UNIVERSIDADE DE SÃO PAULO
ESCOLA DE ARTES, CIÊNCIAS E HUMANIDADES
PROGRAMA DE PÓS-GRADUAÇÃO EM SISTEMAS DE INFORMAÇÃO

GABRIEL LUCAS CANTANHEDE DA SILVA

**Aplicação de algoritmos genéticos em mineração de processos não
estruturados**

São Paulo

2018

GABRIEL LUCAS CANTANHEDE DA SILVA

**Aplicação de algoritmos genéticos em mineração de processos não
estruturados**

Versão original

Dissertação apresentada à Escola de Artes, Ciências e Humanidades da Universidade de São Paulo para obtenção do título de Mestre em Ciências pelo Programa de Pós-graduação em Sistemas de Informação.

Área de concentração: Metodologia e Técnicas da Computação

Orientador: Prof. Dr. Marcelo Fantinato

Coorientador: Prof.^a Dr.^a Sarajane Marques Peres

São Paulo

2018

Autorizo a reprodução e divulgação total ou parcial deste trabalho, por qualquer meio convencional ou eletrônico, para fins de estudo e pesquisa, desde que citada a fonte.

CATALOGAÇÃO-NA-PUBLICAÇÃO

(Universidade de São Paulo. Escola de Artes, Ciências e Humanidades. Biblioteca)
CRB-8 4625

Silva, Gabriel Lucas Cantanhede da

Aplicação de algoritmos genéticos em mineração de processos não estruturados / Gabriel Lucas Cantanhede da Silva ; orientador, Marcelo Fantinato ; coorientadora, Sarajane Marques Peres . – 2018.

106 f. : il.

Dissertação (Mestrado em Ciências) - Programa de Pós-Graduação em Sistemas de Informação, Escola de Artes, Ciências e Humanidades, Universidade de São Paulo.

Versão original

1. Mineração de dados. 2. Negócios - Processos. I. Fantinato, Marcelo, orient. II. Peres, Sarajane Marques, coorient. III. Título.

CDD 22.ed.– 006.312

Dissertação de autoria de Gabriel Lucas Cantanhede da Silva, sob o título “**Aplicação de algoritmos genéticos em mineração de processos não estruturados**”, apresentada à Escola de Artes, Ciências e Humanidades da Universidade de São Paulo, para obtenção do título de Mestre em Ciências pelo Programa de Pós-graduação em Sistemas de Informação, na área de concentração Metodologia e Técnicas da Computação, aprovada em _____ de _____ de _____ pela comissão julgadora constituída pelos doutores:

Prof. Dr. _____

Instituição: _____

Presidente

Prof. Dr. _____

Instituição: _____

Prof. Dr. _____

Instituição: _____

Prof. Dr. _____

Instituição: _____

À minha amada mãe, cujo amor e apoio incondicionais me sustentaram mediante as adversidades, estando comigo mesmo quando distantes um do outro.

Agradecimentos

Agradeço a minha família, a qual me apoiou mesmo antes da notícia de aprovação no mestrado, e que suportaram a vida à distância enquanto eu trilhava o caminho para realizar meus sonhos. Aos amigos que deixei em São Luís, cuja amizade pude desfrutar mesmo longe, e aos amigos que fiz no PPgSI, com os quais eu tive o prazer de compartilhar exercícios, anotações, várias xícaras de café e inúmeros momentos descontraídos.

Ao Prof. Marcelo Fantinato, sem o qual o meu mestrado não teria sido possível, pois além de ser um orientador exemplar e possuir exímios conhecimentos científicos, soube tirar de mim o melhor mesmo em tempos difíceis, propiciando valiosas lições de vida. À Prof.^a Sarajane Marques Peres, cujo talento e empolgação características me fizeram mudar totalmente de perspectiva no mestrado, fazendo eu descobrir e me interessar cada vez mais pela área de inteligência computacional.

Aos professores e funcionários da Escola de Artes, Ciências e Humanidades da Universidade de São Paulo, pois mesmo em pequena ou grande parte todos contribuíram para que esse sonho fosse possível.

*“Não digas: ‘Eu sou assim..., são coisas do meu caráter’.
São coisas da tua falta de caráter. Sê homem! - Esto vir! ”*
(São Josemaria Escrivá)

Resumo

SILVA, Gabriel Lucas Cantanhede da. **Aplicação de algoritmos genéticos em mineração de processos não estruturados**. 2018. 106 f. Dissertação (Mestrado em Ciências) – Escola de Artes, Ciências e Humanidades, Universidade de São Paulo, São Paulo, 2018.

Mineração de processos é um novo campo de pesquisa que liga mineração de dados e gestão de processos de negócio. A mineração de processos segue a premissa de que existe um processo desconhecido em um determinado contexto, e que ao analisar os traços do seu comportamento, com o auxílio da mineração de dados, é possível descobrir o modelo do processo. No entanto, processos de negócio realistas são difíceis de minerar por causa do excesso de comportamento registrado nos logs. Esses processos não estruturados, apesar de complexos, possuem um potencial grande para melhoria, sendo que as abordagens atuais de mineração de processos para esse contexto ainda provém pouco suporte à gestão. Este trabalho de pesquisa de mestrado visou aplicar técnicas computacionais evolutivas na mineração de modelos de processo, usando algoritmos genéticos para descobrir automaticamente modelos de processos não estruturados visando dar suporte à gestão organizacional de processos. Uma revisão da literatura foi realizada para auxiliar a proposição de uma nova abordagem focada na descoberta de modelos de processos não estruturados. A abordagem proposta introduz novas fórmulas de cálculo das métricas de completude e precisão baseadas nas informações de transições entre atividades, reorganizadas por meio de uma estrutura de matriz criada neste trabalho. A abordagem introduz também o uso de operadores genéticos e estratégias de fluxo evolutivo ainda não implementados na literatura relativa a algoritmos genéticos na descoberta de processos. Análises da parametrização da abordagem proposta, bem como os modelos de processos resultantes, indicam que a abordagem é eficaz na mineração de modelos de processos melhores a partir de amostras de um log não estruturado.

Palavras-chaves: Gestão de Processos de Negócio. Processos não Estruturados. Mineração de Processos. Mineração de Dados. Técnicas Evolutivas. Algoritmos Genéticos.

Abstract

SILVA, Gabriel Lucas Cantanhede da. **Application of genetic algorithms on mining non structured processes**. 2018. 106 p. Dissertation (Master of Science) – School of Arts, Sciences and Humanities, University of São Paulo, São Paulo, 2018.

Process mining is a new field of research that links data mining and business process management. Process mining follows the premise that there is an unknown process in a given context, and by analyzing the traces of it's behavior, with the help of data mining, the process model can be discovered. However, realistic business processes are difficult to mine because of excessive behavior recorded in the logs. These unstructured processes, despite being complex, hold great potential for improvement, and the current process mining approaches for that context yet provide little support for management. This master's research project aims to apply evolutionary computational techniques in process mining, using genetic algorithms to automatically discover unstructured process models in order to support process management in organizations. A literature review was carried out to support the proposition of a new approach focused on the discovery of unstructured process models. The proposed approach introduces new formulas for calculating completeness and precision metrics, based on the information of transitions between activities that are reorganized through a matrix structure created in this work. The approach also introduces the use of genetic operators and evolutionary flow strategies not yet implemented in the literature regarding genetic algorithms in process discovery. Analyzes of the parameterization of the proposed approach, as well as the resulting process models, indicate that the approach is effective in mining better process models from samples of a unstructured log.

Keywords: Business Process Management. Unstructured Processes. Process Mining. Data Mining. Evolutive Techniques. Genetic Algorithms.

Lista de figuras

Figura 1 – Exemplo de modelagem usando redes de Petri	27
Figura 2 – Exemplo de modelo de processo, representado em WF-net	29
Figura 3 – Posicionamento dos três principais tipos de mineração de processos: descoberta, conformidade e melhoria	44
Figura 4 – Exemplo de processo descoberto a partir de L1, modelado usando BPMN	48
Figura 5 – Exemplo de um modelo de processo minerado a partir do <i>log</i> de processo não estruturado	52
Figura 6 – Distribuição de estudos primários por ano de publicação	56
Figura 7 – Saída do plugin ETM-ProM para a amostra de 10 traços de eventos . .	85
Figura 8 – Saída do plugin ETM-ProM para a amostra de 10 traços de eventos, com simplicidade = 5	85
Figura 9 – Modelos de processo minerados para os logs com 10, 20 e 35 traços, respectivamente (modelados em BPMN na ferramenta Signavio Academic).	91
Figura 10 – Evolução do <i>fitness</i> para o log com 10 traços em laranja, 20 traços em azul e 35 traços em verde.	92

Lista de algoritmos

Algoritmo 1 – Algoritmo evolutivo	39
---	----

Lista de tabelas

Tabela 1 – Log compacto de eventos relativos a $L1 = [(a,b,c,d)^3, (a,c,b,d)^2, (a,e,d)]$	47
Tabela 2 – Relação entre desafios e técnicas empregadas na descoberta de processos	51
Tabela 3 – Representação cromossômica de um modelo de processo	68
Tabela 4 – Log de um processo estruturado simples, referente ao modelo de processo da Tabela 3	69
Tabela 5 – Matriz B/A (para o log na Tabela 4)	70
Tabela 6 – Exemplo de <i>crossover</i> de um ponto	73
Tabela 7 – Exemplo de <i>crossover</i> uniforme	74
Tabela 8 – Dados gerais sobre o conjunto de dados do curso EVC	78
Tabela 9 – Trecho do <i>log</i> de eventos do EVC	79
Tabela 10 – Dados gerais do novo conjunto de dados do curso EVC	80
Tabela 11 – Trecho do <i>log</i> de eventos pré-processado do EVC	81
Tabela 12 – Amostra do log de eventos do EVC após a reorganização	81
Tabela 13 – Lista de configurações dos parâmetros	87
Tabela 14 – Resultados das execuções para a amostra com 10 traços de eventos . .	88
Tabela 15 – Resultados das melhores configurações para o log com 20 traços. . . .	89
Tabela 16 – Resultados das melhores configurações para o log com 35 traços. . . .	89

