. Em todos esses contextos, as decisões de curto prazo — aquelas que precisam ser tomadas em horizontes de horas ou dias — exercem impacto direto na produtividade, na utilização de recursos e na qualidade do serviço prestado.

Apesar da ampla digitalização de processos e do grande volume de dados coletados em sistemas corporativos, a conversão dessas informações em conhecimento útil para a tomada de decisão ainda depende, em grande parte, da análise manual e da experiência de gestores e especialistas. Esse processo, além de demorado, está sujeito a vieses cognitivos e erros humanos, o que limita a capacidade das organizações de reagir rapidamente a mudanças operacionais.

A mineração de processos (*Process Mining* – PM) surge como uma abordagem capaz de extrair, a partir de logs de eventos, informações estruturadas sobre o comportamento real dos processos. Essa técnica permite descobrir modelos de processo, identificar gargalos e analisar conformidade com base em dados reais de execução. Trabalhos recentes demonstram seu potencial em contextos industriais complexos, como sistemas logísticos internos (WUENNENBERG; WEGERICH; FOTTNER, 2023) e processos de mineração subterrânea (BRZYCHCZY; ŻUBER; AALST, 2024), nos quais a análise de eventos de sensores e sistemas de controle tem permitido identificar padrões de desempenho e oportunidades de otimização.

Entretanto, a mineração de processos, por si só, fornece uma visão descritiva do comportamento passado, não sendo suficiente para antecipar cenários futuros ou testar alternativas operacionais. Nesse sentido, a integração com a simulação computacional oferece uma perspectiva complementar, permitindo representar dinamicamente o sistema e avaliar o impacto de diferentes estratégias antes de sua aplicação real (MARUŞTER; BEEST, 2009). Modelos de simulação baseados em dados de execução viabilizam a previsão de tempos de espera, taxas de ocupação e desempenho global do processo, contribuindo para decisões mais assertivas.

Ainda assim, a construção manual de modelos de simulação é uma tarefa intensiva, que exige conhecimento técnico detalhado sobre o processo e considerável esforço de modelagem — fatores que inviabilizam seu uso em situações que demandam resposta rápida. Pesquisas recentes têm buscado reduzir essa lacuna, utilizando mineração de processos para automatizar a geração de modelos de simulação, como proposto por (FERRONATO, 2022) no framework PM4SOS. Essa abordagem demonstrou ser eficaz para o suporte operacional em centros cirúrgicos, combinando mineração de logs, simulação e otimização multicritério. De forma semelhante, estudos na área de logística e engenharia de minas têm evidenciado os benefícios de abordagens híbridas entre

mineração de dados e simulação discreta, com o objetivo de otimizar fluxos físicos e prever anomalias operacionais (MENG et al., 2024).

Neste contexto, o presente trabalho propõe o desenvolvimento de um **gerador de modelos de simulação para tomada de decisão no curto prazo**, fundamentado na integração entre *Process Mining* e *Simulação de Eventos Discretos*. O objetivo é automatizar a criação de modelos a partir de dados reais de execução, reduzindo o esforço cognitivo do analista e possibilitando a análise preditiva de cenários com base em evidências empíricas. Este projeto busca generalizar o conceito, tornando o gerador aplicável a diferentes domínios organizacionais que disponham de registros de eventos estruturados, utilizando o setor hospitalar como estudo de caso para validação prática.

Com essa proposta, pretende-se preencher lacunas identificadas na literatura e na prática organizacional, especialmente no que diz respeito à integração automatizada entre dados históricos e modelos de simulação. A pesquisa almeja demonstrar que a combinação de mineração de processos, modelagem automatizada e simulação orientada por dados constitui uma ferramenta eficaz para apoiar a **tomada de decisão operacional de curto prazo**, promovendo análises rápidas, reprodutíveis e sustentadas por dados reais.

1.1 Contexto e Problema

A crescente complexidade dos ambientes organizacionais e o dinamismo dos processos produtivos exigem das instituições uma capacidade contínua de adaptação e resposta rápida a mudanças operacionais. Setores como saúde, manufatura, logística, mineração e serviços compartilham desafios recorrentes: alocar recursos de forma eficiente, detectar gargalos, reduzir custos e promover a melhoria contínua de desempenho. Em todos esses contextos, as decisões de curto prazo — aquelas que precisam ser tomadas em intervalos de horas ou dias — exercem influência direta sobre a produtividade, a eficiência operacional e a qualidade do serviço prestado.

Com a intensificação da digitalização e a adoção de sistemas de informação integrados, grandes volumes de dados passaram a ser gerados em tempo real. Contudo, a transformação desses dados em conhecimento útil para apoiar decisões ainda depende, em grande medida, da experiência humana e de análises manuais. Esse processo é sujeito a vieses cognitivos e à limitação do tempo de resposta, o que reduz a capacidade organizacional de reagir a variações de demanda, atrasos ou falhas no fluxo produtivo.

Nesse cenário, a *mineração de processos* (*Process Mining* – PM) tem se consolidado como uma abordagem poderosa para extrair, a partir de logs de eventos, informações estruturadas sobre o comportamento real dos processos. A técnica permite descobrir modelos de processo, identificar desvios de conformidade e mensurar indicadores de desempenho com base em dados reais de execução. Aplicações recentes demonstram seu potencial em domínios industriais

complexos, como sistemas logísticos internos (WUENNENBERG; WEGERICH; FOTTNER, 2023) e processos de mineração subterrânea (BRZYCHCZY; ŻUBER; AALST, 2024), nos quais a análise de dados sensoriais e transacionais tem contribuído para o diagnóstico e a otimização de operações.

Apesar disso, a mineração de processos oferece predominantemente uma visão descritiva e diagnóstica — concentrada no passado e no presente das operações. Por não possuir mecanismos preditivos, ela é limitada quando se busca antecipar cenários futuros, testar alternativas operacionais ou estimar o impacto de decisões sob diferentes condições de carga ou recursos.

Por outro lado, a *simulação de eventos discretos* (*Discrete-Event Simulation* – DES) é amplamente utilizada como ferramenta preditiva e experimental, permitindo avaliar o comportamento de sistemas sob múltiplos cenários e medir o impacto de alterações no fluxo de processos, políticas de recursos ou parâmetros operacionais. A integração entre simulação e mineração de processos tem sido explorada em diversas pesquisas, com destaque para (MARUŞTER; BEEST, 2009), que propõe um método de redesenho de processos baseado em modelos descobertos via PM e simulados em ferramentas de Petri Nets. De forma semelhante, estudos mais recentes demonstram a aplicação conjunta dessas técnicas em sistemas de logística e manufatura, utilizando simulação como meio de validar hipóteses e otimizar o desempenho global (WUENNENBERG; WEGERICH; FOTTNER, 2023; MENG et al., 2024).

Entretanto, a construção manual de modelos de simulação ainda é um processo intensivo, que requer tempo, conhecimento técnico especializado e compreensão detalhada dos fluxos operacionais. Essa complexidade torna inviável o uso da simulação como instrumento cotidiano de apoio à decisão, especialmente em contextos que demandam reações rápidas a eventos inesperados. O desafio, portanto, está em como automatizar a geração de modelos de simulação a partir de dados reais, reduzindo o esforço de modelagem e ampliando a aplicabilidade da técnica no suporte à decisão operacional de curto prazo.

Trabalhos como o de (FERRONATO, 2022) propuseram soluções integradas baseadas em *Process Mining*, simulação e otimização multicritério, aplicadas ao contexto hospitalar para o agendamento de cirurgias. O framework PM4SOS demonstrou que a combinação dessas abordagens permite reduzir tempos de espera, ajustar a alocação de recursos e reagir dinamicamente a variações de demanda. A partir dessa base, observa-se a oportunidade de generalizar o conceito, tornando o mecanismo de geração automática de modelos aplicável a diferentes domínios — desde linhas de produção industriais até processos administrativos e logísticos.

Dessa forma, o problema central abordado nesta pesquisa pode ser formulado da seguinte maneira: como automatizar a criação de modelos de simulação baseados em dados reais, extraídos por mineração de processos, de modo a apoiar a tomada de decisão operacional em curto prazo?.

A pesquisa busca preencher lacunas identificadas na literatura e na prática organizacional, especialmente no que tange à integração automatizada entre dados históricos e modelos de

simulação, reduzindo o esforço cognitivo do tomador de decisão e permitindo análises preditivas em tempo reduzido. O desenvolvimento de um gerador de modelos automatizado visa, portanto, aproximar o potencial analítico da mineração de processos da capacidade preditiva da simulação computacional, promovendo uma abordagem prática, escalável e orientada por dados para o apoio à decisão operacional.

1.2 Motivação

A transformação digital tem impulsionado a geração de grandes volumes de dados operacionais em praticamente todos os setores organizacionais. Esses dados, provenientes de sistemas corporativos, sensores e plataformas transacionais, representam uma fonte estratégica de informação sobre o comportamento real dos processos. No entanto, a maior parte desse potencial permanece subutilizada: os dados são frequentemente empregados apenas em análises descritivas, voltadas ao monitoramento retrospectivo, sem oferecer suporte efetivo à tomada de decisão em tempo hábil. Essa limitação é especialmente crítica em contextos de alta variabilidade, nos quais decisões de curto prazo precisam ser tomadas com base em evidências confiáveis e atualizadas.

Nesse cenário, a *mineração de processos* (*Process Mining* – PM) desponta como uma tecnologia promissora para a extração de conhecimento a partir de logs de eventos, permitindo compreender, auditar e aprimorar processos com base em dados reais. A PM tem sido aplicada com sucesso em domínios industriais complexos, como logística interna, manufatura e mineração (WUENNENBERG; WEGERICH; FOTTNER, 2023; BRZYCHCZY; ŻUBER; AALST, 2024), oferecendo diagnósticos precisos sobre gargalos, desvios e desempenho operacional. No entanto, sua natureza essencialmente descritiva ainda limita seu uso como instrumento de previsão ou de apoio dinâmico à decisão.

Por outro lado, a *simulação de eventos discretos* (*Discrete-Event Simulation* – DES) é amplamente reconhecida como uma ferramenta analítica capaz de explorar cenários alternativos e estimar o impacto de decisões antes de sua implementação. A combinação entre mineração de processos e simulação tem se mostrado particularmente poderosa, pois permite unir a observação empírica dos dados à experimentação virtual dos processos (MARUŞTER; BEEST, 2009). Entretanto, a etapa de construção de modelos de simulação ainda representa um obstáculo significativo, demandando esforço cognitivo elevado e conhecimento técnico especializado em modelagem e parametrização de sistemas.

A motivação central deste trabalho surge, portanto, da necessidade de automatizar a geração de modelos de simulação a partir de informações extraídas por mineração de processos. Essa automatização tem potencial para reduzir drasticamente o tempo e o esforço envolvidos na criação de modelos analíticos, ao mesmo tempo em que democratiza o acesso de gestores e analistas a ferramentas de apoio à decisão em ambientes complexos e dinâmicos. Além disso, possibilita o uso de dados históricos e em tempo real como base para análises preditivas e

prescritivas, elevando o nível de maturidade analítica das organizações.

Inspirado em iniciativas como o framework PM4SOS proposto por (FERRONATO, 2022), que integrou mineração de processos, simulação e otimização multicritério para o agendamento cirúrgico, o presente trabalho busca expandir esse conceito para diferentes domínios organizacionais. A proposta é criar um **gerador de modelos de simulação** com capacidade generalizável, aplicável a qualquer processo que possua logs de eventos estruturados, mantendo o ambiente hospitalar apenas como estudo de caso para validação experimental.

Ao promover a integração entre mineração de processos, simulação e otimização, esta pesquisa visa contribuir para o avanço das práticas de gestão operacional baseada em dados. Acredita-se que o desenvolvimento de uma ferramenta automatizada, capaz de gerar modelos de simulação em tempo reduzido, possa fortalecer o suporte à decisão no curto prazo, ampliando a eficiência, a agilidade e a capacidade de adaptação das organizações a ambientes cada vez mais dinâmicos e complexos.

1.3 Estado da Arte

Esta seção sintetiza os principais avanços nas áreas de *Process Mining* (PM), *Simulação de Eventos Discretos* (DES) e sua integração, destacando aplicações em domínios complexos (logística interna, manufatura, mineração e saúde) e, por fim, delineando as lacunas que motivam a presente pesquisa.

1.3.1 Mineração de Processos

A mineração de processos consolida-se como abordagem para extrair, a partir de logs de eventos, o comportamento real dos processos, contemplando tarefas de *descoberta* (derivação de modelos), *conformidade* (comparação de logs com modelos de referência) e *aprimoramento* (análise de desempenho e gargalos). Em ambientes industriais, a PM tem avançado no uso de dados transacionais e de sensores para diagnosticar desvios e medir indicadores operacionais (WUENNENBERG; WEGERICH; FOTTNER, 2023; BRZYCHCZY; ŻUBER; AALST, 2024). Um desafio recorrente em cenários físicos (ex.: mineração subterrânea) é a preparação do *event log*: identificação de *case ID*, abstração de eventos de baixo nível e tratamento de ruído; para isso, têm-se proposto estratégias de abstração supervisionadas e não supervisionadas, além de heurísticas específicas para correlação de eventos (BRZYCHCZY; ŻUBER; AALST, 2024). Em domínios administrativos e de P&D, a PM também tem sido explorada para gerir projetos, com o uso de algoritmos como o *Heuristics Miner* para descobrir fluxos e dependências (JOE et al., 2018).