Lista de abreviaturas e siglas

BPM	Gestão de Processos de Negócio - <i>Business Process Management</i>
BPMN	Modelagem e Notação de Processos de Negócio - <i>Business Process Model and Notation</i>
KDD	Descoberta de Conhecimento em Bancos de Dados - <i>Knowledge Discovery from Data</i>
GSP	<i>Generalized Sequential Pattern</i>
CLS	<i>Concept Learning System</i>
RBF	Função de Base Radial - <i>Radial Basis Function</i>
XML	Linguagem de Marcação Extensível- <i>Extensible Markup Language</i>
ETM	<i>Evolutionary Tree Miner</i>
WF-net	Rede de Workflow - <i>Workflow net</i>
GA	Algoritmo Genético - <i>Genetic Algorithm</i>
GP	Programação Genética - <i>Genetic Programming</i>
ES	Estratégias Evolutivas - <i>Evolutionary Strategies</i>
EP	Programação Evolutiva - <i>Evolutionary Programming</i>
TP	Precisão Transicional - <i>Transitional Precision</i>
LCP	Compleitude e Precisão Lógica - <i>Logical Completeness and Precision</i>
B/A	Antes/Depois - <i>Before/After</i>

Lista de símbolos

$\bullet t$	Conjunto de locais de entrada de t
$t\bullet$	Conjunto de locais de saída de t
α	Letra grega alfa
β	Letra grega beta
$\cup$	União
$\cap$	Interseção
$\sum$	Somatório
$\forall$	Para todo
$ $	Tal que
$\leftarrow$	Atribuição
$\emptyset$	Conjunto vazio
$\exists$	Existe
$\in$	Pertence
$\subseteq$	Está contido
$\supset$	Contém
$\wedge$	Operador lógico de conjunção
AND	Operador lógico de conjunção
xOR	Operador lógico de disjunção exclusiva

Sumário

1	Introdução	17
1.1	<i>Justificativa e motivação</i>	19
1.2	<i>Hipóteses</i>	22
1.3	<i>Objetivos</i>	22
1.4	<i>Método de pesquisa</i>	23
1.5	<i>Organização deste documento</i>	24
2	Fundamentação teórica	26
2.1	<i>Processos de negócio</i>	26
2.1.1	Redes de Petri	27
2.1.2	Redes de <i>workflow</i>	28
2.2	<i>Mineração de dados</i>	29
2.2.1	Tarefas da mineração de dados	30
2.2.2	Inteligência computacional e aprendizado de máquina	33
2.3	<i>Técnicas evolutivas</i>	38
2.3.1	Estrutura do algoritmo evolutivo	39
2.3.2	Algoritmos genéticos	40
2.3.3	Programação genética	41
2.3.4	Estratégias evolutivas	42
2.3.5	Programação evolutiva	42
2.4	<i>Mineração de processos</i>	43
2.4.1	Descoberta de processos	44
2.4.2	Conformidade de processos	45
2.4.3	Melhoria de processos	45
2.5	<i>Problema da descoberta de processos</i>	46
2.5.1	Descrição geral do problema de descoberta de processos	46
2.5.2	Desafios da descoberta de processos	49
2.6	<i>Definição do problema da descoberta de processos não estruturados . .</i>	51
3	Revisão da literatura	54
3.1	<i>Planejamento da revisão de literatura</i>	54

3.1.1	Necessidade de uma revisão de literatura	54
3.1.2	Questões de pesquisa	55
3.2	<i>Protocolo</i>	56
3.3	<i>Resultados</i>	56
3.3.1	Tipos de mineração de processos	58
3.3.2	Utilização dos <i>logs</i> e criação da população inicial	58
3.3.3	Representação estrutural e cromossômica dos modelos de processo .	59
3.3.4	Técnicas evolutivas empregadas	60
3.3.5	Técnicas auxiliares	60
3.3.6	Operadores de correção	61
3.3.7	Operadores genéticos	61
3.3.8	Crítérios de avaliação (<i>fitness</i>)	63
3.3.9	Crítérios de parada	64
3.3.10	Parametrização	64
3.4	<i>Considerações finais</i>	64
4	Estrutura do GA	67
4.1	<i>Representação cromossômica</i>	67
4.2	<i>Função fitness</i>	68
4.3	<i>Operadores genéticos</i>	72
4.3.1	Crossover	73
4.3.2	Correção do cromossomo	73
4.3.3	Mutação	75
4.3.4	Fluxo evolutivo	75
5	Experimentos e Resultados	77
5.1	<i>Conjunto de dados</i>	77
5.1.1	<i>Logs</i> de um ambiente virtual de aprendizado	77
5.1.2	Pré-processamento	80
5.2	<i>Configuração dos experimentos e resultados</i>	82
5.2.1	Ferramenta <i>ETM-ProM</i>	83
5.2.2	Abordagem proposta de GA	86
6	Conclusão	94

Referências ¹	97
------------------------------------	----

APÊNDICES 102

Apêndice A – Protocolo da revisão de literatura	103
<i>A.1 Fontes de dados e estratégia de busca</i>	<i>103</i>
A.1.1 Strings de busca - Scopus:	104
A.1.2 Strings de busca - WoS:	104
<i>A.2 Estratégia de seleção dos estudos primários</i>	<i>104</i>
A.2.1 Critérios de inclusão	104
A.2.2 Critérios de exclusão	105
<i>A.3 Condução da revisão sistemática de literatura</i>	<i>105</i>
A.3.1 Identificação e seleção dos estudos primários	106
A.3.2 Extração e síntese dos dados	106

¹ De acordo com a Associação Brasileira de Normas Técnicas. NBR 6023.

1 Introdução

Processos de negócio podem ser entendidos como um conjunto de atividades cuja execução coordenada tem por fim atingir um determinado objetivo organizacional. A gestão de processos de negócio (BPM)¹ é baseada no fato de que cada produto gerado por uma determinada organização é o fruto de um conjunto de atividades realizadas pela mesma, sendo os processos de negócio uma ferramenta fundamental para a organização dessas atividades de forma que o relacionamento entre elas se torne compreensível. BPM envolve métodos, técnicas e ferramentas que tem por objetivo apoiar o projeto, execução, gestão e análise de processos de negócio. O ciclo de vida de BPM contém basicamente cinco fases, sendo: modelagem de processos; implementação; execução e supervisão; monitoramento e auditoria; e avaliação e melhoria de modelos de processo (WESKE, 2012).

A mineração de dados é um processo computacional de extração de conhecimento a partir de um conjunto volumoso de dados, sendo uma das etapas de um processo maior conhecido como KDD². Esse processo é composto pelas etapas de limpeza, integração, seleção, transformação dos dados, mineração, avaliação de padrões e apresentação do conhecimento (HAN; KAMBER; PEI, 2012). Dentro da etapa de mineração de dados, existem tarefas nas quais podem ser empregadas várias técnicas computacionais. Dentre as técnicas de inteligência computacional aplicáveis, destaca-se o uso de técnicas pertencentes ao campo da computação evolutiva (MAITA, 2015), as quais são inspiradas no mecanismo de evolução observado nos seres vivos (SIVANANDAM; DEEPA, 2008).

A computação evolutiva é composta de técnicas heurísticas que abstraem esses princípios evolutivos, incorporando-os a algoritmos que buscam soluções ótimas para um problema. A principal característica desses algoritmos é o processo de busca baseado no conceito de populações. Por meio da adaptação de uma sequência de populações compostas de indivíduos, os quais são estruturados de acordo com uma representação cromossômica (estruturas de dados como vetores, matrizes, árvores de processo, etc.), os algoritmos evolutivos realizam um processo de busca direcionada pela capacidade de sobrevivência (chamada de *fitness*) dos indivíduos, resultando numa busca mais eficiente e que abrange um espaço bem maior que outras técnicas tradicionais (SIVANANDAM; DEEPA, 2008).

¹ Do inglês: *Business Process Management*

² Do inglês: *Knowledge Discovery from Data*

A computação evolutiva é compreendida em quatro tipos principais de algoritmos: algoritmos genéticos, programação genética, estratégias evolucionárias e programação evolucionária. Dentre as quatro técnicas, os algoritmos genéticos são os mais amplamente desenvolvidos na aplicação dos conceitos de evolução. Algoritmos genéticos reproduzem mecanismos naturais como a seleção, *crossover* e mutação, criando sucessivamente populações de indivíduos com o uso desses operadores. O objetivo da técnica é conseguir populações cada vez melhores, sendo essa evolução guiada por uma métrica de qualidade (*fitness*) associada a cada indivíduo da população, a qual representa a capacidade do indivíduo de solucionar o problema sendo otimizado (SIVANANDAM; DEEPA, 2008). Uma característica associada a essas técnicas é a necessidade de escolher os parâmetros de inicialização do algoritmo, tais como o tamanho da população, a quantidade de gerações, as taxas e tipos de mutação e cruzamento, o método de seleção dos pais para a reprodução, etc. Essas escolhas afetam diretamente o desempenho do algoritmo na solução do problema (DE JONG, 2006).

A mineração de dados, associada à BPM, deu origem a um novo campo de pesquisa conhecido como mineração de processos. A mineração de processos segue a premissa de que existe um processo desconhecido em um determinado contexto, e que ao analisar os traços do comportamento real do processo (os eventos registrados nos logs durante a execução) com o auxílio de técnicas de mineração de dados, é possível obter uma descrição agregada desses comportamentos na forma de um modelo de processo (AALST; GUNTHER, 2007). O objetivo da mineração de processos é extrair informações dos logs de eventos de um processo de negócio, obtendo conhecimento a respeito dos eventos e dados relativos às tarefas executadas em suas diferentes fases, de forma a melhorá-lo, descobrir associações entre variáveis, comportamentos ou mesmo anomalias no processo (AALST, 2011).

Na mineração de processos, lida-se com dois tipos principais de processos de negócio: processos estruturados (denominados de processos "*lasanha*") e não estruturados (denominados de processos "*espaguete*"). Os processos estruturados possuem poucos comportamentos infrequentes e uma forte dependência causal entre as atividades do processo. As técnicas de mineração de processos atuais funcionam bem nesse ambiente estruturado, resultando na mineração de modelos de processo que se assemelham a uma "*lasanha*" (com camadas bem definidas). No entanto, os processos mais interessantes de minerar não possuem essa qualidade nos logs, sendo mais difíceis de minerar pelo excesso de comportamento registrado nos logs. Embora complexos, esses processos não

estruturados possuem um potencial grande de melhoria (AALST; GUNTHER, 2007). A ocorrência de processos não estruturados é comum nas organizações, onde o fluxo de execução do processo depende principalmente de decisões eventuais tomadas pelos participantes do processo, tornando cada execução deste essencialmente diferente das outras (DIAMANTINI; GENGA; POTENA, 2016).

Nesses casos, as técnicas atuais de mineração de processos se revelam ineficientes, minerando modelos de processo excessivamente complexos que provém pouco suporte à análise (DIAMANTINI; GENGA; POTENA, 2016). Seguindo a analogia dos modelos de processo “lasanha”, na mineração de logs de processos não estruturados o resultado costuma ser um modelo de processo “espaguete” (com excesso de comportamento representado). Tais modelos não provém uma abstração significativa do conjunto de logs (AALST; GUNTHER, 2007).

Este trabalho de pesquisa de mestrado visa aplicar técnicas computacionais evolutivas na mineração de modelos de processo, usando algoritmos genéticos para descobrir automaticamente modelos de processos não estruturados, visando suportar a gestão organizacional desses processos. O conjunto de dados a ser usado neste trabalho é referente a um ambiente virtual de aprendizado, o qual foi desenvolvido para apoiar o processo de ensino e aprendizagem do curso “Ética, Valores e Cidadania na Escola” (EVC). A plataforma do curso foi desenvolvida pela Universidade Virtual de São Paulo (UNIVESP) com o uso da ferramenta Moodle, e o processo de ensino e aprendizagem pode ser compreendido como não estruturado devido à natureza irrestrita da plataforma em que o curso é oferecido, além do mesmo não ter sido desenvolvido com base em modelagem prévia do processo de negócio. Visa-se com isso descobrir o modelo do processo de aprendizado realizado pelos alunos, que é ocultado pelo alto nível de ruído presente nos logs.

1.1 Justificativa e motivação

A mineração de processos de negócio é o elo entre as áreas de BPM e a mineração de dados. A mineração de processos tem como objetivo descobrir, monitorar e aprimorar os processos de negócio das organizações por meio da extração de conhecimento, tendo base nos *logs* de eventos dos processos os quais se encontram prontamente disponíveis na grande maioria dos sistemas de informação modernos. Com a crescente disponibilidade

de conjuntos de *logs* de eventos, viabilizou-se o estudo e desenvolvimento de inúmeras técnicas de descoberta, conformidade e melhoria de processos, afirmando que na última década a área tem visto um crescimento significativo da aplicação concreta dessas técnicas em ambientes corporativos (AALST, 2016).

Em Turner et al. (2012), é possível observar um estudo da aplicação de diferentes técnicas à mineração de processos. Nesse estudo afirma-se que a categoria de técnicas computacionais (tais como algoritmos evolutivos, redes neurais, técnicas *fuzzy*, etc.) têm sido o foco de pesquisa da área de mineração de processos nos anos recentes, sendo a mais usada na literatura para a solução dos problemas da mineração de processos.

No estudo realizado pelo nosso grupo de pesquisa em BPM na EACH-USP (MAITA, 2015), fora feito um mapeamento da área de mineração de processos tendo como objetivo realizar uma análise da produção científica na área. Foram levantados 705 artigos nos quais foram analisadas, dentre outras informações, os tipos de mineração de processos aplicados, tarefas de mineração de dados empregadas e as técnicas aplicadas nas abordagens. O mapeamento revelou que há uma preferência pelo estudo da descoberta de processos, o que pode ser explicado uma vez que a existência de um modelo do processo é um recurso fundamental para a realização dos outros tipos de mineração de processos.

Dentre as técnicas aplicadas, o mapeamento mostra que os estudos usam em sua maioria uma variedade de técnicas tradicionais baseadas em grafos, estatística, heurísticas, dentre outras. No entanto, em Turner et al. (2012) e no mapeamento feito pelo nosso grupo de pesquisa (MAITA, 2015) é possível observar uma tendência na aplicação de técnicas de inteligência computacional e aprendizado de máquina na mineração de processos, com destaque às técnicas de computação evolutiva e de lógica *fuzzy*.

Este trabalho é motivado pelo interesse do grupo de pesquisa na mineração de processos aplicada ao contexto do conjunto de *logs* do EVC. Devido ao caráter irrestrito da plataforma de ensino que suporta o curso, e pelo fato da mesma ter sido desenvolvida sem a referência de um modelo de processo, o seu processo de negócio pode ser caracterizado como não estruturado. Processos não estruturados possuem um alto nível de irregularidade na execução das tarefas, uma vez que cada instância do processo tem liberdade para executar um comportamento completamente heterogêneo em comparação às outras instâncias do mesmo processo, o que é refletido no conjunto de *logs* do processo. As técnicas de inteligência computacional são as que obtiveram melhores resultados na mineração de processos estruturados, sendo capazes de descobrir diversas estruturas complexas de fluxo

de controle presentes nos logs (TURNER et al., 2012). No entanto, quando aplicadas à mineração de processos não estruturados, essas técnicas resultam em modelos de processo altamente complexos, os quais dão pouco suporte à análise do processo (DIAMANTINI; GENGA; POTENA, 2016).

Segundo Aalst e Gunther (2007), a descoberta de processos de negócio não estruturados é um desafio da mineração de processos, tendo revelado consideráveis pontos falhos nas técnicas tradicionais desenvolvidas. As melhores abordagens atuais para a mineração de processos não estruturados viabilizam uma análise em apenas parte do processo, necessitando de grandes quantidades de pré-processamento nos logs, ou em alto nível de granularidade, resultando em um modelo que se assemelha a um mapa do processo, mas que não representa as estruturas de paralelismo entre atividades (DIAMANTINI; GENGA; POTENA, 2016). Embora seja possível observar informações como os agrupamentos e a relação de causalidade entre as atividades, tais mapas não possibilitam saber visualmente qual é o fluxo de controle do processo.

Dessa forma, a realização deste trabalho se justifica pelo desempenho insatisfatório das abordagens atuais de mineração de processos na descoberta de modelos de processo não estruturados. As técnicas atuais não lidam eficientemente com o alto nível de ruído nos logs de processos não estruturados como o EVC, e como resultado os modelos descobertos não expressam claramente as estruturas de paralelismo e de fluxo de controle inerentes nesse ambiente complexo, tendo utilidade limitada para a análise do processo.

A escolha da técnica de algoritmos genéticos para o problema de mineração de processos não estruturados é motivada pela relevância da técnica observada no mapeamento realizado pelo nosso grupo de pesquisa (MAITA, 2015) e pela característica desta técnica de lidar bem com a otimização de problemas difíceis, sendo capaz de encontrar soluções em espaços de busca complexos e de se adaptar a ambientes sob constante mudança (DE JONG, 2006). Tal característica é fundamental ao problema dos processos não estruturados, os quais possuem alto nível de ruído consequente do ambiente de execução não restritivo e altamente flexível destes (AALST; GUNTHER, 2007).

Outra motivação para a escolha dessa técnica é que na revisão de literatura realizada, foi possível observar que os operadores implementados nos estudos avaliados foram inspirados em conceitos básicos dos operadores evolutivos. A teoria da *inexistência do almoço gratuito* afirma que não existe um algoritmo genérico que tenha bom desempenho para todos os problemas de busca, sendo essencial inserir o máximo de conhecimento do

problema no algoritmo de busca (SIVANANDAM; DEEPA, 2008). Técnicas de propósito geral (como os algoritmos evolutivos) tendem a ser ineficientes caso não sejam projetadas especificamente para o problema em mãos, e uma forma de inserir conhecimento do problema nos algoritmos evolutivos é através da escolha da representação cromossômica e dos operadores genéticos usados (ASHLOCK, 2006). Assim, o emprego de operadores genéticos específicos (projetados considerando informações do problema) é um ponto a explorar que pode levar a melhoras significativas no resultado da mineração.

1.2 Hipóteses

Esta seção elenca as hipóteses levantadas para este trabalho, como resultado da revisão de literatura e pesquisa bibliográfica realizadas para este projeto.