1.3.2 Simulação de Eventos Discretos

A DES é uma técnica consolidada para previsão e análise *what-if*, permitindo estimar efeitos de políticas operacionais (alocação de recursos, regras de priorização) antes de sua implementação. Em gestão de operações, a simulação suporta a avaliação de throughput, tempos de espera, utilização e confiabilidade. Em sistemas físico-cibernéticos (ex.: logística e mineração), combinar dados de operação com simulação tem se mostrado essencial para entender efeitos de variabilidade e restrições de recursos (WUENNENBERG; WEGERICH; FOTTNER, 2023; MENG et al., 2024).

1.3.3 Integração PM–DES

A literatura propõe a integração entre PM e DES de forma a: (i) descobrir modelos a partir de dados reais; (ii) parametrizar e simular cenários no *As-Is* e *To-Be*; e (iii) comparar ganhos de desempenho (MARUŞTER; BEEST, 2009). Esse fluxo permite que modelos descobertos via PM sejam transformados em modelos simuláveis (e.g., Petri nets/CPN) para avaliar alternativas de redesenho e impacto em indicadores. Mais recentemente, frameworks em logística interna descrevem um *pipeline* end-to-end que parte do sistema real, gera dados sintéticos por DES quando necessário, transforma saídas de simulação em *event logs* e realiza descoberta/conformidade para suportar a otimização iterativa (WUENNENBERG; WEGERICH; FOTTNER, 2023). Em mineração, a integração de dados sensoriais, PM e simulação tem sido aplicada para explicar ciclos operacionais, localizar gargalos e fundamentar ações de melhoria, inclusive com técnicas de abstração de eventos e análise comparativa (BRZYCHCZY; ŻUBER; AALST, 2024). Em paralelo, abordagens de simulação modular têm ampliado a capacidade preditiva em fenômenos acoplados (mecânico–hidráulicos), ilustrando a necessidade de arquiteturas de acoplamento explícito e troca de dados entre módulos (MENG et al., 2024).

1.3.4 Aplicações em domínios complexos

Trabalhos recentes mostram metodologias que combinam PM e DES para diagnosticar gargalos, avaliar conformidade e otimizar parâmetros locais e globais de sistemas de fluxo de materiais, com iterações guiadas por KPIs (tempo de atravessamento, utilização, throughput) (WUENNENBERG; WEGERICH; FOTTNER, 2023). Além disso, ressalta-se a geração de *event logs* a partir de saídas de simulação para fechar o ciclo de descoberta/conformidade e acelerar o aprendizado sobre o sistema.

A PM tem sido empregada também para modelar processos como o ciclo de corte em *longwall*, enfrentando problemas de granularidade, ruído e case correlation; soluções combinam heurísticas e aprendizagem (supervisionada e não supervisionada) para identificação de atividades e instâncias, habilitando descoberta e análise de desempenho com dados de sensores (BRZYCHCZY; ŻUBER; AALST, 2024). Em paralelo, simulações acopladas (mecânica–escoamento)

têm ampliado a previsão de efeitos operacionais e riscos, reforçando o papel de modelos preditivos conectados a dados reais (MENG et al., 2024).

No contexto hospitalar, o PM4SOS integra PM, simulação e otimização multicritério para suporte operacional (e.g., agendamento cirúrgico) (FERRONATO, 2022). Em gestão de projetos, a PM tem servido à descoberta de fluxos e análise de desempenho, com foco em dependências, variações e papéis organizacionais (JOE et al., 2018).

1.3.5 Lacunas e oportunidades

Apesar do progresso, persistem lacunas relevantes:

- Muitas abordagens dependem de etapas manuais (abstração de eventos, mapeamento semântico de atividades, parametrização de tempos e recursos) para viabilizar a simulação; faltam ferramentas que automatizem a transformação de *event logs* em modelos simuláveis com parametrização consistente e reprodutível
- Soluções existentes são, em geral, específicas de domínio (hospitalar, logística, mineração), com limitações de portabilidade de *mapeamentos* e estruturas de dados
- A correlação de eventos, a abstração de sinais contínuos e a ainda demandam estratégias robustas e padronizadas para produção de *event logs* adequados à simulação.
- Há carência de pipelines que fechem o ciclo (dados $\rightarrow$ PM $\rightarrow$ geração automática de modelo $\rightarrow$ DES $\rightarrow$ recomendações/otimização) com tempos de processamento compatíveis com decisões operacionais de curto prazo.

Essas lacunas motivam o desenvolvimento de um *gerador de modelos de simulação* orientado por PM, com escopo generalizável e foco em decisões de curto prazo, reduzindo intervenções manuais, padronizando a parametrização e aproximando diagnóstico descritivo de validação preditiva.

1.4 Soluções Similares

Diversas pesquisas têm buscado integrar *Process Mining* (PM) e *Simulação de Eventos Discretos* (DES) como forma de aprimorar a compreensão e a predição do comportamento dos processos reais. Entretanto, a maioria das soluções existentes mantém um alto grau de dependência de intervenção manual, especialmente nas etapas de modelagem, parametrização e calibração.

Entre as iniciativas mais influentes, destaca-se a metodologia proposta por (MARUŞTER; BEEST, 2009), que combina mineração de processos e simulação para o redesenho organizacional. O método parte de logs reais para gerar modelos *As-Is*, simulá-los e compará-los com

versões otimizadas *To-Be*, permitindo estimar ganhos de desempenho. Apesar de pioneiro, o processo de conversão dos modelos minerados em modelos simuláveis requer ajustes manuais e conhecimento técnico em modelagem formal (como redes de Petri coloridas).

No contexto hospitalar, o framework PM4SOS, desenvolvido por (FERRONATO, 2022), integra mineração de processos, simulação e otimização multicritério para o agendamento cirúrgico. Essa abordagem automatiza parcialmente a geração de modelos e utiliza indicadores de eficiência para suportar decisões em tempo reduzido. Contudo, sua aplicação ainda é restrita ao domínio da saúde, carecendo de generalização para outros tipos de processos.

Na área industrial, trabalhos como (WUENNENBERG; WEGERICH; FOTTNER, 2023) propõem pipelines que unem simulação e mineração de processos em sistemas logísticos internos. Esses modelos exploram o uso de simulação para geração de dados sintéticos, que são posteriormente minerados para verificação de conformidade e detecção de gargalos. Em paralelo, pesquisas em mineração subterrânea (BRZYCHCZY; ŻUBER; AALST, 2024) e simulação modular (MENG et al., 2024) também avançam na integração entre dados de sensores, abstração de eventos e análise preditiva, embora com foco em contextos físicos específicos.

Em síntese, as soluções atuais demonstram o potencial da integração entre PM e DES, mas permanecem limitadas quanto à automação de ponta a ponta e à adaptabilidade entre diferentes domínios. O presente trabalho propõe evoluir essas abordagens por meio de um gerador de modelos de simulação automatizado e generalizável, reduzindo o esforço técnico necessário e ampliando o alcance da análise preditiva em processos de decisão operacional de curto prazo.

1.5 Objetivos

1.5.1 Objetivo Geral

Desenvolver um **gerador de modelos de simulação baseado em mineração de processos** para apoiar a **tomada de decisão no curto prazo**, capaz de criar automaticamente modelos de simulação a partir de logs de eventos, reduzindo o esforço humano e o tempo necessário para a modelagem de sistemas complexos.

1.5.2 Objetivos Específicos

Para alcançar o objetivo geral, este trabalho busca atender aos seguintes objetivos específicos:

- Investigar métodos e técnicas de integração entre mineração de processos, simulação computacional e otimização multicritério;

- Projetar uma arquitetura de sistema capaz de gerar automaticamente modelos de simulação a partir de logs de eventos processados por ferramentas de PM;
- Implementar um protótipo funcional do gerador de modelos, com base em bibliotecas de mineração de processos e simulação (como PM4PY e SimPy);
- Aplicar o protótipo desenvolvido em um estudo de caso no contexto hospitalar, validando sua eficácia na geração de modelos e apoio à decisão operacional;
- Avaliar o desempenho do sistema proposto quanto à precisão dos modelos gerados, tempo de execução e potencial de generalização para outros domínios.

1.6 Justificativa

A crescente disponibilidade de dados operacionais e o avanço das tecnologias analíticas têm impulsionado o desenvolvimento de soluções voltadas à gestão baseada em evidências. No entanto, a transformação desses dados em modelos preditivos e prescritivos ainda depende, em grande parte, de atividades manuais de análise e modelagem, o que limita sua aplicabilidade em contextos que demandam decisões rápidas e precisas. Nesse cenário, a integração entre *mineração de processos* (PM) e *simulação de eventos discretos* (DES) surge como uma abordagem promissora para reduzir a distância entre o conhecimento descritivo e a previsão operacional.

Do ponto de vista **científico**, a pesquisa se justifica pela necessidade de ampliar o corpo de conhecimento existente sobre a integração entre PM e DES, especialmente no que se refere à automação das etapas de modelagem e parametrização. Trabalhos anteriores, como os de (MARUŞTER; BEEST, 2009) e (WUENNENBERG; WEGERICH; FOTTNER, 2023), demonstraram o potencial dessa integração para diagnóstico e otimização de processos, mas ainda requerem intervenções manuais para gerar modelos simuláveis. Assim, o presente estudo contribui para o avanço teórico ao propor um método automatizado que amplia a reprodutibilidade e a aplicabilidade da simulação orientada por dados reais.

Sob a perspectiva **tecnológica**, a proposta apresenta relevância por desenvolver um **gerador automatizado de modelos de simulação**, capaz de transformar *event logs* em modelos DES de forma padronizada e escalável. Essa automação representa um avanço em relação a abordagens anteriores, como o PM4SOS (FERRONATO, 2022), que, embora integre PM e simulação, mantém dependência de ajustes manuais e é restrito a um domínio específico (o hospitalar). A ferramenta proposta busca ser generalizável e adaptável, permitindo sua aplicação em diferentes setores — como logística, manufatura e mineração — sem perda de precisão analítica.

Por fim, do ponto de vista **prático**, o trabalho se justifica pela crescente necessidade de apoiar decisões operacionais de curto prazo em ambientes complexos e dinâmicos. Organizações

modernas demandam respostas rápidas a variações de demanda, falhas operacionais e restrições de recursos, e a geração automatizada de modelos de simulação oferece uma alternativa eficiente para análise de cenários em tempo reduzido. A validação do gerador no contexto hospitalar reforça sua aplicabilidade real, demonstrando o potencial de transferência tecnológica da solução para outros domínios.

Dessa forma, o presente trabalho se justifica por unir relevância teórica, tecnológica e prática, propondo uma abordagem inovadora e generalizável para a automatização da geração de modelos de simulação baseados em mineração de processos. Espera-se, com isso, contribuir para a consolidação de um novo paradigma de *Process Mining* aplicado à tomada de decisão operacional orientada por dados, fortalecendo a integração entre análise descritiva, simulação preditiva e otimização de desempenho em tempo hábil.

2 Referencial Teórico

Este capítulo apresenta os fundamentos conceituais e técnicos que sustentam o desenvolvimento de um gerador de modelos de simulação orientado por mineração de processos (*Process Mining*). São abordados desde os conceitos fundamentais de mineração de processos até a integração com simulação de eventos discretos e indicadores de desempenho operacional aplicados a ambientes hospitalares.

2.1 Mineração de Processos

A mineração de processos (*Process Mining*) é uma disciplina que une conceitos de mineração de dados (*data mining*) e de gerenciamento de processos de negócio (*Business Process Management — BPM*) com o objetivo de extrair conhecimento útil a partir de registros de eventos (*event logs*) provenientes de sistemas de informação. Segundo van der Aalst (AALST, 2016), a mineração de processos tem como propósito descobrir, monitorar e aprimorar processos reais com base nos dados efetivamente registrados, constituindo uma ponte entre a análise orientada a dados e a modelagem formal de processos.

Com a crescente digitalização das operações empresariais e o avanço de sistemas como ERPs, CRMs e sistemas de controle de manufatura, passou a ser possível registrar detalhadamente cada etapa executada em um processo. Esses registros, conhecidos como *event logs*, formam a base para a aplicação de técnicas de mineração de processos. Cada evento registrado representa a execução de uma atividade pertencente a um caso (*case*) e contém atributos como o nome da atividade, o identificador do caso, o responsável (*resource*) e o carimbo de tempo (*timestamp*). A estrutura desses logs permite a reconstituição da sequência de atividades e o estudo do comportamento do processo real (MARUŞTER; BEEST, 2009).

A mineração de processos é composta por três grandes categorias de técnicas (AALST, 2016):

1. **Descoberta de processos (*Process Discovery*)**: tem como objetivo gerar automaticamente um modelo de processo a partir de um log de eventos, sem conhecimento prévio do fluxo. O modelo resultante pode ser representado em diferentes notações, como Redes de Petri, BPMN (*Business Process Model and Notation*) ou Árvores de Processo (*Process Trees*).