- Hipótese 1: devido à alta presença de ruídos nos *logs*, técnicas heurísticas como os algoritmos genéticos obterão um resultado mais conciso que técnicas tradicionais na descoberta de processos.
- Hipótese 2: o uso de operadores genéticos específicos ao problema levarão à obtenção de melhores resultados na mineração de processos não estruturados.

1.3 Objetivos

O objetivo geral deste trabalho é desenvolver uma solução mais eficaz para a mineração de modelos de processo não estruturados por meio da aplicação de técnicas de computação evolutiva. Visa-se com essa solução descobrir modelos de processo mais adequados que os obtidos com as abordagens atuais, revelando estruturas de fluxo de controle e paralelismo nos processos de negócio não estruturados. Decorrente do objetivo acima delineado, os objetivos específicos são:

1. Aplicar algoritmos genéticos dentro do contexto de processos não estruturados, propondo o uso de operadores genéticos específicos (que levem em consideração as estruturas do processo, a ordenação das tarefas, a factibilidade do cromossomo, etc.) para o problema da descoberta de modelos de processo.

2. Apresentar um estudo da parametrização da solução proposta, de forma a expor o comportamento da técnica empregada e obter uma parametrização ótima da solução para o problema abordado.

1.4 Método de pesquisa

Este capítulo tem por objetivo descrever, a nível geral, o método de pesquisa usado neste trabalho. De forma ampla, o método de pesquisa usado enquadra-se como:

- **Natureza “aplicada”:** uma vez que se tem o desejo de conhecer o objeto de estudo com vistas a fazer algo mais eficiente ou eficazmente. O objeto de estudo são as técnicas de inteligência computacional e aprendizado de máquina, e o objetivo é a aplicação dessas técnicas para a obtenção de melhores resultados na mineração de processos não estruturados.
- **Gênero “prático”:** já que este trabalho é direcionado principalmente à prática do conhecimento científico, aplicando técnicas já existentes na literatura com fins de intervenção na realidade dos processos de negócio.
- **Objetivo “explicativo”:** este trabalho tem o caráter explicativo porque busca identificar fatores que determinam ou contribuem para a ocorrência de um fenômeno específico. Neste caso, o objetivo é descobrir um modelo de processo que explique o comportamento observado no conjunto de log do processo estudado, determinando também a relação entre as atividades do processo.
- **Abordagem mista “qualitativa” e “quantitativa”:** esta porque o trabalho leva em consideração o raciocínio dedutivo, regras de lógica e atributos mensuráveis da experiência humana (por exemplo, na aplicação de métricas de avaliação dos modelos de processo), e aquela porque se enfatiza também aspectos subjetivos dessa experiência, com o objetivo de compreender e interpretar elementos (como determinar se um modelo de processo descoberto é bom ou ruim com base no seu aspecto visual).

Como procedimento técnico, este trabalho tem base principalmente na experimentação, por meio da formulação do problema da mineração de processos não estruturados e das hipóteses, delimitando as variáveis precisas e controladas que possam atuar no fenômeno estudado. A fonte de dados para este trabalho é a pesquisa em laboratório, caracterizada pelo ambiente controlado que permite a experimentação na manipulação das

variáveis do problema, para que sejam formuladas as hipóteses. A técnica para a coleta de dados foi o registro dos eventos no processo de aprendizado dos alunos por meio de *logs* do sistema. A coleta de dados foi feita diretamente no ambiente virtual de aprendizado do EVC, desenvolvido sobre a ferramenta Moodle, que faz o registro automático dos eventos de interação dos usuários com os objetos do sistema. A análise quantitativa dos dados será feita nos elementos do trabalho que possam ser mensurados em números, usando estatística descritiva básica para descrever, analisar e classificar os dados. Já a análise qualitativa será realizada por meio de análise do conteúdo, sustentada com base nas teorias encontradas na literatura de BPM.

De forma geral, o desenvolvimento deste trabalho compreende três etapas importantes, sendo que em cada uma delas atividades para alcançar os objetivos específicos serão realizadas. Tais etapas são resumidamente descritas a seguir:

1. Delimitação da área de mineração de processos, em relação às técnicas e tarefas de mineração aplicadas em tal contexto, buscando evidências para justificar o menor desempenho das atuais técnicas de mineração de processos na descoberta de processos não estruturados.
2. Desenvolvimento de uma aplicação para a descoberta de processos não estruturados, utilizando a técnica de algoritmos genéticos. Serão realizados experimentos com a função *fitness* e com operadores genéticos especificamente projetados para o problema da descoberta de processos, além da análise da parametrização do algoritmo.

Anteriormente a este trabalho, foi realizada uma pesquisa bibliográfica com o fim de propiciar uma visão geral do estado da arte relacionado ao tema deste projeto. Nessa revisão de literatura, fez-se o levantamento de artigos que realizaram alguma das tarefas de mineração de processos com o uso das técnicas computacionais evolutivas. O protocolo da revisão de literatura, que seguiu as diretrizes e técnicas de revisão sistemática de literatura, se encontra no apêndice A deste documento.

1.5 Organização deste documento

O presente trabalho de pesquisa está dividido em 6 capítulos, considerando esta introdução. Os demais capítulos estão organizados da seguinte forma:

- O capítulo 2 contém a fundamentação teórica usada neste trabalho;

- O capítulo 3 apresenta a revisão de literatura realizada para este projeto;
- O capítulo 4 descreve a estrutura da abordagem baseada em algoritmos genéticos proposta neste trabalho.
- No capítulo 5 é apresentado o conjunto de dados e o processo de pré-processamento dos dados, bem como a configuração e análise dos resultados dos experimentos de validação da abordagem proposta.
- No capítulo 6 conclui este trabalho, apresentando as considerações finais e destacando as contribuições deste.
- Por fim, no apêndice é apresentado o protocolo da revisão de literatura, que seguiu os procedimentos de uma revisão sistemática.

2 Fundamentação teórica

Este capítulo apresenta conceitos básicos envolvidos na realização deste trabalho, incluindo: processos de negócio, mineração de dados, técnicas de inteligência computacional evolutiva e mineração de processos.

2.1 Processos de negócio

Processos de negócio (ou *workflow*) podem ser entendidos como um conjunto de atividades desempenhadas de forma coordenada dentro de um contexto técnico e organizacional, visando alcançar um objetivo organizacional por meio dessa operação conjunta. Esses processos são em geral realizados internamente na organização, mas também podem interagir com processos de organizações externas (WESKE, 2012). Processos de negócio podem ainda ser definidos como uma coleção de eventos, atividades e pontos de decisão inter-relacionados, os quais envolvem vários atores e objetos e que, de forma conjunta, levam a um resultado que apresenta valor para ao menos um dos clientes da organização (DUMAS et al., 2013).

A gestão de processos de negócio (BPM) é um conjunto de conceitos, métodos e técnicas que visam auxiliar no projeto, administração, configuração, execução e análise dos processos de negócio, tendo como base a representação formal desses processos de forma a discriminar explicitamente suas atividades e as restrições de execução existentes entre elas. BPM facilita a compreensão da ordem e das interações entre as atividades do processo, o que é fundamental visto que os produtos da organização dependem da correta execução dessas atividades (WESKE, 2012).

Na área de BPM, diferentes técnicas podem ser utilizadas para a modelagem visual dos processos de negócio, no entanto, a técnica de modelagem mais usada na literatura é a de redes de Petri. As redes de Petri foram uma das primeiras técnicas de modelagem a serem propostas para a modelagem de sistemas computacionais, sendo a técnica mais testada e validada na literatura (REISIG, 2013). Essa preferência ocorre devido a maioria das outras técnicas usarem noções que se originaram em redes de Petri, como a semântica de execução baseada em *tokens*, o que permite converter os modelos dessas outras técnicas em redes de Petri (AALST, 2015).

2.1.1 Redes de Petri

Redes de Petri são compostas de dois tipos de elementos: locais e transições. Os locais modelam componentes passivos do processo, podendo armazenar, acumular ou mostrar objetos. Locais possuem estados discretos, e são representados graficamente por círculos ou elipses. As transições modelam componentes ativos do processo, podendo produzir, consumir ou mudar artefatos no processo, representadas graficamente por quadriláteros (REISIG, 2013).

Locais e transições são logicamente conectados em uma rede de Petri por meio de arcos direcionados, sendo representados graficamente por setas. Esses arcos não modelam componentes do sistema em questão, mas sim as relações abstratas entre os locais e as transições. Uma restrição das redes de Petri é que arcos só podem conectar elementos distintos (ou seja, locais à transições e vice-versa) (REISIG, 2013). Na figura 1, pode-se observar um exemplo de rede de Petri com quatro locais (p_1 a p_4), quatro transições (t_1 a t_4) e oito arcos direcionados ligando esses elementos.

Figura 1 – Exemplo de modelagem usando redes de Petri

Fonte: Gabriel Lucas Cantanhede da Silva, 2018

Uma rede de Petri N pode ser denotada da seguinte forma:

$$N = (P, T, F)$$

Onde P , T e F representam, respectivamente, os conjuntos de todos os locais, transições e a relação de pares de arcos da rede. F pode ainda ser definido como a relação de fluxo de N (ou seja, o conjunto de caminhos da rede), denotado da seguinte forma (REISIG, 2013):

$$F \subseteq (P \times T) \cup (T \times P)$$

Um local p é classificado como entrada de uma transição t se e somente se existir um arco direcionado de p a t . De forma análoga, um local p é tido como saída de uma transição t se e somente se existir um arco direcionado de t a p . A notação $\bullet t$ é usada para representar o conjunto de entradas de uma transição t , assim como $t\bullet$ representa o conjunto de saídas partindo de uma transição t , sendo a mesma notação aplicada aos locais da rede. O *token* (representado graficamente por um ponto preto) é um elemento que transita pela rede, simulando a sua execução. O estado da rede (também chamado marcação da rede) é a distribuição de *tokens* pelos locais da rede, que pode ser representado da seguinte forma (AALST, 1998):

$$M_n = 0p_1 + 1p_2 + 1p_3 + 0p_4$$

M_n denota o estado da rede de Petri na figura 1. O estado M_n denota que a rede em questão possui nenhum *token* no local p_1 , um *token* no local p_2 , um no local p_3 e nenhum no local p_4 . A semântica de execução da rede ocorre por meio da atuação dos componentes ativos da rede (transições), os quais modificam o estado da rede de acordo com as seguintes regras de execução (AALST, 1998):

1. Uma transição t é tida como habilitada se e somente se cada local de entrada p pertencente ao conjunto $\bullet t$ contiver ao menos um *token*.
2. Uma transição habilitada pode ser executada. Se isso ocorrer, consome-se um *token* de cada local pertencente ao conjunto $\bullet t$ e é produzido um *token* em cada local pertencente ao conjunto $t\bullet$.

Aalst (1998) ressalta que, apesar de redes de Petri permitirem a modelagem de processos de negócio, tal notação possibilita a definição de modelos de processo com potenciais inconsistências estruturais de execução (como múltiplos locais de início e fim do processo). Essa necessidade levou à definição de um tipo específico de rede de Petri mais adequado à modelagem de processos, denominado redes de workflow.

2.1.2 Redes de *workflow*

Uma rede de *workflow* (ou *WF-net*) pode ser entendida como um caso específico de rede de Petri, a qual modela um processo de negócio. A principal diferença entre uma *WF-net* e uma rede de Petri é que esta permite começar a execução do processo modelado

por mais de um local, enquanto que naquela há um local explícito para o início e fim da execução do processo. Em uma *WF-net*, o conjunto de locais P correspondem a condições de execução das tarefas, enquanto que o conjunto de transições T representa todas as tarefas do processo. Uma *WF-net* deve satisfazer os seguintes requisitos (AALST, 1998):

1. Deve possuir apenas um local de início i e um local de fim o , sendo os conjuntos $\bullet i$ e $o \bullet$ vazios. Um *token* em i corresponde a um caso que deverá ser tratado, enquanto que um *token* em o simboliza um caso que já fora tratado.
2. Não deve possuir transições/locais isolados, ou seja, cada elemento na rede deve contribuir para o processamento dos casos do processo. Dessa forma, cada transição t ou local p deve pertencer a um caminho que vai do local de início ao local de fim do processo.

Figura 2 – Exemplo de modelo de processo, representado em WF-net

Fonte: Aalst (2015)

2.2 Mineração de dados

A mineração de dados é formalmente um dos passos de um processo maior conhecido como KDD. Porém, tanto na indústria quanto no meio social acadêmico o termo mineração de dados é usado para descrever o processo por completo. Dessa forma, a mineração de dados é definida como sendo o processo de descoberta de padrões e conhecimentos de interesse não evidentes, obtidos a partir da exploração e análise (automática ou semi-automática) de um grande volume de dados (HAN; KAMBER; PEI, 2012).

Há aplicações de mineração de dados em diversas áreas. Exemplos de aplicações são: (i) agrupamento de clientes de acordo com perfis para gestão do relacionamento com clientes; (ii) predição de séries temporais hidrológicas a partir de redes neurais

(TAMPELINI et al., 2009); e *(iii)* classificação (início de um módulo de reconhecimento de padrões) na Língua Brasileira de Sinais (LIBRAS) (MADEO et al., 2010). Há também diversos exemplos de aplicação de mineração de dados em áreas como marketing, vendas, manufatura, finanças, saúde e energia.

Um problema associado à mineração de dados (e à KDD de uma forma geral) é que além da possibilidade de garimpar relacionamentos inúteis, o número de correlações possíveis de serem obtidas geralmente é muito grande, o que impede a análise exaustiva de cada uma (THOMÉ, 2002). Apesar de existirem ferramentas para auxiliar na execução das técnicas de mineração, as saídas dessas ferramentas ainda demandam análise humana. Não obstante, a mineração de dados contribui de forma significativa no processo de descoberta de conhecimento, permitindo aos especialistas focarem seus esforços nas partes mais relevantes dos dados (CAMILO C. O.; SILVA, 2009).

Mineração de dados é comumente classificada levando em consideração sua capacidade em realizar determinadas tarefas (GRZYMALA-BUSSE, 2008). Para cada tarefa de mineração de dados, diferentes técnicas podem ser aplicadas (HAN J.; KAMBER, 2006), incluindo aquelas das áreas de inteligência computacional e de aprendizado de máquina (WANG; FU, 2006). A seguir são apresentadas as principais e mais comuns tarefas de mineração de dados.

2.2.1 Tarefas da mineração de dados

As tarefas de mineração de dados podem ser classificadas em dois grupos, de acordo com o seu objetivo primário (KANTARDZIC, 2011): as técnicas de mineração *preditivas* focam no uso de variáveis ou campos do conjunto de dados para realizar a predição de valores desconhecidos ou futuros de outras variáveis de interesse, produzindo o modelo do sistema descrito pelo conjunto; as técnicas de mineração *descritivas* têm por objetivo encontrar padrões que descrevam os dados e que possam ser interpretados por pessoas, produzindo informações novas e não triviais baseadas no conjunto de dados em questão. As principais tarefas da mineração de dados, segundo as definições de Han J.; Kamber (2006), são: *(i)* predição categórica (ou classificação), predição numérica (tipo “regressão”), e predição numérica (tipo “identificação de tendências”); *(ii)* análise de agrupamentos;

(*iii*) descoberta de padrões frequentes, regras de associação e correlação; (*iv*) análise de outliers e ruído; (*v*) análise de evolução; e (*vi*) análise de dados estruturados.

- **Predição:** trata da construção de um modelo preditivo baseado em exemplos contidos em uma base histórica. A construção do modelo, segundo essa tarefa, pressupõe o conhecimento prévio das possíveis classes (ou rótulos) e a correta classificação dos exemplos usados na modelagem. Existem várias aplicações para esse tipo de abordagem; em análise de risco, o objetivo pode ser a classificação dos clientes em potencial, rotulando cada um como “excelente”, “bom” ou “mau” pagador. Segundo Han e Kamber (2006), há três tipos de predição:
 - **Predição categórica (ou classificação):** consiste em buscar uma função que classifique uma determinada ocorrência em uma categoria dentre um conjunto finito e predefinido de classes. As principais técnicas que podem ser usadas para esse tipo de predição são árvores de decisão e redes neurais.
 - **Predição numérica (do tipo “regressão”):** consiste em buscar uma função que represente (de forma aproximada) o comportamento apresentado pelo fenômeno em estudo. Esse modelo estatístico relaciona uma ou mais variáveis independentes (ou variáveis de predição) e variáveis dependentes (ou variáveis de resposta), as quais têm valores contínuos. A variável de resposta representa o que se quer prever. A mais conhecida é a regressão linear, a exemplo de uma reta que minimize o erro médio entre todos os valores considerados. No entanto, a regressão não linear é mais adequada a problemas reais, e uma das principais técnicas usadas para esse tipo de predição são as redes neurais.
 - **Predição numérica (do tipo “identificação de tendências”):** envolve um componente temporal, isto é, representa problemas em que o interesse é prever o valor futuro de uma determinada variável (mono-variável) ou valores anteriores da variável de interesse e de outras variáveis (multi-variável). As principais técnicas que podem ser usadas nesse tipo de predição são métodos estatísticos, máquinas de vetores de suporte e redes neurais.
- **Análise de agrupamentos:** consiste em buscar similaridades entre dados para definir um conjunto finito de grupos, organizando esses dados em agrupamentos de forma a maximizar a similaridade dentro do grupo (intra-agrupamento) e minimizar a similaridade entre os grupos (interagrupamento). Consiste também em determinar

um conjunto não conhecido de classes para esses dados de forma a descrevê-los. Um exemplo de aplicação para essa tarefa é a descoberta de grupos homogêneos de clientes, que pode ajudar na definição de estratégias de marketing a serem aplicadas nesses grupos. As principais técnicas que usadas para essa tarefa são os métodos estatísticos e as redes neurais.

- **Descoberta de padrões frequentes, regras de associação e regras de correlação:** consiste em identificar ocorrências frequentes que possam estar direta ou indiretamente associados (ou correlacionados) na ocorrência de eventos ou transações. Essa estratégia é geralmente usada em aplicações em que se busca identificar itens que possam ser colocados juntos em um mesmo pacote de negociação. Por exemplo, a constatação de que vendas de cerveja e de fraldas descartáveis apresentam um comportamento ascendente às sextas-feiras pode levar à construção de uma hipótese que associe ambos os itens a um tipo especial de cliente. Regras de associação também podem ser usadas para avaliar a existência de algum tipo de relação temporal entre os itens constantes em uma base de dados. Podem ser usadas para essa tarefa, por exemplo, métodos estatísticos.
- **Análise de outliers e de ruído:** uma base de dados pode conter objetos que não obedecem ao comportamento geral do modelo dos dados, os quais são chamados outliers. Em muitos casos, a mineração de dados descarta os outliers como ruído ou exceções. No entanto, para aplicações como detecção de fraude, esses eventos raros podem ser mais importantes do que as ocorrências regulares. Os outliers podem ser detectados usando, por exemplo: (i) métodos estatísticos que assumem um modelo de distribuição de probabilidades para os dados; (ii) modelos de distância em que os objetos que são substancialmente distantes de qualquer outro grupo são considerados outliers; ou métodos baseados em desvio que examina deferências nas principais características dos objetos no grupo.
- **Análise de evolução:** descreve modelos regulares e tendências de objetos, os quais mudam de comportamento no tempo. Apesar desse tipo de análise poder ser incluído em tarefas como “Descoberta de padrões frequentes, regras de associação e regras de correlação”, “Predição (classificação, regressão ou identificação de tendências)” ou “Análise de agrupamentos” de dados quando relacionados ao tempo, existem características próprias que os diferenciam, incluindo: análise de dados de séries temporais; união de padrões em sequências ou periodicidades; e análise de dados

baseada em similaridade. As principais técnicas que podem ser usadas para essa tarefa são: métodos para lidar com sliding windows model, mineração de padrões frequentes em fluxos de dados, mineração de padrões sequenciais em bases de dados transacionais, e algoritmo GSP¹ (baseado em geração-e-teste para um novo candidato).