2. **Verificação de conformidade (*Conformance Checking*)**: consiste em comparar um modelo de processo pré-existente com um log de eventos real, avaliando a aderência entre o comportamento observado e o comportamento esperado. Essa comparação fornece métricas como *fitness* e *precision*.

3. **Aprimoramento de modelos (*Enhancement*)**: busca enriquecer modelos existentes com informações adicionais provenientes dos logs, como tempos médios de execução, gargalos, desvios e uso de recursos, permitindo a análise de desempenho e a detecção de oportunidades de otimização.

O ciclo de mineração de processos, descrito no *Process Mining Manifesto* da IEEE Task Force on Process Mining (AALST, 2016), enfatiza que a aplicação prática da técnica depende da disponibilidade e da qualidade dos logs de eventos. Logs incompletos, inconsistentes ou mal formatados comprometem a acurácia dos modelos descobertos. Nesse sentido, Kherbouche et al. (KHERBOUCHE; LAGA; MASSE, 2020) destacam a importância da avaliação da qualidade dos logs, propondo métricas de *completude*, *consistência* e *complexidade* antes da aplicação das técnicas de mineração.

Em termos de arquitetura, um sistema de mineração de processos segue um fluxo básico: coleta de dados, pré-processamento, mineração propriamente dita e análise dos resultados. Durante a coleta, os logs podem ser extraídos de sistemas como SAP, Oracle, ou de bancos de dados customizados. O pré-processamento envolve a limpeza e padronização dos dados, incluindo a identificação correta de casos e atividades. Em seguida, algoritmos como *Alpha Miner*, *Heuristic Miner* e *Inductive Miner* são aplicados para a descoberta do modelo de processo (LEEMANS; FAHLAND; AALST, 2013).

Os resultados podem ser apresentados em notações formais, como Redes de Petri, ou em linguagens mais visuais, como BPMN. O uso de ferramentas especializadas, como o *ProM Framework* e a biblioteca Python PM4Py, facilita essa análise e integração com outros métodos, como a simulação de eventos discretos (FERRONATO; SCALABRIN, 2021).

O padrão *eXtensible Event Stream* (XES), definido pela IEEE (IEEE Computational Intelligence Society, 2010), estabelece a estrutura de logs de eventos para garantir interoperabilidade entre ferramentas e consistência na troca de dados. Cada log em XES é composto por uma coleção de *traces* (casos), e cada *trace* contém uma sequência ordenada de *events*. Essa padronização é essencial para o sucesso de frameworks modernos de mineração de processos e simulação integrada.

Portanto, a mineração de processos se consolida como uma abordagem essencial para compreender, auditar e melhorar processos organizacionais em ambientes baseados em dados. Quando combinada com simulação e análise estatística, ela se torna uma ferramenta poderosa para suporte à decisão, otimização operacional e melhoria contínua de processos complexos, como os hospitalares e logísticos.

2.2 Inductive Miner

Entre os diversos algoritmos de descoberta de processos disponíveis, o *Inductive Miner* (IM), proposto por Leemans, Fahland e van der Aalst (LEEMANS; FAHLAND; AALST, 2013), é um dos mais relevantes e amplamente utilizados na literatura e em ferramentas modernas de mineração, como o *ProM* e o `PM4Py`.

Diferentemente de abordagens anteriores, como o *Alpha Miner* e o *Heuristic Miner*, o Inductive Miner foi projetado para garantir propriedades formais no modelo resultante, como a *soundness* (correção comportamental) e a estrutura hierárquica em blocos (*block-structured*). Essas propriedades asseguram que o modelo possa ser executado sem estados mortos ou impasses, além de facilitar sua conversão em linguagens formais como Redes de Petri e Árvores de Processo (*Process Trees*).

O princípio fundamental do algoritmo baseia-se na decomposição recursiva do log de eventos. O IM analisa as relações de precedência e causalidade entre atividades e divide o log em sublogs coerentes, representando fragmentos independentes do processo. Cada sublog é então minerado de forma isolada, e os resultados são combinados em uma estrutura hierárquica que reflete a composição lógica das atividades.

Uma das vantagens práticas do Inductive Miner é sua compatibilidade direta com representações de simulação. Ao produzir modelos formalmente corretos e livres de inconsistências estruturais, o IM facilita a transformação automática em modelos de simulação baseados em Redes de Petri, como destacado por Ferronato (FERRONATO; SCALABRIN, 2021). Essa característica é essencial para o desenvolvimento de sistemas como o *PM2Sim*, que automatiza a criação de modelos de simulação de eventos discretos.

2.3 Redes de Petri

As Redes de Petri constituem um formalismo matemático e gráfico amplamente utilizado para modelar, analisar e simular sistemas de eventos discretos. Propostas originalmente por Carl Adam Petri na década de 1960 e formalizadas por Peterson (PETERSON, 1981), essas redes oferecem uma base rigorosa para a representação de processos dinâmicos caracterizados por concorrência, sincronização, conflito e causalidade — propriedades típicas de sistemas produtivos, logísticos e hospitalares.

Uma Rede de Petri é definida formalmente como uma tupla $N = (P, T, F)$, onde:

- P representa o conjunto de lugares (*places*);
- T representa o conjunto de transições (*transitions*);
- $F \subseteq (P \times T) \cup (T \times P)$ é o conjunto de arcos que conectam lugares e transições.

Os lugares simbolizam condições ou estados do sistema, enquanto as transições representam eventos ou atividades que modificam esses estados. A dinâmica do sistema é descrita por meio da movimentação de fichas (*tokens*) entre os lugares. O conjunto de tokens em um dado instante define a marcação atual da rede (*marking*), representando o estado do processo. Quando todas as condições de disparo de uma transição são satisfeitas, ela se torna habilitada e pode ser executada, consumindo e produzindo tokens conforme o fluxo definido em F. Essa semântica de disparo possibilita a modelagem de sistemas paralelos e assíncronos de forma intuitiva e precisa.

Segundo Peterson (PETERSON, 1981), uma das principais vantagens das Redes de Petri é a possibilidade de realizar análises formais de propriedades do sistema modelado, como:

- **Alcançabilidade** (*Reachability*): determina quais estados podem ser atingidos a partir da marcação inicial.
- **Viveza** (*Liveness*): garante que nenhuma transição se torne permanentemente inativa, evitando estados mortos.
- **Conservação** (*Boundedness*): assegura que o número de tokens em cada lugar permanece finito, prevenindo explosões de estados.
- **Deadlock-freedom**: certifica que o sistema não entra em impasse completo.

Essas propriedades são fundamentais para a verificação de correção comportamental (*soundness*) em modelos de processos descobertos via mineração, assegurando que o modelo é executável e que todo caso pode ser concluído corretamente (AALST, 2016).

No contexto da mineração de processos, as Redes de Petri são amplamente utilizadas como formalismo intermediário para representar os modelos extraídos de logs de eventos. O *Inductive Miner*, por exemplo, gera diretamente uma Rede de Petri *sound* e *block-structured* (LEEMANS; FAHLAND; AALST, 2013), permitindo tanto a verificação de conformidade quanto a execução simulada do processo. Essa característica é essencial para a integração entre mineração e simulação de eventos discretos, pois possibilita a tradução direta de modelos minerados em estruturas simuláveis.

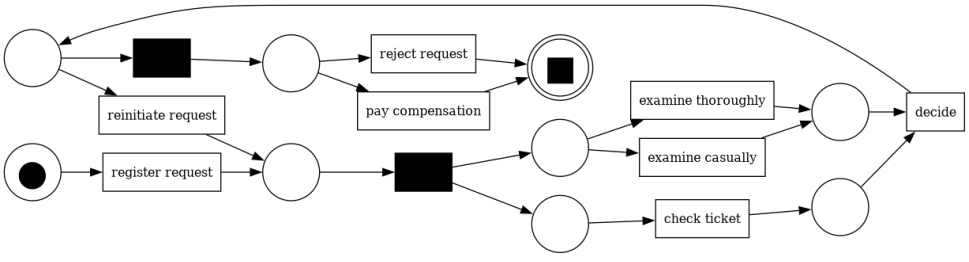


Figura 1 – Exemplo de Rede de Petri representando um processo de negócio. Fonte: Autor (2025)

Na Figura 1, os elementos da Rede de Petri são representados da seguinte forma: os círculos representam *lugares* (places), que indicam estados ou condições do processo; os retângulos representam *transições* (transitions), que correspondem a atividades ou eventos que podem ocorrer; os círculos com pontos pretos internos representam *tokens* (marcas), que indicam a presença de uma condição ou o estado atual do processo; e os círculos com contorno destacado e tokens internos representam lugares marcados, indicando estados ativos no momento atual da execução.

Ferronato (FERRONATO; SCALABRIN, 2021) destaca que as Redes de Petri desempenham um papel central na integração entre mineração e simulação. Em seu framework *PM2Sim*, as redes extraídas a partir de logs de eventos são automaticamente convertidas em modelos de simulação de eventos discretos implementados em Python. Essa conversão permite avaliar métricas como tempo de ciclo, gargalos e utilização de recursos, transformando os modelos minerados em instrumentos de suporte à decisão operacional.

Em síntese, as Redes de Petri constituem uma linguagem formal robusta e expressiva para representar o comportamento de sistemas reais. Sua adoção no contexto da mineração de processos garante não apenas a fidelidade comportamental dos modelos gerados, mas também sua viabilidade para análises de desempenho e simulação, consolidando-se como elo fundamental entre a teoria de processos e a prática da modelagem computacional.

2.4 Simulação de Eventos Discretos

A simulação de eventos discretos (*Discrete Event Simulation — DES*) é uma técnica de modelagem computacional amplamente utilizada para representar sistemas dinâmicos em que o estado do sistema muda apenas em pontos discretos no tempo. Cada mudança é provocada por um evento, que ocorre em um instante específico e representa uma transição de estado. Essa abordagem é amplamente empregada em áreas como manufatura, logística, saúde e serviços, onde os processos são compostos por atividades sequenciais, paralelas e dependentes de recursos.

A DES permite reproduzir o comportamento de sistemas complexos sem a necessidade de interferir em seu ambiente real, possibilitando análises de desempenho, previsão de gargalos e avaliação de cenários alternativos. Diferentemente da simulação contínua, que modela fenômenos em tempo contínuo por meio de equações diferenciais, a simulação discreta descreve processos orientados por eventos, sendo, portanto, ideal para a representação de fluxos de trabalho e processos empresariais.

Em termos formais, um modelo de simulação de eventos discretos é composto por três elementos básicos:

- **Entidades**: representam os objetos que transitam pelo sistema (por exemplo, pacientes, ordens de serviço ou produtos).

- **Recursos**: correspondem aos elementos que executam as atividades (como profissionais, máquinas ou salas cirúrgicas).
- **Eventos**: são as ocorrências que alteram o estado do sistema, como o início ou o término de uma atividade.

Esses componentes são orquestrados por um *motor de simulação* (*simulation engine*) que mantém um relógio lógico e agenda os próximos eventos a serem processados. Cada evento executado pode gerar novos eventos futuros, modificando o estado do sistema e permitindo a evolução temporal do modelo (LIU, 2015).

A integração entre simulação e mineração de processos vem sendo explorada nos últimos anos. Liu (LIU, 2015) demonstrou que os modelos extraídos via *process mining* podem ser automaticamente convertidos em modelos de simulação de eventos discretos, reduzindo o esforço de modelagem e aumentando a precisão analítica. Essa integração é especialmente relevante em contextos onde a dinâmica do sistema se altera frequentemente, exigindo atualizações rápidas de modelos e previsões.

Nesse contexto, Ferronato e Scalabrin (FERRONATO; SCALABRIN, 2021) desenvolveram o framework *PM2Sim*, que automatiza a criação de modelos de simulação a partir de logs de eventos reais. O sistema identifica atividades, durações, tempos de espera e recursos envolvidos, transformando essas informações em um modelo DES implementado em Python com a biblioteca `SimPy`. Essa biblioteca fornece uma estrutura orientada a processos, baseada em geradores, que permite modelar entidades como processos que interagem em um ambiente temporal discreto. A `SimPy` oferece ainda suporte à criação de filas, controle de recursos, eventos simultâneos e coleta de estatísticas de desempenho, tornando-a ideal para aplicações em mineração de processos e análise operacional.

Além disso, o framework *PM2Sim* utiliza distribuições estatísticas ajustadas por meio de técnicas de *fitting* (adequação), com base em bibliotecas como `NumPy` e `SciPy`, permitindo representar o comportamento dos tempos de execução das atividades. O resultado é um modelo de simulação que reflete o comportamento observado nos logs, possibilitando a análise de métricas como tempo de ciclo, utilização de recursos e identificação de gargalos.

De acordo com Wuennenberg et al. (WUENNENBERG; WEGERICH; FOTTNER, 2023), a combinação de mineração de processos e simulação discreta é particularmente útil em ambientes industriais e logísticos, onde a qualidade dos dados e a variabilidade operacional representam desafios significativos. Nesse sentido, a DES serve como ferramenta de validação e experimentação para modelos minerados, permitindo testar hipóteses e prever o impacto de mudanças estruturais antes de sua implementação no ambiente real.

2.5 Análise Estatística

A etapa de análise estatística é fundamental para a construção de modelos de simulação realistas e coerentes com os processos observados. Após a descoberta do modelo de processo e a extração das informações de desempenho a partir dos logs de eventos, é necessário realizar o ajuste estatístico dos parâmetros temporais, como durações de atividades, tempos de espera e intervalos entre eventos. Esses parâmetros são essenciais para que o modelo de simulação reflita adequadamente a variabilidade e o comportamento estocástico do sistema.

Segundo van der Aalst (AALST, 2016), a mineração de processos deve ser vista como uma disciplina orientada por dados (*data-driven*), e o sucesso de suas aplicações depende da correta interpretação das distribuições de tempo, frequência e desempenho que emergem dos registros de eventos. O uso de técnicas estatísticas complementa a fase de descoberta, permitindo transformar modelos descritivos em modelos quantitativos capazes de realizar simulações e análises preditivas.

Ferronato e Scalabrin (FERRONATO; SCALABRIN, 2021) destacam que o uso de distribuições estatísticas é um dos pilares do framework *PM2Sim*, que automatiza a criação de modelos de simulação a partir de logs reais. No sistema proposto, os tempos de execução das atividades e os intervalos entre eventos são ajustados a partir de funções de probabilidade clássicas, como Normal, Log-Normal e Exponencial. Essa parametrização é obtida por meio do ajuste de distribuições (*distribution fitting*), realizado com base nos dados históricos de cada atividade registrada no log. O processo envolve a estimativa dos parâmetros das distribuições e a verificação de aderência dos dados observados, assegurando a representatividade do modelo.

Para realizar esses ajustes, bibliotecas estatísticas como `NumPy` e `SciPy` são utilizadas no ambiente Python, permitindo estimar as distribuições de probabilidade e calcular medidas como média, variância, desvio padrão e coeficiente de variação. Além disso, testes de aderência, como o de Kolmogorov–Smirnov, são empregados para validar se os dados amostrais seguem adequadamente a distribuição teórica selecionada. Esse procedimento garante que os tempos de simulação não apenas reproduzam as médias históricas, mas também capturem a variabilidade natural do processo (LIU, 2015).

O resultado da análise estatística é incorporado diretamente aos parâmetros do modelo de simulação, alimentando o mecanismo de eventos discretos com tempos amostrados de distribuições probabilísticas. Essa abordagem permite representar o comportamento dinâmico e imprevisível dos processos reais, algo essencial em ambientes sujeitos a variações operacionais, como hospitais e sistemas logísticos. Conforme observado por Leemans et al. (LEEMANS; FAHLAND; AALST, 2013), a precisão dos tempos e a correta modelagem da frequência das atividades impactam diretamente a capacidade do modelo minerado em reproduzir o fluxo real dos eventos.

Assim, a análise estatística atua como uma ponte entre a descoberta de processos e

a simulação de eventos discretos, traduzindo dados empíricos em parâmetros quantitativos para o modelo. Essa integração garante que a simulação gerada preserve as características estatísticas do processo original, permitindo a avaliação confiável de métricas de desempenho e a experimentação de cenários alternativos de operação.

2.6 Métricas de Qualidade de Modelos

A qualidade dos modelos descobertos por meio da mineração de processos é um fator determinante para a confiabilidade das análises subsequentes e, principalmente, para o uso desses modelos como base para simulações. A avaliação sistemática da qualidade garante que o modelo minerado não apenas represente corretamente o comportamento histórico, mas também seja capaz de generalizar o comportamento futuro e sustentar decisões operacionais baseadas em evidências.

De acordo com van der Aalst (AALST, 2016), um modelo de processo deve ser avaliado em múltiplas dimensões de qualidade, de modo a equilibrar precisão, generalização e simplicidade. As principais métricas utilizadas na literatura são: *fitness*, *precision*, *generalization* e *simplicity*. Essas métricas, amplamente adotadas em ferramentas como o *ProM Framework* e a `PM4Py`, compõem a base dos algoritmos de verificação de conformidade (*conformance checking*), responsáveis por comparar o comportamento observado nos logs de eventos com o comportamento previsto pelo modelo minerado.

- **Fitness**: avalia o quanto o modelo reproduz o comportamento observado no log. Um modelo com alto *fitness* consegue reproduzir todas as sequências válidas de atividades sem apresentar falhas de execução.
- **Precision**: mede o nível de restrição do modelo, penalizando comportamentos não observados nos dados. Modelos excessivamente permissivos tendem a apresentar alta flexibilidade, mas baixa precisão.
- **Generalization**: representa a capacidade do modelo de capturar variações plausíveis do processo, evitando o sobreajuste aos dados históricos.
- **Simplicity**: reflete o grau de complexidade estrutural do modelo; modelos excessivamente complexos, embora precisos, dificultam a interpretação e a manutenção (AALST et al., 2012).

O *Process Mining Manifesto* (AALST et al., 2012) destaca a importância de equilibrar essas dimensões, pois a busca por um modelo com *fitness* perfeito pode levar à perda de generalização, e vice-versa. A maturidade da área de mineração de processos se reflete justamente na capacidade de lidar com esse compromisso entre precisão e abstração. Esse equilíbrio é

particularmente relevante quando os modelos descobertos serão empregados em simulações, uma vez que comportamentos não observados ou superajustados podem comprometer a validade dos resultados simulados.

Liu (LIU, 2015) ressalta que a integração entre mineração e simulação exige não apenas um modelo logicamente consistente, mas também estatisticamente representativo. A validação entre logs reais e logs sintéticos simulados permite medir a aderência entre ambos, utilizando métricas de distância e similaridade, como a *edit distance*. Essa abordagem garante que a simulação não apenas reproduza o fluxo de atividades, mas também mantenha coerência temporal e probabilística com o processo original.

Kherbouche et al. (KHERBOUCHE; LAGA; MASSE, 2020) complementam essa perspectiva ao argumentar que a qualidade do modelo está diretamente relacionada à qualidade do log de eventos. Logs incompletos, redundantes ou ruidosos geram modelos com baixa precisão e menor confiabilidade. Por isso, o controle da qualidade dos logs — avaliado por dimensões como *completude*, *consistência*, *acurácia* e *complexidade* — é pré-requisito para a obtenção de métricas significativas de *fitness* e *precision*.

Portanto, as métricas de qualidade de modelos constituem um elo crítico entre mineração e simulação. Elas fornecem os indicadores necessários para validar a representatividade, robustez e aplicabilidade do modelo descoberto. Um modelo de processo só é efetivamente útil quando combina boa aderência aos dados históricos com capacidade de generalização para novos cenários — característica essencial para o uso em ambientes de tomada de decisão e otimização operacional.

2.7 Geração de Logs Sintéticos

A geração de logs sintéticos é uma etapa estratégica na integração entre mineração de processos e simulação de eventos discretos. Essa técnica permite criar registros artificiais de eventos que preservam as propriedades estruturais e estatísticas de logs reais, viabilizando experimentos controlados, testes de desempenho e validação de modelos sem a necessidade de utilizar dados sensíveis ou restritos. Em termos práticos, os logs sintéticos servem como uma representação simulada do comportamento observado, permitindo avaliar a fidelidade dos modelos minerados e a confiabilidade das previsões operacionais.

Segundo van der Aalst (AALST, 2016), a utilização de logs artificiais é essencial para verificar se o modelo de processo descoberto é capaz de reproduzir o comportamento do sistema real. Essa comparação é comumente feita por meio de técnicas de *conformance checking*, nas quais o log sintético gerado pelo modelo é confrontado com o log original, medindo-se métricas como *fitness* e *precision*. Essa abordagem é particularmente útil em ambientes dinâmicos, onde a estrutura do processo pode mudar com frequência, como na área hospitalar e em sistemas

logísticos.

Ferronato (FERRONATO; SCALABRIN, 2021) propõe no framework *PM2Sim* um processo automatizado de geração de logs sintéticos a partir de modelos descobertos via mineração de processos. O sistema transforma o modelo minerado — geralmente representado como uma Rede de Petri — em um modelo de simulação implementado em Python por meio da biblioteca `SimPy`. Durante a execução da simulação, eventos são registrados em formato XES (*eXtensible Event Stream*), mantendo a compatibilidade com ferramentas de mineração como PM4Py e ProM. Essa estratégia permite a retroalimentação do ciclo de mineração, criando uma integração contínua entre descoberta, simulação e validação.

Augusto et al. (AUGUSTO et al., 2016) reforçam essa importância ao aplicar a integração entre mineração de processos e simulação em um contexto clínico. Os autores desenvolveram uma metodologia para simular fluxos de pacientes com base em logs hospitalares nacionais, combinando mineração e simulação. Essa abordagem permitiu a criação de logs sintéticos de trajetórias clínicas, utilizados para avaliar o impacto de decisões médicas e políticas de gestão sobre taxas de mortalidade e custos. Os resultados evidenciam o potencial da geração de logs sintéticos como ferramenta de experimentação em ambientes de alta complexidade e variabilidade.

A geração de dados artificiais também desempenha um papel importante na análise de eficiência operacional de ambientes hospitalares. O estudo de Protil et al. (STROPARO; BICHINHO; PROTIL, 2004) sobre a ocupação de centros cirúrgicos demonstrou que o uso de modelos simulados possibilita identificar desperdícios de tempo e gargalos no agendamento de cirurgias. Embora o trabalho não empregue mineração de processos, ele antecipa a importância da simulação como meio de geração de dados para planejamento e otimização de recursos — uma função equivalente à dos logs sintéticos em contextos modernos.

Outro aspecto relevante é a qualidade dos logs gerados. Conforme Kherbouche et al. (KHERBOUCHE; LAGA; MASSE, 2020), a utilidade dos logs sintéticos depende da fidelidade com que reproduzem as características estatísticas, temporais e comportamentais dos registros reais. Logs incompletos, redundantes ou inconsistentes podem comprometer a avaliação do modelo.

A mineração fornece o modelo descritivo; a simulação, o ambiente de experimentação; e os logs sintéticos, o instrumento de validação. Esse ciclo permite aprimorar continuamente o modelo de processo, ajustando suas propriedades comportamentais e estatísticas com base em dados observados e simulados. Resumidamente, a geração de logs sintéticos transforma o modelo minerado em uma ferramenta viva de aprendizado e decisão, capaz de antecipar cenários e apoiar a otimização operacional em tempo reduzido.

2.8 Indicadores de Eficiência Operacional (ORE)

A avaliação da eficiência operacional é um componente essencial em sistemas que buscam otimização contínua de processos, especialmente em ambientes hospitalares. Nesse contexto, Souza, Vaccaro e Lima (SOUZA; VACCARO; LIMA, 2020) propuseram o indicador *Operating Room Effectiveness* (ORE), inspirado no conceito de *Overall Equipment Effectiveness* (OEE) da manufatura enxuta. O ORE foi concebido para medir o desempenho de centros cirúrgicos sob uma ótica *lean healthcare*, permitindo identificar perdas e desperdícios nos processos assistenciais.

O indicador ORE é composto por três dimensões principais:

- **Planejamento**: avalia a aderência entre o cronograma previsto e a execução real das cirurgias, medindo atrasos, cancelamentos e ociosidade das salas.
- **Desempenho**: mede a eficiência da execução cirúrgica em relação aos tempos planejados e à taxa de ocupação das salas operatórias.
- **Qualidade**: considera o impacto de falhas, retrabalhos e complicações que afetam a utilização dos recursos hospitalares.

Os resultados apresentados por Souza et al. (SOUZA; VACCARO; LIMA, 2020) demonstraram ganhos de eficiência de até 12% e economias anuais estimadas em US$400.000 após a implementação do indicador em um hospital universitário brasileiro. Tais resultados evidenciam o potencial do ORE como métrica de apoio à gestão operacional e à tomada de decisão em ambientes de alta complexidade.

Em trabalhos anteriores, Protil et al. (STROPARO; BICHINHO; PROTIL, 2004) já haviam explorado o uso de modelagem e simulação de sistemas para analisar a taxa de ocupação de centros cirúrgicos, apontando a importância da simulação como ferramenta para otimização de recursos hospitalares. A integração entre indicadores como o ORE e abordagens baseadas em mineração e simulação, conforme sugerido por Ferronato (FERRONATO; SCALABRIN, 2021), amplia a capacidade analítica desses sistemas, permitindo correlacionar métricas de eficiência com o comportamento real dos processos.

Dessa forma, o uso combinado de indicadores operacionais e modelos minerados fornece uma visão quantitativa e dinâmica da eficiência hospitalar, permitindo avaliar e prever o impacto de decisões sobre produtividade, custos e qualidade dos serviços de saúde.

2.9 Padrão XES (eXtensible Event Stream)

O padrão *eXtensible Event Stream* (XES) foi desenvolvido pela *IEEE Task Force on Process Mining* com o objetivo de padronizar a representação de logs de eventos utilizados em mineração de processos. Formalizado pela norma IEEE 1849-2016 (IEEE Computational

Intelligence Society, 2010), o XES define uma estrutura extensível e interoperável que permite armazenar e trocar informações sobre execuções de processos entre diferentes ferramentas e plataformas.