- **Análise de dados estruturados:** mineração de dados foi inicialmente concebida para ser usada com base em repositórios estruturados de dados, tais como arquivos, banco de dados, e data warehouse. Porém, atualmente, os dados são representados por diversos formatos não estruturados, tais como: espacial e temporal, multimídia, web, entre outros. Assim, existe uma necessidade crescente de mineração de dados nesses tipos de dados não estruturados. As principais técnicas aplicáveis a essa tarefa são: mineração de sub-grafos frequentes e mineração baseada em restrições. Alguns exemplos específicos são apresentados a seguir:

- **Grafos:** usados na modelagem e na mineração de estruturas complexas tais como: circuitos, imagens, proteínas, redes biológicas, redes sociais, processos, documentos XML², etc. Um tipo de subestrutura frequente em grafos são os padrões básicos, que podem ser descobertos em uma coleção de grafos, como cliques ou componentes conexas. Tais padrões são úteis para caracterizar conjuntos de grafos, discriminar diferentes grupos de grafos, classificar e agrupar grafos, e buscar similaridades em bases de grafos.
- **Relacionamentos:** baseados na análise de redes sociais, em que é representado o relacionamento entre as entidades envolvidas (que é similar a uma estrutura de grafos). As redes sociais são conjuntos de dados heterogêneos e relacionais representados por grafos. Elas não estão necessariamente em um contexto social, podendo estar relacionadas especificamente ao contexto de tecnologia, negócios, economia, entre outros.

2.2.2 Inteligência computacional e aprendizado de máquina

Aprendizado de máquina é uma área da computação que une elementos das áreas de inteligência artificial e estatística. Os algoritmos de aprendizado de máquina tem por objetivo passar por um processo de aprendizado, o qual ocorre por meio da busca

¹ Do inglês: *Generalized Sequential Pattern*

² Do inglês: *Extensible Markup Language*

por um espaço n -dimensional relativo a um conjunto de dados, visando obter uma generalização adequada ao problema. Uma das principais tarefas do aprendizado de máquina é o aprendizado indutivo, que é o processo de estimar uma dependência ou estrutura desconhecida entre a entrada e saída de um determinado sistema, usando uma quantidade limitada de amostras da entrada/saída do sistema (KANTARDZIC, 2011).

O processo de aprendizado indutivo envolve três componentes (KANTARDZIC, 2011): um componente responsável por gerar um vetor X de entradas aleatórias; um sistema que retorne um vetor de saídas Y relativo ao vetor de entrada X ; uma máquina de aprendizagem que estime o mapeamento desconhecido entre um valor de entrada X e de saída Y , com base nas amostras de entrada e saída observadas do sistema em questão.

No aprendizado indutivo, existem duas abordagens denominadas aprendizado supervisionado e aprendizado não supervisionado (KANTARDZIC, 2011): No aprendizado supervisionado, estima-se a dependência de dados de entrada e saída conhecidos. Nessa abordagem, um professor cumpre a função de avaliar o modelo em questão. Esse tipo de aprendizado é chamado de supervisionado porque os dados corretos de saída para uma entrada específica são conhecidos previamente. Dessa forma, o professor fornece conhecimento a respeito do contexto do aprendizado, e os parâmetros do sistema de aprendizado são ajustados em relação à influência do conjunto de dados de treinamento e do erro observado. O erro é definido como sendo a diferença entre a resposta desejada e aquela de fato provida pelo sistema, onde esse deve ajustar seus parâmetros de forma que o erro observado seja progressivamente minimizado. Já no aprendizado não supervisionado, apenas os dados de entrada são fornecidos ao sistema de aprendizado, sem qualquer noção de qual seria a saída ideal para esses dados. Nesse tipo de aprendizado, é necessário que o próprio “aluno” desenvolva uma forma de avaliar o seu modelo, sendo o objetivo descobrir a estrutura natural presente nos dados de entrada. O foco desse processo é a avaliação independente da qualidade da representação que é aprendida pelo sistema, sendo os parâmetros do mesmo otimizados de acordo com essa métrica.

Outro conceito importante na área de mineração de dados é o de inteligência computacional. Eberhart e Shi (2007) definem inteligência computacional como uma metodologia que envolve o uso de computadores para criar um sistema com a capacidade de aprender a lidar com situações inéditas, de forma que este possa ser considerado possuidor de um ou mais atributos do raciocínio (tal como generalização, descoberta, associação e abstração). Os sistemas de computação inteligente costumam incorporar

diferentes paradigmas de técnicas, tais como redes neurais artificiais, sistemas de lógica *fuzzy*, computação evolutiva, etc., sendo projetados para imitar determinados aspectos da inteligência biológica.

A área de inteligência computacional pode ser compreendida sob a perspectiva da adaptabilidade, abrangendo conceitos, paradigmas, algoritmos e implementações relacionadas à adaptação, que permitem a tomada de ações apropriadas (ou seja, comportamento inteligente) por parte dos sistemas em meio a um ambiente complexo e sob constantes mudanças (EBERHART; SHI, 2007).

Uma vez que a tarefa de mineração tenha sido definida, podem ser aplicadas diversas técnicas para a tarefa de mineração. Kantardzic (2011) lista os principais grupos de técnicas como sendo:

- **Métodos estatísticos:** a análise estatística de dados é uma das técnicas mais desenvolvidas dentro da área de mineração de dados, envolvendo desde a organização e descrição das características gerais de um conjunto de dados até a inferência de conclusões a partir desses. Os métodos estatísticos envolvem a análise de dados sob uma perspectiva uni ou multidimensional, envolvendo técnicas como regressão e análise discriminante. Exemplos de métodos estatísticos são a inferência bayesiana, regressão logística, análise de variância, análise discriminante, etc.
- **Análise de agrupamentos:** são um conjunto de técnicas que usam métricas de associação para a classificação automática de amostras de dados em um determinado número de grupos, de forma que os dados agrupados em um mesmo grupo sejam altamente semelhantes e que os pertencentes a grupos diferentes não apresentem similaridade. A entrada para um sistema de análise de agrupamentos é o conjunto de amostras e a métrica de similaridade entre os dados, sendo a saída um conjunto de agrupamentos dos dados formando uma partição ou estrutura de partições do conjunto de dados de entrada. A saída de uma análise de agrupamentos pode ser a descrição generalizada de cada agrupamento, sendo esse resultado fundamental para uma análise mais aprofundada das características do conjunto de dados em questão. As técnicas mais comuns de análise de agrupamentos são as de algoritmos aglomerativos, agrupamento por partições, agrupamento incremental, etc.
- **Árvores e regras de decisão:** as árvores e regras de decisão são métodos altamente relacionados com o problema de classificação na mineração de dados. Enquanto

outras técnicas usam operadores matemáticos, os modelos gerados por essas técnicas são baseados no uso de expressões lógicas, aplicando operadores comparativos e booleanos aos elementos do conjunto de dados. Isso resulta em uma classificação mais precisa dos dados, dando resultados que proveem um poder explanatório mais significativo que outras técnicas. As árvores de decisão são métodos supervisionados de aprendizado que constroem modelos hierárquicos com base nas amostras de entrada e saída do conjunto de dados, onde um elemento local é identificado com base numa sequência de segmentações recursivas, as quais ocorrem por meio da aplicação das decisões lógicas contidas nos nós. Essas árvores e regras são um método não paramétrico eficiente para realizar tarefas de classificação e regressão. Uma árvore de decisão pode também ser expressa como um conjunto de regras de decisão do tipo “SE-ENTÃO”, onde a parte contida no “SE” denota os testes lógicos a serem executados e a parte relativa ao “ENTÃO” é a consequência da avaliação lógica. Exemplos de técnicas incluem os algoritmos ID3 e C4.5, o método CLS³, além dos algoritmos de poda.