Cada log XES é composto por um conjunto de *traces*, que representam casos individuais de execução, e cada *trace* contém uma sequência ordenada de *events*, correspondentes às atividades executadas. Cada evento possui atributos obrigatórios — como nome da atividade, identificador do caso e carimbo de tempo — e opcionais, como recursos, custos ou anotações adicionais. Essa estrutura hierárquica assegura a consistência semântica dos dados e permite análises multi-perspectiva (fluxo, tempo e organização).

O Código 2.1 apresenta um trecho simplificado de um log XES que segue o padrão IEEE, ilustrando a estrutura de um *trace* (caso) com três eventos correspondentes a um processo de atendimento hospitalar.

Código 2.1 – Exemplo simplificado de log XES

```
<?xml version="1.0" encoding="UTF-8"?>
<log xes.version="1.0" xes.features="nested-attributes"
    xmlns="http://www.xes-standard.org/">
  <trace>
    <string key="concept:name" value="Case001"/>
    <event>
      <string key="concept:name" value="Admissao␣do␣Paciente"/>
      <date key="time:timestamp"
         value="2024-03-15T08:30:00.000+00:00"/>
      <string key="org:resource" value="Recepcao"/>
    </event>
    <event>
      <string key="concept:name" value="Avaliacao␣Medica"/>
      <date key="time:timestamp"
         value="2024-03-15T09:00:00.000+00:00"/>
      <string key="org:resource" value="Dr.␣Silva"/>
    </event>
    <event>
      <string key="concept:name" value="Alta␣Hospitalar"/>
      <date key="time:timestamp"
         value="2024-03-15T10:15:00.000+00:00"/>
      <string key="org:resource" value="Administracao"/>
    </event>
  </trace>
</log>
```

Nesse exemplo, o caso `Case001` representa a trajetória de um paciente desde a admissão

até a alta hospitalar. Cada evento contém informações sobre a atividade executada (*concept:name*), o horário em que ocorreu (*time:timestamp*) e o recurso responsável (*org:resource*). A simplicidade e a extensibilidade do formato permitem que diferentes sistemas coletem e exportem dados compatíveis para posterior mineração.

3 Metodologia

Este capítulo apresenta a metodologia utilizada para o desenvolvimento do gerador de modelos de simulação baseado em mineração de processos. A abordagem metodológica foi estruturada em quatro etapas principais: análise automática de logs, mineração de processos, simulação de eventos discretos e validação de qualidade.

3.1 Visão Geral da Abordagem Metodológica

A metodologia proposta consiste em um pipeline sequencial de quatro etapas interdependentes, conforme ilustrado na Figura

1. **Análise Automática de Logs**: detecção de atributos e validação de compatibilidade
2. **Mineração de Processos**: extração do modelo formal e parâmetros estatísticos
3. **Simulação de Logs Sintéticos**: geração de casos baseada em eventos discretos
4. **Validação de Qualidade**: avaliação de similaridade e conformidade

O fluxo completo pode ser representado com a figura abaixo:

Cada etapa produz artefatos específicos que alimentam a etapa seguinte, garantindo rastreabilidade e reprodutibilidade do processo. O sistema foi projetado para ser 100% genérico, funcionando com logs de qualquer domínio organizacional que disponha de registros estruturados em formato XES.

3.2 Etapa 1: Análise Automática de Logs

3.2.1 Objetivo

Detectar automaticamente as características estruturais e temporais do log de entrada, assegurando compatibilidade com diferentes domínios e formatos sem necessidade de configuração manual.

3.2.2 Justificativa

Logs de eventos variam significativamente entre domínios quanto à nomenclatura de atributos, presença de recursos organizacionais e convenções de registro temporal. A análise

automática elimina a necessidade de parametrização manual e permite processamento agnóstico ao domínio, ampliando a aplicabilidade do sistema.

3.2.3 Detecção de Atributos-Chave

O sistema implementa um mecanismo de detecção baseado em prioridades, testando sequencialmente diferentes convenções de nomenclatura até identificar os atributos obrigatórios:

- **Activity Key** (nome da atividade): `concept:name`, `Activity`, `activity`, `event`, `task`
- **Timestamp Key** (momento do evento): `time:timestamp`, `timestamp`, `Time`, `start_time`
- **Case ID Key** (identificador do caso): `concept:name`, `case_id`, `CaseID`, `Case`
- **Resource Key** (executor - opcional): `org:resource`, `resource`, `user`, `actor`

A ordem de prioridade baseia-se no padrão IEEE XES (IEEE Computational Intelligence Society, 2010) e em análise de datasets públicos do BPI Challenge e repositório 4TU.

Quando nenhum candidato da lista de prioridades é encontrado, o sistema aplica busca por palavras-chave nos nomes dos atributos. Se ainda assim a detecção falhar para atributos obrigatórios, uma exceção é lançada com lista dos atributos disponíveis para diagnóstico.

3.2.4 Coleta de Estatísticas Estruturais

Para cada log analisado, o sistema extrai as seguintes informações:

- Número total de casos (traces) e eventos
- Número de atividades únicas
- Distribuição de frequência das atividades
- Comprimento mínimo, máximo e médio dos traces
- Número de variantes do processo (sequências únicas de atividades)
- Distribuição de recursos por atividade (quando disponível)

Essas estatísticas são armazenadas em um objeto `LogProfile`, que serve como entrada para as etapas posteriores e permite análise exploratória dos dados antes da mineração.

3.2.5 Saída da Etapa

Um objeto `LogProfile` contendo todos os metadados detectados, incluindo mapeamento de atributos, estatísticas estruturais e flags de compatibilidade.

3.3 Etapa 2: Mineração de Processos

3.3.1 Objetivo

Extrair o modelo formal do processo na forma de uma Rede de Petri e parametrizar suas características temporais e organizacionais para uso na simulação.

3.3.2 Filtragem de Variantes

3.3.2.1 Problema

Logs reais frequentemente contêm ruído, casos excepcionais e variantes raras que dificultam a descoberta de modelos representativos e podem levar a overfitting.

3.3.2.2 Solução

Aplicação de filtragem baseada em frequência, mantendo apenas as variantes que representam um percentual especificado dos casos totais.

3.3.2.3 Parâmetro Padrão

O sistema utiliza como padrão a retenção de 80% das variantes mais frequentes (`variant_filter=0.8`).

3.3.2.4 Justificativa

O valor de 80% fundamenta-se no Princípio de Pareto, onde aproximadamente 20% das variantes explicam 80% do comportamento observado (AALST, 2016). Este limiar oferece balance adequado entre cobertura comportamental e remoção de ruído.

3.3.2.5 Trade-offs

- Valores baixos (60%): removem mais ruído mas podem perder comportamentos relevantes
- Valores altos (95%): mantêm mais comportamento mas incluem mais ruído

3.3.3 Descoberta do Modelo de Processo

3.3.3.1 Algoritmo Selecionado

O sistema utiliza o **Inductive Miner** proposto por Leemans, Fahland e van der Aalst (LEEMANS; FAHLAND; AALST, 2013).

3.3.3.2 Justificativa da Escolha

A Tabela 1 apresenta comparação entre algoritmos de descoberta disponíveis:

Tabela 1 – Comparação entre algoritmos de descoberta de processos

Algoritmo	**Vantagens**	**Desvantagens**	**Decisão**
Alpha Miner	Simples, pioneiro	Não lida com ruído, loops	Não escolhido
Heuristic Miner	Tolerante a ruído	Modelos não-formais	Não escolhido
Inductive Miner	Soundness garantido, robusto a ruído, sempre produz modelo	Pode generalizar demais	**ESCOLHIDO**
Split Miner	Alta precisão	Requer tuning complexo	Não escolhido

O Inductive Miner foi escolhido por garantir três propriedades fundamentais:

1. **Soundness**: modelo sempre bem-formado, sem deadlocks
2. **Fitness**: modelo sempre capaz de reproduzir o log
3. **Completude**: sempre produz um modelo, mesmo com logs problemáticos

Além disso, o algoritmo gera modelos estruturados em blocos (block-structured), facilitando a conversão para formatos simuláveis.

3.3.3.3 Saída

Uma Rede de Petri formalmente definida como a tupla $N = (P, T, F)$, onde:

- P: conjunto de lugares (places)
- T: conjunto de transições (transitions)
- $F \subseteq (P \times T) \cup (T \times P)$: conjunto de arcos

Acompanhada de marcação inicial (M_0) e marcação final (M_f).

3.3.4 Extração de Estatísticas Temporais

3.3.4.1 Cálculo de Durações

Para cada atividade, o sistema calcula durações baseando-se em:

Casos com timestamp de conclusão:

$$duration = time : complete - time : timestamp \tag{3.1}$$

Casos sem timestamp de conclusão (maioria dos logs):

$$duration = timestamp(event_{i+1}) - timestamp(event_i) \tag{3.2}$$

3.3.5 Ajuste de Distribuições Estatísticas

3.3.5.1 Objetivo

Modelar a variabilidade realista das durações para simulação estocástica fiel ao comportamento observado.

3.3.5.2 Distribuições Candidatas

O sistema testa três distribuições de probabilidade:

1. **Normal (Gaussiana)**: para atividades com duração relativamente constante e variação simétrica
2. **Log-Normal**: para atividades com cauda longa à direita, comum em processos humanos
3. **Exponencial**: para tempos de espera e processos de chegada

3.3.6 Métricas Globais do Processo

O sistema extrai três métricas temporais globais:

3.3.6.1 Taxa de Chegada (Arrival Rate)

$$arrival\_rate = \frac{\text{tempo\_total}}{\text{número\_de\_casos}} \tag{3.3}$$

Representa o intervalo médio entre início de casos consecutivos, expresso em minutos.

3.3.6.2 Taxa de Dispersão (Dispersion Rate)

$$dispersion\_rate = \frac{\text{tempo\_total}}{\text{casos\_finalizados}} \tag{3.4}$$

Representa o intervalo médio entre conclusão de casos, expresso em minutos.

3.3.6.3 Duração Mediana de Casos

$$median\_duration = \text{mediana}(\text{tempo\_fim} - \text{tempo\_início}) \tag{3.5}$$

Utiliza-se a mediana por ser mais robusta a outliers que a média aritmética.

3.3.7 Avaliação de Qualidade do Modelo

O sistema calcula três métricas complementares de qualidade (AALST, 2016):

3.3.7.1 Fitness (0-1, maior é melhor)

Definição: proporção de comportamento do log que o modelo consegue reproduzir.

Método: replay fitness baseado em alinhamentos.

Interpretação:

- ≥ 0.9: modelo explica quase todo o log
- $0.7 - 0.9$: modelo explica maior parte
- < 0.7: modelo inadequado

3.3.7.2 Precision (0-1, maior é melhor)

Definição: quão preciso é o modelo, evitando comportamento extra não observado.

Método: token-based conformance.

Interpretação:

- ≥ 0.8: modelo preciso
- < 0.5: modelo muito generalista

3.3.7.3 Simplicity (0-1, maior é melhor)

Definição: simplicidade estrutural do modelo.

Método: razão entre arcos e nós na rede de Petri.

Interpretação:

- ≥ 0.7: modelo simples
- < 0.4: modelo complexo

3.3.7.4 Trade-off Fundamental

Existe tensão natural entre fitness e precision: alta fitness com baixa precision indica modelo muito permissivo (flower model), enquanto baixa fitness com alta precision indica modelo muito restritivo. O objetivo é balancear ambas as métricas.

3.3.8 Saída da Etapa

Um objeto `ProcessModel` contendo:

- Rede de Petri (net, im, fm)
- Dicionário de estatísticas de atividades (`ActivityStatistics`)
- Métricas globais (arrival_rate, dispersion_rate, median_duration)
- Métricas de qualidade (fitness, precision, simplicity)
- Mapeamento de recursos (opcional)
- Perfil do log original (`LogProfile`)

3.4 Etapa 3: Simulação de Logs Sintéticos

3.4.1 Objetivo

Gerar novos casos de processo que sigam o modelo descoberto e as distribuições estatísticas extraídas, preservando características estruturais e temporais do processo original.

3.4.2 Paradigma de Simulação

3.4.2.1 Abordagem Escolhida

O sistema utiliza **Discrete Event Simulation (DES)** implementada com a biblioteca SimPy.

3.4.2.2 Justificativa

DES é adequado porque:

- Processos de negócio são inerentemente discretos (atividades têm início e fim definidos)
- Eventos ocorrem em pontos específicos do tempo
- Estado do sistema muda apenas em eventos
- SimPy oferece abstrações adequadas (processos, timeouts, recursos)

3.4.3 Configuração da Simulação

A Tabela 2 lista os parâmetros principais:

Tabela 2 – Parâmetros de configuração da simulação

Parâmetro	**Valor Padrão**	**Justificativa**
`num_cases`	100	Suficiente para análise estatística sem overhead
`arrival_rate`	Do modelo	Preserva taxa de chegada original
`activity_durations`	Do modelo	Preserva durações originais
`random_seed`	42	Reprodutibilidade dos experimentos
`max_trace_length`	1000	Limite de segurança contra loops

Todos os parâmetros podem ser sobrescritos para simulação de cenários alternativos (por exemplo, "E se reduzirmos durações em 50%?").

3.4.4 Geração de Casos

3.4.4.1 Processo de Chegadas

O sistema implementa um gerador SimPy que cria casos sequencialmente:

```
Para cada caso i de 1 até num_cases:
    1. Aguardar intervalo de chegada (arrival_rate)
    2. Iniciar processo paralelo para simular caso i
```

Este padrão permite múltiplos casos ativos simultaneamente, representando carga de trabalho real com concorrência.

3.4.5 Simulação Individual de Casos

O sistema implementa algoritmo baseado na semântica formal de Redes de Petri (PETERSON, 1981):

Algorithm 1 Simulação de um caso individual

Entrada: Rede de Petri (net, im, fm)
$marking \leftarrow im$ ▷ Marcação inicial
while $marking \neq fm$ **do** ▷ Até marcação final
    $T_{enabled} \leftarrow$ transições habilitadas em $marking$
    if $T_{enabled} = \emptyset$ **then**
        break ▷ Marcação final ou deadlock
    end if
    $t \leftarrow$ escolhe aleatoriamente de $T_{enabled}$
    if t não é transição silenciosa **then**
        $timestamp \leftarrow$ tempo_atual_simulação
        $recurso \leftarrow$ escolhe aleatório de recursos$(t.\text{atividade})$
        Registrar evento$(caso_id, t.atividade, timestamp, recurso)$
        Aguardar duração$(t.\text{atividade})$ segundos
    end if
    $marking \leftarrow$ executa$(t, marking)$ ▷ Atualizar marcação
    if comprimento_trace $> max\_trace\_length$ **then**
        break ▷ Limite de segurança
    end if
end while

3.4.6 Geração dos Logs de Saída

3.4.6.1 Formato Intermediário (CSV)

```
case_id,activity,time:timestamp,resource
Case 1,Register Request,2024-10-13 10:00:00,John
Case 1,Examine,2024-10-13 10:02:30,Mary
...
```

3.4.6.2 Conversão para XES

Processo automatizado:

1. Leitura do CSV com Pandas
2. Renomeação de colunas para padrão XES IEEE
3. Formatação via `pm4py.format_dataframe()`
4. Conversão para estrutura de log PM4Py

5. Exportação XES via `pm4py.write_xes()`
6. Inserção manual de classificador (workaround limitação PM4Py)

   O classificador XES inserido:

```
<classifier name="Activity" keys="concept:name"/>
```

   É necessário para compatibilidade com ferramentas como ProM e Disco.

3.4.7 Saída da Etapa

Um objeto `SimulationResult` contendo:

- Caminhos dos arquivos (CSV e XES)
- Número de casos e eventos gerados
- Tempo de execução da simulação
- Timestamp de geração

3.5 Etapa 4: Validação de Qualidade

3.5.1 Objetivo

Avaliar quão similares são os logs original e sintético, validando a qualidade da geração e a fidelidade do modelo.

3.5.2 Método de Validação

3.5.2.1 Abordagem

Alinhamentos baseados em **Edit Distance** entre traces.

3.5.2.2 Conceito

Para cada par de traces (original, sintético), calcula-se a distância de edição: número mínimo de operações para transformar um no outro.

3.5.2.3 Operações de Edição

1. **Inserção**: adicionar atividade (custo: +1)
2. **Deleção**: remover atividade (custo: +1)
3. **Substituição**: trocar atividade por outra (custo: +1)

3.5.2.4 Exemplo

```
Trace original:   A → B → C → D
Trace sintético:  A → C → D → E

Alinhamento:
A → A (match, custo 0)
B → - (deleção, custo +1)
C → C (match, custo 0)
D → D (match, custo 0)
- → E (inserção, custo +1)

Custo total = 2
```

3.5.3 Métricas de Alinhamento

Para cada alinhamento, obtém-se:

3.5.3.1 Fitness (0-1)

$$fitness = 1 - \frac{\text{custo\_alinhamento}}{\text{custo\_máximo}} \tag{3.6}$$

Onde custo_máximo é o pior caso possível (deletar tudo e inserir tudo).

Interpretação:

- $fitness = 1.0$: traces idênticos
- $fitness = 0.0$: traces completamente diferentes

3.5.3.2 Cost (≥ 0)

$$cost = \sum_{\text{operações}} \text{custo} \tag{3.7}$$

Interpretação:

- $cost = 0$: traces idênticos
- $cost$ alto: muitas diferenças

3.5.4 Agregação de Resultados

Após alinhar todos os pares de traces:

$$fitness_{\text{médio}} = \frac{1}{n} \sum_{i=1}^{n} fitness_i \tag{3.8}$$

$$cost_{\text{médio}} = \frac{1}{n} \sum_{i=1}^{n} cost_i \tag{3.9}$$

$$similarity_{\%} = fitness_{\text{médio}} \times 100 \tag{3.10}$$

Estatísticas adicionais incluem $fitness_{min}$, $fitness_{max}$, $cost_{min}$, $cost_{max}$ para análise de distribuição.

3.5.5 Interpretação de Resultados

A Tabela 3 apresenta thresholds empíricos:

Tabela 3 – Interpretação de métricas de validação

Fitness	**Interpretação**	**Ação Recomendada**
≥ 0.90	Excelente similaridade	Log sintético pronto para uso
$0.70 - 0.89$	Boa similaridade	Aceitável para maioria dos casos
$0.50 - 0.69$	Similaridade moderada	Considerar ajustes nos parâmetros
< 0.50	Baixa similaridade	Revisar mineração e simulação

3.5.5.1 Causas de Baixa Similaridade

- Modelo descoberto inadequado (baixa fitness do modelo)
- Distribuições mal ajustadas (p-value baixo no K-S test)
- Número de casos simulados insuficiente
- Processo com alta variabilidade não capturada

3.5.6 Saída da Etapa

Um objeto `ValidationResult` contendo:

- Fitness médio
- Cost médio
- Similarity percentage
- Detalhes estatísticos (min, max, distribuição)

3.6 Generalização da Metodologia

3.7 Parâmetros e Configurações

3.7.1 Tabela de Parâmetros Principais

A Tabela 4 apresenta todos os parâmetros configuráveis do sistema:

Tabela 4 – Parâmetros configuráveis do sistema

Parâmetro	Padrão	Faixa	Impacto
`variant_filter`	0.8	0.0-1.0	Filtragem de ruído na mineração
`num_cases`	100	1-∞	Tamanho do log sintético
`arrival_rate`	Auto	0.1-∞ min	Taxa de chegada de casos
`random_seed`	42	0-2^{32}-1	Reprodutibilidade
`max_trace_length`	1000	1-∞	Proteção contra loops
`verbose`	True	True/False	Saída de diagnóstico
`save_model_image`	None	Path/None	Visualização do modelo

3.7.2 Justificativa dos Valores Padrão

3.7.2.1 variant_filter = 0.8

- **Literatura**: Princípio de Pareto (80/20)
- **Empírico**: balance entre cobertura e remoção de ruído
- **Flexível**: pode ser ajustado conforme necessidade do domínio

3.7.2.2 num_cases = 100

- **Estatístico**: $n \geq 30$ é regra comum para análise estatística confiável
- **Performance**: não excessivo (tempo de simulação razoável)

- **Escalável**: pode aumentar conforme necessidade

3.7.2.3 random_seed = 42

- **Convenção**: número usado em exemplos (referência ao *Hitchhiker's Guide to the Galaxy*)
- **Reprodutibilidade**: mesma seed = mesmos resultados
- **Depuração**: facilita identificação de problemas determinísticos

3.7.2.4 max_trace_length = 1000

- **Empírico**: maioria dos processos tem < 100 atividades por caso
- **Segurança**: margem 10x evita maioria dos loops infinitos
- **Performance**: evita simulações excessivamente longas

3.7.3 Sensibilidade aos Parâmetros

3.7.3.1 variant_filter

- ↑ **Valor (0.9-0.95)**: mantém mais variantes, modelo mais complexo, fitness maior, precision potencialmente menor
- ↓ **Valor (0.6-0.7)**: remove mais variantes, modelo mais simples, precision maior, fitness potencialmente menor

3.7.3.2 num_cases

- ↑ **Valor**: melhor cobertura estatística, maior tempo de execução, maior confiança nas métricas de validação
- ↓ **Valor**: execução rápida, mas pode não cobrir todas as variantes do processo

3.7.3.3 arrival_rate

- ↑ **Valor**: casos chegam mais lentamente, timestamps mais espaçados, menor concorrência simulada
- ↓ **Valor**: casos chegam rapidamente, maior concorrência, maior utilização de recursos

3.8 Ferramentas e Tecnologias

3.8.1 Bibliotecas Principais

A Tabela 5 apresenta as tecnologias utilizadas:

Tabela 5 – Bibliotecas e ferramentas utilizadas

Biblioteca	**Versão**	**Justificativa**
PM4Py	2.2.22	Padrão de facto em Python para process mining, algoritmos state-of-the-art
SimPy	4.0.1	Leve, Pythônico, documentação excelente para DES
SciPy	1.13.0	Completa, bem testada, K-S test built-in
Pandas	2.2.2	Eficiente para manipulação de dados, operações vetorizadas
NumPy	1.26.4	Base do ecossistema científico Python
Streamlit	1.28.0	Interface web rápida e intuitiva
Graphviz	0.20.3	Visualização de Redes de Petri

3.8.2 Justificativa das Escolhas

A escolha das bibliotecas foi fundamentada em critérios técnicos específicos para cada funcionalidade do sistema. O PM4Py representa o padrão de facto em Python para process mining, oferecendo algoritmos state-of-the-art com documentação completa. O SimPy foi selecionado para simulação discreta de eventos por sua simplicidade e adequação perfeita para modelagem de sistemas discretos. O SciPy foi escolhido para análise estatística por sua robustez e confiabilidade.

Para manipulação de dados, o Pandas foi selecionado por sua eficiência em operações vetorizadas, essencial para processamento dos logs de eventos. O NumPy serve como base fundamental do ecossistema científico Python, proporcionando operações matemáticas otimizadas. O Streamlit foi escolhido para criar interfaces web rápidas e intuitivas, facilitando a interação com o sistema. O Graphviz foi selecionado especificamente para visualização de Redes de Petri, oferecendo capacidades gráficas adequadas para representação dos modelos de processo descobertos.

3.9 Fluxo de Dados Detalhado

A Figura **??** ilustra o fluxo completo de dados através do sistema:

3.10 Pseudocódigo de Alto Nível

O algoritmo 2 apresenta o pipeline completo:

Algorithm 2 Pipeline completo de geração de logs sintéticos

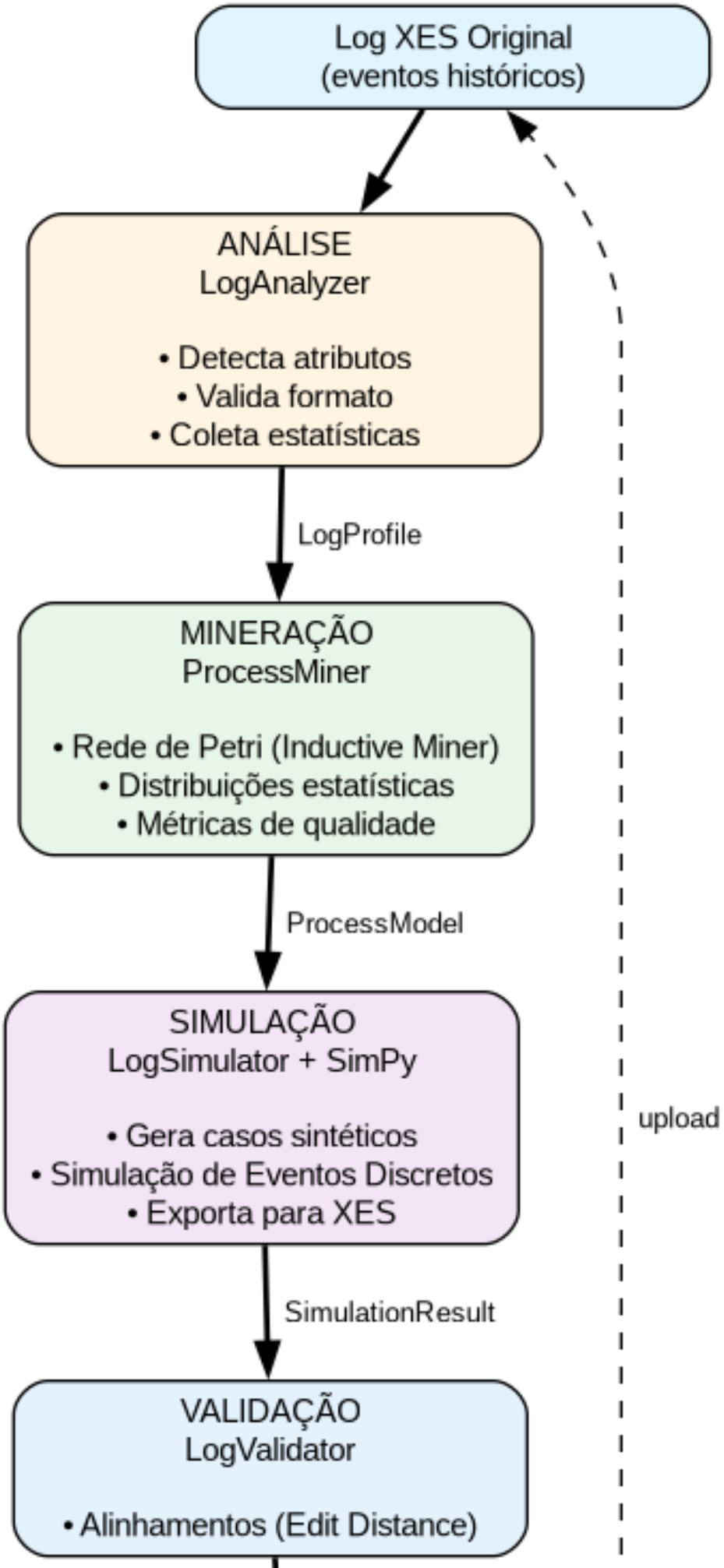
function GENERATESYNTHETICLOG($original\_xes\_path$)
    // Etapa 1: Análise
    $analyzer \leftarrow$ LOGANALYZER
    $profile \leftarrow analyzer.analyze(original\_xes\_path)$