- **Regras de associação:** ao contrário de técnicas interessadas em obter um modelo global, ou seja, aplicável para todo o conjunto de entradas, as regras de associação estabelecem modelos locais que são afetados apenas por elementos semelhantes no conjunto de treinamento. Essas técnicas resultam na descoberta de inúmeros padrões locais contidos de forma não explícita no conjunto de dados, sendo que isso é considerado tanto uma vantagem quanto uma desvantagem da técnica devido ao esforço necessário para analisar a relevância de todas as regras resultantes. A mineração de regras de associação é fortemente associada com a mineração de conjuntos de itens frequentes. As técnicas mais comuns para a obtenção de regras de associação são o algoritmo *apriori* e a análise de cesta de compras.
- **Redes neurais artificiais:** redes neurais artificiais são um modelo de processador paralelo distribuído, composto por pequenas unidades de processamento. Redes neurais têm a capacidade de adquirir conhecimento experimental, que é expresso por meio de conexões estabelecidas entre as unidades que a compõem. As redes neurais são projetadas para imitar, de forma abstrata, o funcionamento complexo observado no cérebro humano, o qual é composto por neurônios interligados por sinapses. Dessa forma, uma rede neural possui uma estrutura de grafo, contendo uma quantidade de

³ Do inglês: *Concept Learning System*

nós (neurônios) conectados por meio de arestas direcionadas (sinapses). Cada nó representa uma unidade de processamento, e as arestas ligando os nós representam o relacionamento causal entre esses. Todos os neurônios são adaptáveis, de forma que a saída obtida desses depende dos parâmetros pertinentes aos mesmos. Os tipos de redes neurais mais usados são *multilayer perceptron* com retropropagação, redes de função de base radial (RBF)⁴ e redes de Kohonen.

- **Sistemas de inferência *fuzzy*:** os sistemas *fuzzy* são baseados na teoria dos conjuntos *fuzzy*, os quais modelam a incerteza comumente observada nos fenômenos do mundo real. O sistema de inferência *fuzzy* é baseado em dois elementos principais: as regras *fuzzy* e o raciocínio *fuzzy*. As regras *fuzzy* possuem uma estrutura do tipo SE(“x” é “A”) – ENTÃO(“y” é “B”), onde “A” e “B” são os valores linguísticos definidos pelos conjuntos *fuzzy* no universo de discurso “x” e “y”. No entanto, em casos onde x é equivalente a A’, se A’ é próximo de A então pode-se concluir que “y” é também próximo de “B”, sendo esse processo de inferência denominado raciocínio aproximado ou *fuzzy*. A modelagem e o processo de decisão *fuzzy* são dois passos frequentemente implementados em processos de mineração de dados.
- **Algoritmos genéticos:** os algoritmos genéticos (GA)⁵ são uma técnica bio-inspirada proposta por John Holland, em que se busca imitar computacionalmente os processos de reprodução e seleção natural observados na natureza. No GA, cada parâmetro a ser otimizado (o qual é pertencente a um espaço de soluções) é codificado em uma string binária denominada cromossomo. Cada cromossomo na população representa uma potencial solução para o problema de busca em questão, sendo que cada posição na string representa uma característica ou parâmetro, se forma análoga aos genes. Essas estruturas cromossômicas então passam por diversos operadores genéticos, com o objetivo de explorar o espaço de busca visando gerar soluções melhores para o problema. A qualidade dos indivíduos gerados pela técnica é mensurada pela função *fitness*, que visa fornecer um critério de comparação entre soluções boas e ruins. Em vez de realizar a busca em apenas um ponto do espaço de busca, GA mantém um conjunto de possíveis soluções de forma simultânea, denominadas população, a qual é evoluída de forma iterativa até que se alcance um valor de *fitness* ótimo. Durante o processo evolutivo, cada geração é obtida a partir do uso de operadores genéticos

⁴ Do inglês: *Radial Basis Function*

⁵ Do inglês: *Genetic Algorithm*

como o *crossover* e a mutação na população anterior onde, como na natureza, os melhores indivíduos tem a maior chance de sobreviver e se reproduzir, passando adiante seus genes bons para as gerações seguintes. Essa técnica é usada para problemas de otimização difíceis, como quando a função objetiva a ser otimizada apresenta perturbações que tornam a busca pelo espaço de soluções suscetível a inúmeros ótimos locais.

Várias técnicas podem ser usadas para realizar as tarefas presentes na mineração de dados, no entanto, as técnicas relativas ao campo da inteligência computacional e aprendizado de máquina se mostram particularmente eficientes na mineração de dados. Isso ocorre devido a sua capacidade de lidar com contextos onde os dados estão incompletos ou são imprecisos, o que é frequentemente observável nos casos reais, tornando as técnicas mais precisas e restritivas ineficientes (ABONYI; FEIL; ABRAHAM, 2005).

Dessa forma, as técnicas de inteligência computacional e aprendizado de máquina funcionam com base em raciocínios aproximados, indutivos ou evolutivos, o que torna estas altamente compatíveis com problemas de mineração cujo contexto envolve informações empíricas e comportamentais (ABONYI; FEIL; ABRAHAM, 2005).

2.3 *Técnicas evolutivas*

As técnicas da inteligência computacional de raciocínio evolutivo são caracterizadas por serem inspiradas em mecanismos de evolução, os quais foram conceituados por meio da observação do comportamento dos seres vivos e da forma como eles transmitem sua herança genética, levando essa abordagem o nome de computação evolutiva. As técnicas de computação evolutiva abstraem esses princípios evolutivos, incorporando-os a algoritmos que são usados em problemas de busca por soluções ótimas a um problema. A principal característica desses algoritmos é que o processo de busca é baseado no conceito de populações. Por meio da adaptação de uma sequência de gerações compostas por um determinado número de indivíduos, os algoritmos evolutivos realizam um processo de busca direcionada pela capacidade de sobrevivência (*fitness*) dos indivíduos, resultando numa busca mais eficiente e que cobre um espaço de busca bem maior que outras técnicas tradicionais (SIVANANDAM; DEEPA, 2008).

2.3.1 Estrutura do algoritmo evolutivo

Apesar de cada técnica evolutiva possuir suas características particulares, todas elas compartilham uma estrutura fundamental, que pode ser observada no algoritmo 1, adaptado de Ashlock (2006).

Algoritmo 1 Algoritmo evolutivo

- 1: Gerar população inicial de estruturas (indivíduos);
 - 2: Avaliar a população inicial;
 - 3: **while** Condição de parada não satisfeita **do**:
 - 4: Selecionar indivíduos para reprodução;
 - 5: Produzir variações novas dos indivíduos selecionados;
 - 6: Testar a qualidade dos indivíduos;
 - 7: Substituir a população anterior pela nova;
 - return** Indivíduo com maior fitness;
-

Nas técnicas de computação evolutiva, o primeiro passo do processo evolutivo é a criação da população inicial de soluções, codificadas de acordo com determinada estrutura de dados. Essas soluções iniciais podem ser geradas de forma estocástica, de acordo com algum padrão ou heurística, ou mesmo ser a saída de outros algoritmos. Em seguida, uma função objetiva é usada para mensurar a qualidade das soluções atuais, servindo de referência para decidir quais indivíduos pertencem a possíveis áreas de interesse na superfície de soluções. O processo de seleção é o passo seguinte, que ocorre com base nas métricas da função *fitness*. Os indivíduos selecionados são passados adiante sendo submetidos a operadores de variação genética, que pode ocorrer na forma de perturbações aleatórias na estrutura de um único indivíduo ou com a troca de material genético entre diferentes indivíduos da população. Os operadores que realizam mudanças em apenas uma estrutura são chamados de variação unitária ou mutação, e aqueles que efetivam a troca de material entre estruturas diferentes, variação binária ou *crossover*. Uma diretriz do processo evolutivo é que no laço principal do algoritmo sejam escolhidos indivíduos de forma que, em média, mais soluções adequadas sejam selecionadas (ASHLOCK, 2006).

O laço principal do algoritmo é um processo de iteração em que a população do algoritmo é continuamente atualizada por meio dos operadores de seleção e variação, cuja operação deve resultar numa movimentação dos indivíduos em direção à obtenção de estruturas genéticas melhores. O laço de evolução só é encerrado quando uma condição de parada do algoritmo é satisfeita, geralmente quando é encontrada uma solução ótima para o problema, determinada de acordo com a função *fitness* usada. Essa solução pode ser a

melhor possível ou apenas a melhor entre as soluções de uma região, sendo denominadas respectivamente de ótimos global e local (ASHLOCK, 2006).

As técnicas evolutivas não garantem a obtenção do ótimo global, o qual pode até mesmo ser desconhecido no problema. No entanto, esses algoritmos tendem a conseguir bons resultados no geral devido à forma como são projetados para explorar a superfície de soluções. Em contraste com outras abordagens de otimização, os algoritmos evolutivos são capazes de transitar entre diferentes regiões ótimas do espaço de busca. Mesmo que um dos indivíduos da população tenha alcançado um *fitness* ótimo, os outros membros da população estarão espalhados em volta desse ponto, podendo encontrar outras regiões de ótimo. Essa amplitude de busca é característica dos algoritmos evolutivos, e dependendo do problema tratado, pode se tornar tanto uma vantagem quanto desvantagem do seu uso (ASHLOCK, 2006).

Outra característica dos algoritmos evolutivos é que eles são pouco sensíveis ao estado inicial do problema, em contraste a abordagens de otimização como o gradiente descendente. Como os algoritmos evolutivos trabalham paralelamente com vários indivíduos, nem todas as soluções precisam ter um *fitness* bom para que a técnica funcione satisfatoriamente. Na maioria dos problemas é necessário que indivíduos ruins existam na população, os quais podem ser “ancestrais” de uma solução ótima global obtida em gerações futuras do algoritmo (ASHLOCK, 2006).

2.3.2 Algoritmos genéticos

Os algoritmos genéticos são uma das técnicas evolutivas mais comumente empregadas em problemas de otimização. Em um algoritmo genético tradicional, usa-se um vetor de *bits* de tamanho fixo como representação cromossômica dos indivíduos na população do algoritmo. Nesse vetor, cada posição (*bit*) representa uma característica específica do indivíduo, enquanto que o valor contido na posição representa como tal característica é expressa na solução (indivíduo) em questão. Nessa técnica, costuma-se assumir que a representação em forma de vetor é uma coleção de características estruturais as quais possuem pouca ou nenhuma interação entre si, de forma análoga a como ocorre na biologia, onde cada gene em um organismo representa uma entidade que é estruturalmente independente das representadas nos outros genes (SIVANANDAM; DEEPA, 2008).