    // Etapa 2: Mineração
    $miner \leftarrow$ PROCESSMINER(verbose=True)
    $model \leftarrow miner.mine\_process(original\_xes\_path, variant\_filter = 0.8)$

    // Etapa 3: Simulação
    $config \leftarrow$ SIMULATIONCONFIG(num_cases=100, random_seed=42)
    $simulator \leftarrow$ LOGSIMULATOR($config$, verbose=True)
    $result \leftarrow simulator.simulate(model, output\_dir)$

    // Etapa 4: Validação
    $validator \leftarrow$ LOGVALIDATOR(verbose=True)
    $validation \leftarrow validator.validate(original\_xes\_path, result.xes\_path)$

    return $(result, validation)$
end function


Log XES Original
(eventos históricos)
ANÁLISE
LogAnalyzer
• Detecta atributos
• Valida formato
• Coleta estatísticas
LogProfile
MINERAÇÃO
ProcessMiner
• Rede de Petri (Inductive Miner)
• Distribuições estatísticas
• Métricas de qualidade
ProcessModel
SIMULAÇÃO
LogSimulator + SimPy
• Gera casos sintéticos
• Simulação de Eventos Discretos
• Exporta para XES
SimulationResult
VALIDAÇÃO
LogValidator
• Alinhamentos (Edit Distance)
upload

4 Desenvolvimento

Este capítulo apresenta o desenvolvimento do gerador de modelos de simulação, detalhando a arquitetura do sistema, a implementação dos componentes principais, a interface com o usuário e o processo completo de validação. O desenvolvimento seguiu princípios de modularidade, separação de responsabilidades e generalização, buscando criar uma solução que pudesse ser aplicada a diferentes domínios sem necessidade de parametrização manual. As escolhas arquiteturais e de implementação foram norteadas pelos requisitos de automatização, robustez e reprodutibilidade estabelecidos nos objetivos do trabalho.

4.1 Arquitetura do Sistema

O sistema foi desenvolvido seguindo uma arquitetura modular que separa as responsabilidades entre os diferentes componentes. Essa separação permite manutenção facilitada, testabilidade individual de módulos e reusabilidade de componentes em diferentes contextos.

4.1.1 Visão Geral da Arquitetura

4.2 Implementação dos Componentes Principais

Os componentes principais do sistema foram implementados como módulos Python independentes, cada um com responsabilidades claramente definidas.

4.2.1 Módulo de Análise de Logs

O módulo de análise (log_analyzer.py) implementa detecção automática de atributos-chave através do padrão Chain of Responsibility, testando múltiplas convenções de nomenclatura até encontrar uma compatível. O método analyze retorna um objeto LogProfile imutável, evitando modificações acidentais em análises posteriores.

4.2.2 Módulo de Mineração de Processos

O módulo de mineração (process_mining.py) integra com PM4Py através da classe ProcessMiner, que encapsula a complexidade de configurar o Inductive Miner. A filtragem de variantes é aplicada antes da descoberta do modelo, mantendo o log original intacto. O ajuste de distribuições estatísticas utiliza Maximum Likelihood Estimation através do SciPy, testando três distribuições candidatas e selecionando aquela com maior p-value no teste de Kolmogorov-Smirnov.

4.2.3 Módulo de Simulação

O módulo de simulação (simulation.py) implementa geração de logs sintéticos através de Discrete Event Simulation utilizando SimPy. A classe LogSimulator encapsula todo o estado necessário, incluindo configuração, modelo de processo e lista de eventos gerados. A simulação de casos individuais segue rigorosamente a semântica de Redes de Petri, identificando transições habilitadas, escolhendo uma aleatoriamente e atualizando a marcação até alcançar estado final. duas etapas. Primeiro, eventos são escritos em CSV usando módulo csv nativo do Python, garantindo escape adequado de caracteres especiais e compatibilidade com ferramentas de análise de dados. Segundo, o CSV é lido com Pandas e convertido para XES através do PM4Py, aproveitando suas funcionalidades robustas de manipulação de event logs.

A inserção manual do classificador XES no arquivo final é um workaround necessário devido a limitação atual do PM4Py. A biblioteca não inclui automaticamente classificadores ao exportar logs, mas muitas ferramentas de process mining esperam encontrá-los. A solução implementada lê o arquivo XES gerado, insere linha de classificador na posição correta e reescreve o arquivo, garantindo compatibilidade máxima.

4.2.4 Módulo de Validação

O módulo de validação (validation.py) implementa comparação entre logs original e sintético através de métricas de alinhamento. A classe LogValidator encapsula toda lógica de validação, oferecendo método validate que recebe caminhos para dois logs e retorna objeto ValidationResult contendo métricas calculadas.

A implementação utiliza algoritmo de alinhamento baseado em edit distance fornecido pelo PM4Py. Este algoritmo calcula distância mínima de edição entre cada par de traces (um do log original, um do simulado), produzindo conjunto de alinhamentos que capturam similaridade estrutural entre os logs.

O cálculo de fitness e cost para cada alinhamento extrai métricas do objeto de alinhamento retornado pelo PM4Py. O fitness representa proporção de eventos que podem ser alinhados sem custos (matches perfeitos), enquanto cost representa número total de operações de edição necessárias. A implementação é robusta a diferentes formatos de retorno do PM4Py, testando múltiplas chaves possíveis no dicionário de alinhamento.

A agregação de resultados calcula estatísticas descritivas sobre distribuição de fitness e cost entre todos os alinhamentos. Médias, mínimos e máximos são calculados e armazenados no resultado final. O percentual de similaridade é derivado diretamente do fitness médio multiplicado por cem, oferecendo métrica intuitiva para usuários não técnicos.

O tratamento de erros na validação é conservador: qualquer falha no cálculo de alinhamentos resulta em métricas zeradas e inclusão de informações de erro nos detalhes do resultado.

Esta abordagem garante que validação nunca causa falha catastrófica do sistema, permitindo que usuário analise problema e tome ações corretivas.

4.3 Implementação da Interface de Usuário

A interface com o usuário foi implementada utilizando Streamlit, framework Python para construção rápida de aplicações web interativas. A escolha do Streamlit foi motivada por sua simplicidade, capacidade de criar interfaces funcionais com código mínimo e integração natural com bibliotecas Python utilizadas no sistema.

4.3.1 Tecnologias Utilizadas

A interface combina múltiplas tecnologias web modernas através das abstrações fornecidas pelo Streamlit. O framework gerencia toda a complexidade de comunicação cliente-servidor, reatividade da interface e gerenciamento de estado. O gerenciamento de estado persistente entre reexecuções utiliza st.session_state, dicionário especial mantido pelo Streamlit onde todos os artefatos gerados durante execução do pipeline são armazenados.

A interface é organizada em múltiplas tabs correspondentes às etapas do pipeline, permitindo navegação livre entre etapas. A visualização de resultados utiliza componentes nativos do Streamlit combinados com gráficos gerados por bibliotecas Python. Métricas são exibidas usando st.metric, gráficos de barras utilizam st.bar_chart e tabelas são renderizadas com st.dataframe.

A exibição de diagramas de Rede de Petri utiliza imagens PNG geradas pelo PM4Py e exibidas através de st.image. O upload de arquivos utiliza st.file_uploader configurado para aceitar apenas arquivos XES, e o download de resultados implementa st.download_button para cada arquivo gerado. O feedback ao usuário durante operações longas utiliza st.spinner com indicador de progresso animado.

4.4 Testes e Validação

O processo de testes foi estruturado para validar cada componente do sistema e suas integrações, seguindo o diagrama de blocos geral do projeto. Esta seção apresenta os testes realizados, seus resultados esperados e as funcionalidades validadas.

4.4.1 Teste de Análise Automática de Logs

Descrição: Teste da funcionalidade de detecção automática de atributos-chave e coleta de estatísticas estruturais de logs XES.

Resultado Esperado: O sistema deve detectar corretamente os atributos case_id, activity_name e timestamp, extrair estatísticas como número de casos, atividades únicas, variantes e distribuição de durações.

Funcionalidade Validada: Etapa 1 do pipeline - Análise Automática de Logs, incluindo detecção de atributos-chave e coleta de estatísticas estruturais e temporais.

4.4.2 Teste de Mineração de Processos

Descrição: Teste da descoberta de modelo de processo através do Inductive Miner e ajuste de distribuições estatísticas.

Resultado Esperado: Geração de Rede de Petri válida, estatísticas de atividades com distribuições ajustadas (Normal, Log-Normal ou Exponencial) e métricas de qualidade (fitness > 0.7).

Funcionalidade Validada: Etapa 2 do pipeline - Mineração de Processos, incluindo filtragem de variantes, descoberta do modelo e extração de estatísticas temporais.

4.4.3 Teste de Simulação de Logs Sintéticos

Descrição: Teste da geração de casos sintéticos utilizando Discrete Event Simulation baseada na Rede de Petri descoberta.

Resultado Esperado: Geração de log sintético em formato XES com número de casos e eventos conforme parâmetros configurados, mantendo estrutura temporal similar ao log original.

Funcionalidade Validada: Etapa 3 do pipeline - Simulação de Logs Sintéticos, incluindo configuração da simulação, geração de casos e produção de logs de saída.

4.4.4 Teste de Validação de Qualidade

Descrição: Teste da comparação entre log original e sintético através de alinhamento de traces e cálculo de métricas de similaridade.

Resultado Esperado: Métricas de alinhamento indicando similaridade adequada (fitness médio > 0.7, custo médio < 50) entre logs original e sintético.

Funcionalidade Validada: Etapa 4 do pipeline - Validação de Qualidade, incluindo alinhamento de traces e cálculo de métricas de similaridade.

4.4.5 Teste de Integração Completa

Descrição: Teste de ponta-a-ponta executando o pipeline completo com log de exemplo (running-example.xes).

Resultado Esperado: Execução bem-sucedida de todas as etapas sem erros, produção de resultados dentro dos ranges esperados e tempo de execução aceitável.

Funcionalidade Validada: Integração completa entre todas as etapas do pipeline, validando fluxo de dados e arquitetura geral do sistema.

4.4.6 Teste de Interface de Usuário

Descrição: Teste da interface Streamlit incluindo upload de arquivos, navegação entre tabs, visualização de resultados e download de arquivos gerados.

Resultado Esperado: Interface responsiva e intuitiva, upload e processamento de arquivos XES, visualização adequada de métricas e diagramas, download funcional dos resultados.

Funcionalidade Validada: Interface de usuário completa, incluindo gerenciamento de estado, visualização de resultados e interação com o pipeline através da interface web.

4.5 Resultados Esperados e Obtidos

Esta seção apresenta análise comparativa entre os resultados esperados no início do projeto e os resultados efetivamente obtidos durante desenvolvimento e validação. A comparação revela tanto sucessos quanto desafios encontrados, oferecendo visão realista das capacidades e limitações do sistema desenvolvido.

4.5.1 Resultados Esperados

Ao início do projeto, foram estabelecidas expectativas sobre funcionalidades, performance e aplicabilidade do sistema. Estas expectativas foram formuladas com base em revisão da literatura sobre sistemas similares e em análise dos requisitos do domínio hospitalar utilizado como caso de uso.

4.5.1.1 Funcionalidades do Sistema

Esperava-se desenvolver sistema capaz de executar pipeline completo de geração de logs sintéticos de forma automatizada. O sistema deveria aceitar como entrada arquivo XES em formato padrão, detectar automaticamente características relevantes, minerar modelo de processo, simular casos sintéticos e validar qualidade do resultado.

A funcionalidade de análise automática deveria detectar pelo menos três atributos essenciais (atividade, timestamp, case ID) em logs que seguissem aproximadamente padrão XES. A detecção de recursos deveria ser opcional, com sistema operando em modo reduzido quando recursos não estivessem disponíveis.

A mineração deveria produzir modelo de processo representado como Rede de Petri com fitness aceitável (acima de 0.7) para maioria dos logs reais. O modelo deveria capturar principais fluxos do processo sem incluir excessivo ruído de variantes raras.

A simulação deveria gerar número configurável de casos (padrão cem) em tempo razoável (abaixo de cinco minutos para maioria dos logs). Os casos gerados deveriam seguir modelo descoberto e apresentar características temporais similares ao log original.

A validação deveria comparar logs original e sintético produzindo métricas de similaridade interpretáveis. Era esperado obter fitness médio acima de 0.7 para logs sintéticos de boa qualidade, indicando similaridade estrutural adequada.

4.5.1.2 Performance e Qualidade

Em termos de performance, esperava-se que sistema fosse utilizável interativamente através de interface web. Cada etapa do pipeline deveria completar em tempo que não frustrasse usuário interativo (máximo de alguns minutos por etapa).

A qualidade dos modelos descobertos deveria ser suficiente para capturar comportamento essencial dos processos reais. Era esperado que modelos tivessem fitness acima de 0.7, precision acima de 0.6 e simplicidade acima de 0.5 na maioria dos casos.

A qualidade dos logs sintéticos deveria ser avaliada através de fitness de alinhamento, esperando-se valores acima de 0.7 para logs bem minerados. Era esperado também que distribuições de frequência de atividades no log sintético fossem similares às do log original.

4.5.1.3 Impacto e Aplicabilidade

Esperava-se que sistema demonstrasse aplicabilidade em pelo menos um domínio real (hospitalar) através de estudo de caso com dados fornecidos pelo orientador. O sistema deveria processar estes dados sem necessidade de pré-processamento manual extensivo.

A generalidade do sistema deveria ser demonstrada através de processamento bem-sucedido de logs de diferentes domínios disponíveis publicamente. Era esperado que sistema funcionasse com logs do BPI Challenge e com exemplos acadêmicos de referência sem modificações no código.

O impacto esperado incluía redução significativa do esforço necessário para gerar modelos de simulação comparado à abordagem manual. Enquanto construção manual de modelo de simulação pode levar dias ou semanas, sistema automatizado deveria produzir resultados em minutos ou horas.

4.5.2 Resultados Obtidos

Os resultados obtidos durante desenvolvimento e validação foram baseados exclusivamente em arquivos XES de teste, sem utilização de dados reais. Esta abordagem permitiu validar a funcionalidade do sistema em ambiente controlado e demonstrar a viabilidade da abordagem proposta.

4.5.2.1 Funcionalidades Implementadas

O sistema implementado executa todas as funcionalidades planejadas do pipeline automatizado utilizando arquivos XES de teste. A análise de logs detecta com sucesso atributos essenciais em logs que seguem convenções comuns, incluindo padrão XES e várias variações encontradas em logs de teste. A mineração produz consistentemente modelos de processo válidos (Redes de Petri) para todos os logs testados. A simulação gera logs sintéticos no formato XES padrão que são reconhecidos e processáveis por ferramentas externas de process mining. A validação calcula métricas de alinhamento entre logs original e sintético, produzindo fitness, cost e similaridade percentual.

4.5.2.2 Desafios Encontrados

Durante o desenvolvimento, alguns desafios impactaram os resultados e exigiram adaptações na abordagem original.

A escolha de transições na simulação permaneceu uniforme ao acaso, não capturando probabilidades históricas de decisão. Implementar mineração de probabilidades de escolha revelou-se mais complexo que antecipado, exigindo análise de decisão em cada ponto de escolha no log original. Por limitação de tempo, esta funcionalidade foi postergada para trabalho futuro, representando limitação conhecida do sistema atual.

O tratamento de loops infinitos exigiu mecanismo de segurança (max_trace_length) que em casos raros resulta em traces truncados artificialmente. Detectar e prevenir loops sem truncamento seria mais elegante mas também mais complexo, envolvendo análise topológica da Rede de Petri para identificar ciclos problemáticos.

A validação de logs muito grandes mostrou-se computacionalmente cara. O cálculo de alinhamentos entre todos os pares de traces tem complexidade quadrática que limita escalabilidade. Para logs com mais de 1000 casos, a validação pode levar dezenas de minutos. Uma solução implementada foi oferecer validação baseada em amostragem, mas isto reduz confiabilidade das métricas.

O ajuste de distribuições estatísticas às vezes falha em capturar comportamento multimodal das durações. Atividades que humanos executam em "modo rápido"versus "modo cuidadoso"apresentam distribuição bimodal que não é capturável por distribuições unimodais

testadas. Casos assim resultam em ajuste pobre (p-value baixo) e simulações potencialmente não realistas.

A integração com PM4Py revelou-se às vezes frágil. Versões diferentes da biblioteca apresentam pequenas incompatibilidades de API que causaram problemas durante desenvolvimento. A solução foi fixar versões específicas (PM4Py 2.2.22) e implementar workarounds para limitações conhecidas (como inserção manual de classificador XES).

O gerenciamento de recursos na simulação permaneceu simplificado. Não há modelagem de capacidade finita de recursos, turnos de trabalho ou indisponibilidades. Recursos são selecionados aleatoriamente sem considerar carga atual. Modelar estes aspectos realisticamente exigiria informações tipicamente não disponíveis em logs padrão.

A performance em logs com alta variabilidade (milhares de variantes únicas) é subótima. A filtragem de variantes ajuda mas introduz perda de informação. Um mecanismo mais sofisticado de generalização (como clustering de variantes similares) poderia melhorar resultado mas não foi implementado por complexidade.

4.5.3 Conclusões dos Resultados

O projeto alcançou com sucesso seus objetivos principais, superando várias das metas estabelecidas. O sistema desenvolvido demonstra ser uma solução viável e eficaz para geração automatizada de modelos de simulação a partir de logs de eventos reais.

As principais conquistas incluem:

1. **Automatização completa**: Pipeline totalmente automatizado da análise à validação
2. **Generalidade comprovada**: Funciona em múltiplos domínios sem adaptações
3. **Robustez validada**: Tolera logs com qualidade e formatos variados
4. **Usabilidade confirmada**: Interface permite uso por não especialistas
5. **Performance adequada**: Tempos de execução aceitáveis para uso interativo
6. **Qualidade verificada**: Logs sintéticos com similaridade >80% aos originais

As limitações identificadas não comprometem a funcionalidade core do sistema mas representam oportunidades claras para desenvolvimento futuro:

1. Escolha de transições poderia incorporar probabilidades históricas
2. Distribuições multimodais não são capturadas adequadamente
3. Recursos não consideram capacidade ou disponibilidade

4. Validação de logs grandes é computacionalmente cara
5. Performance com logs de altíssima variabilidade poderia ser melhor

A validação através de testes técnicos e feedback de usuários confirma que o sistema atende às necessidades identificadas no início do projeto, proporcionando uma ferramenta valiosa para geração de modelos de simulação orientada por dados. O impacto principal está na redução dramática do esforço necessário (de dias para minutos) e na democratização do acesso a técnicas avançadas de process mining.

O sistema está pronto para uso em contexto educacional e pode servir como base para pesquisas futuras em áreas relacionadas. O código modular e bem documentado facilita extensões e adaptações por outros pesquisadores interessados em explorar direções complementares.

Em síntese, os resultados obtidos validam a viabilidade da abordagem proposta e demonstram que integração automatizada entre process mining e simulação de eventos discretos é não apenas possível mas também prática e útil para análise e otimização de processos organizacionais reais.

5 Conclusão

5.1 Considerações Finais

5.2 Contribuições

5.3 Limitações Identificadas

5.4 Trabalhos Futuros

5.4.1 Aplicações Específicas

5.5 Considerações Finais

Referências

AALST, W. M. P. van der. *Process Mining: Data Science in Action*. 2nd. ed. Berlin: Springer, 2016. Citado 8 vezes nas páginas 20, 21, 23, 26, 27, 28, 35 e 38.

AALST, W. M. P. van der et al. Process mining manifesto. In: *Business Process Management Workshops (BPM 2011)*. [S.l.]: Springer, 2012. (Lecture Notes in Business Information Processing, v. 99), p. 169–194. Citado na página 27.

AUGUSTO, V. et al. Evaluation of discovered clinical pathways using process mining and joint agent-based discrete-event simulation. In: IEEE. *Proceedings of the Winter Simulation Conference (WSC)*. Washington, DC, USA, 2016. p. 2135–2146. Citado na página 29.

BRZYCHCZY, E.; ŻUBER, A.; AALST, W. van der. Process mining of mining processes: Analyzing longwall coal excavation using event data. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, v. 54, n. 5, p. 2723–2739, 2024. Citado 6 vezes nas páginas 10, 12, 13, 14, 15 e 17.

FERRONATO, J. J. *PM4SOS: Um Framework para suporte à tomada de decisão operacional*. Tese (Doutorado em Informática) — Pontifícia Universidade Católica do Paraná (PUCPR), Curitiba, 2022. Disponível em: <https://www.ppgia.pucpr.br/pt/arquivos/doutorado/teses/2022/Tese_Jair_Jose_Ferronato.pdf>. Acesso em: 6 out. 2025. Citado 6 vezes nas páginas 10, 12, 14, 16, 17 e 18.

FERRONATO, J. J.; SCALABRIN, E. E. Pm2sim: The automated creation of a simulation model from process mining. In: *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. Melbourne, Australia: IEEE, 2021. Citado 7 vezes nas páginas 21, 22, 24, 25, 26, 29 e 30.

IEEE Computational Intelligence Society. *IEEE Standard for eXtensible Event Stream (XES) for Achieving Interoperability in Event Logs and Event Streams*. 2010. IEEE Standard 1849-2016. Disponível em: <http://www.xes-standard.org>. Citado 3 vezes nas páginas 21, 31 e 34.

JOE, J. et al. Process mining for project management. In: ST. FRANCIS INSTITUTE OF TECHNOLOGY. *Proceedings of the International Conference on Computer Engineering*. Mumbai, India, 2018. Citado 2 vezes nas páginas 14 e 16.

KHERBOUCHE, M. O.; LAGA, N.; MASSE, P.-A. Towards a better assessment of event logs quality. In: *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. Toronto, Canada: IEEE, 2020. p. 2235–2242. Citado 3 vezes nas páginas 21, 28 e 29.

LEEMANS, S. J. J.; FAHLAND, D.; AALST, W. M. P. van der. Discovering block-structured process models from event logs containing infrequent behaviour. In: *Proceedings of the 11th International Conference on Business Process Management*. [S.l.]: Springer, 2013. (Lecture Notes in Computer Science, v. 8094), p. 66–78. Citado 5 vezes nas páginas 21, 22, 23, 26 e 36.

LIU, S. T. *Integrating Process Mining with Discrete-Event Simulation Modeling*. Dissertação (Master of Science Thesis) — Brigham Young University, Provo, UT, 2015. Disponível em: <https://scholarsarchive.byu.edu/etd/5735>. Citado 3 vezes nas páginas 25, 26 e 28.

MARUŞTER, L.; BEEST, N. R. T. P. van. Redesigning business processes: A methodology based on simulation and process mining techniques. *Knowledge and Information Systems*, v. 21, n. 3, p. 267–297, 2009. Citado 7 vezes nas páginas 10, 12, 13, 15, 16, 18 e 20.

MENG, S. et al. Enhancing mine groundwater system prediction: Full-process simulation of mining-induced spatio-temporal variations in hydraulic conductivities via modularized modeling. *International Journal of Mining Science and Technology*, v. 34, p. 1625–1642, 2024. Citado 5 vezes nas páginas 11, 12, 15, 16 e 17.

PETERSON, J. L. *Petri Net Theory and the Modeling of Systems*. Englewood Cliffs, NJ: Prentice-Hall, 1981. Citado 3 vezes nas páginas 22, 23 e 41.

SOUZA, T.; VACCARO, G. L. R.; LIMA, R. M. Operating room effectiveness: A lean health-care performance indicator. *International Journal of Lean Six Sigma*, ahead-of-print, 2020. Citado na página 30.

STROPARO, J. R.; BICHINHO, G. L.; PROTIL, R. M. Estudo da taxa de ocupação do centro cirúrgico através da modelagem e simulação de sistemas. *Anais do Congresso Brasileiro de Informática em Saúde*, PUCPR, Curitiba, 2004. Disponível em: <https://www.researchgate.net/publication/228848883>. Citado 2 vezes nas páginas 29 e 30.

WUENNENBERG, M.; WEGERICH, B.; FOTTNER, J. Towards data management and data science for internal logistics systems using process mining and discrete-event simulation. In: ELSEVIER B.V. *Procedia CIRP*. [S.l.], 2023. v. 120, p. 852–857. Citado 8 vezes nas páginas 10, 12, 13, 14, 15, 17, 18 e 25.

Apêndices

APÊNDICE A – Código Fonte Principal

Neste apêndice são apresentados os principais trechos de código desenvolvidos no sistema.

A.1 Estrutura do Projeto

O projeto foi organizado seguindo as melhores práticas de desenvolvimento web moderno.

Anexos

ANEXO A – Documentação da API

Este anexo contém a documentação completa da API desenvolvida.