Os principais operadores genéticos empregados nesta técnica são o *crossover* e a mutação de *bits*. No *crossover*, dois vetores (indivíduos) da população atual são selecionados para a reprodução, e novos indivíduos são gerados por meio da combinação de segmentos de *bits* desses vetores. Já no operador de mutação, modifica-se o valor de um único *bit* do vetor selecionado, inibindo ou expressando a característica referente ao *bit* e assim gerando um novo indivíduo. Uma distinção fundamental entre os operadores aplicáveis nos algoritmos genéticos é se estes visam ou não inserir novos genes na população do algoritmo (SIVANANDAM; DEEPA, 2008).

Em uma implementação canônica dessa técnica, a manipulação cromossômica é feita de forma restrita seguindo a interpretação estrutural dos genes tomada da biologia. Por exemplo, dois genes referentes a uma mesma característica nos pais podem ser recombinados nos filhos em relação ao todo, mas não em relação aos seus valores. Outras características tradicionais são a seleção probabilística de indivíduos baseada nos valores de *fitness* e a evolução sem intercessão entre as gerações (SIVANANDAM; DEEPA, 2008).

2.3.3 Programação genética

A programação genética (GP)⁶ funciona de forma análoga aos algoritmos genéticos, possuindo, no entanto, algumas modificações quanto à representação cromossômica e aos operadores empregados. Essa técnica usa uma estrutura de árvore para a representação cromossômica dos indivíduos, a qual possui tamanho variado e é composta de nós contendo funções e valores. Num indivíduo, os nós nas extremidades (folhas) representam variáveis pertencentes ao conjunto de todas as variáveis do universo de discurso, enquanto que os nós internos da árvore representam funções pertencentes ao conjunto de funções disponíveis. Dessa forma, cada árvore representa uma expressão a ser executada. A execução de um indivíduo é feita partindo do nó mais profundo e percorrendo a árvore da esquerda para a direita, onde as folhas assumem valores e os nós internos avaliam as respectivas funções com base nos valores de suas folhas. Os operadores usados na programação genética são similares aos dos algoritmos genéticos, no entanto, esses são projetados levando em consideração a representação estrutural de árvore. Um exemplo é o *crossover* de sub-árvore, onde uma das sub-árvores dos pais selecionados é trocada entre os mesmos, gerando dois novos indivíduos (SIVANANDAM; DEEPA, 2008).

⁶ Do inglês: *Genetic Programming*

2.3.4 Estratégias evolutivas

As estratégias evolutivas (ES)⁷ são técnicas que assumem uma representação cromossômica vetorial de tamanho fixo, e como nos algoritmos genéticos cada posição representa uma característica do indivíduo, possuindo, porém, valores reais em cada gene. Uma diferença apontada por Angeline (1996) é que essas características são encaradas sob uma perspectiva comportamental, em contraste ao pensamento estrutural dos algoritmos genéticos. Uma consequência disso é que os operadores empregados nas estratégias evolutivas envolvem interações não lineares arbitrárias entre genes de uma mesma característica, o que resulta numa exploração mais holística do espaço de soluções.

Os principais operadores genéticos dessa técnica são a mutação gaussiana e a recombinação intermediária. Na mutação gaussiana, um valor aleatório pertencente a uma distribuição gaussiana é adicionado a cada elemento do vetor de um indivíduo, gerando um novo indivíduo. Na recombinação intermediária é tomada a média dos vetores de dois indivíduos, gene a gene, gerando um novo indivíduo. Os efeitos desses dois operadores refletem a interpretação comportamental característica dessa técnica, uma vez que o valor de cada gene é levado em consideração nas operações genéticas para derivar novos vetores de indivíduos. Nas estratégias evolutivas, a seleção de pais é menos restritiva que em GA e GP. Em uma abordagem tradicional de ES, N indivíduos são selecionados uniforme e aleatoriamente, ou seja, sem usar um viés baseado no *fitness* dos indivíduos. Assim, produz-se com o uso de operadores de recombinação uma quantidade maior que N , dos quais serão selecionados N indivíduos de forma determinística. A seleção passa adiante os N melhores indivíduos, podendo ocorrer de duas formas: considerando apenas os indivíduos gerados ou estes e a população atual do algoritmo (SIVANANDAM; DEEPA, 2008).

2.3.5 Programação evolutiva

A programação evolutiva (EP)⁸ se assemelha às estratégias evolutivas, assumindo para a representação dos indivíduos o conceito de máquina de estados finitos. Os indivíduos nesta técnica são capazes de responder a estímulos do meio ambiente, e os operadores desenvolvidos para a técnica têm o objetivo de efetuar mudanças estruturais e comportamentais

⁷ Do inglês: *Evolutionary Strategies*

⁸ Do inglês: *Evolutionary Programming*

nos indivíduos com o decorrer do tempo. As representações cromossômicas nessa técnica tendem a ser específicas ao domínio do problema, sendo uma das representações mais comuns a de vetor real de tamanho fixo. Uma característica que distingue a programação evolutiva das outras técnicas evolutivas é que não há nessa a troca de material genético entre indivíduos, desta forma, os únicos operadores usados nessa técnica são os de mutação. Uma forma típica de seleção empregada em EP é a de escolher toda a população atual para participar das operações genéticas, mutando todos os N indivíduos e gerando N novos indivíduos, passando para a próxima geração apenas os N melhores dos $2N$ indivíduos (SIVANANDAM; DEEPA, 2008).

2.4 Mineração de processos

Mineração de processos é um novo campo de pesquisa que associa as áreas de mineração de dados e aprendizado de máquina ao contexto da modelagem e análise de processos de negócio. Na mineração de processos visa-se descobrir, monitorar ou melhorar processos de negócio por meio da extração de conhecimento sobre os dados gerados pelos mesmos, que são representados na forma de *logs* contendo traços da ocorrência de eventos durante a execução do processo. Essas informações, no entanto, tendem a estar presentes de forma não estruturada nos sistemas organizacionais, dessa forma, a extração de dados relativos aos processos organizacionais (que podem estar disponíveis em estruturas e sistemas diferentes) é um dos passos prévios para a mineração de processos (AALST, 2016).

A área de mineração de processos é composta por três tipos principais de mineração: descoberta, conformidade e melhoria de modelos de processo. Na figura 3 pode-se observar um esquema de como esses tipos de mineração se relacionam com o ambiente organizacional, ligando o processo real da empresa (e seus dados) aos modelos conceituais do processo. Atualmente, as abordagens propostas de mineração de processos, em sua maioria, usam técnicas e algoritmos heurísticos tradicionais, os quais funcionam com base em “regras de ouro” relacionadas a padrões nos processos de negócio, geralmente levando a bons resultados (AALST, 2016). No entanto, o número de estudos aplicando técnicas de inteligência computacional e aprendizado de máquina cresceu relativamente nos últimos anos (MAITA, 2015), onde destaca-se o uso de técnicas como algoritmos genéticos nos

tipos de mineração de processos, motivado pela sua relativa simplicidade e aplicabilidade a problemas diversos e a seus mecanismos de busca direcionada comprovadamente eficientes (ASHLOCK, 2006).

Figura 3 – Posicionamento dos três principais tipos de mineração de processos: descoberta, conformidade e melhoria

Fonte: Traduzido de Aalst (2016)

2.4.1 Descoberta de processos

Na descoberta, tem-se por objetivo obter um modelo de processo a partir de um conjunto de *logs* de eventos, sem qualquer conhecimento a priori de como é de fato o processo executado. A descoberta de modelos é o principal tipo de mineração de processos (uma vez que nem sempre o modelo do processo é conhecido), pois fornece o modelo que será usado como recurso na melhoria e conformidade. A descoberta de processos tem por objetivo visualizar, com base no comportamento concreto observado nos *logs* de eventos, qual é o processo de fato sendo executado na organização. A descoberta de processos pode também gerar modelos com informações a respeito de outras perspectivas do processo, como os recursos usados durante o processo ou o tempo transcorrido entre as atividades

(AALST, 2016). Por ser o foco deste trabalho, os aspectos da descoberta de processos são mais profundamente discutidos na seção 2.5.

Um dos primeiros algoritmos para a descoberta de modelos de processo foi o algoritmo *alpha*, que tem como entrada um conjunto de *logs* de eventos e produz na saída uma rede de Petri demonstrando o comportamento observado no conjunto. No *alpha*, percorre-se o conjunto de *logs* de forma a detectar padrões de dependência causal entre atividades, construindo o modelo com base nessas informações locais (AALST, 2016).

2.4.2 Conformidade de processos

Na conformidade de processos, um modelo já existente é confrontado com um conjunto de *logs* relativos ao mesmo processo. A conformidade tem por fim verificar se a realidade de um processo (que é expressa pelo conjunto de *logs*) está de acordo com o que foi formalizado no modelo de processo. Uma das principais aplicações desse tipo de mineração é na detecção de fraudes, em que o corpo de *logs* é confrontado com o modelo de processo, revelando possíveis fraudes nos pontos onde o modelo não é seguido pelo comportamento observado. Dessa forma, a conformidade de processos pode ser usada para detectar, localizar e explicar divergências entre o que foi planejado e o que de fato foi executado, mensurando a gravidade desses desvios (AALST, 2016).

2.4.3 Melhoria de processos

Na melhoria de processos, o objetivo é expandir ou melhorar um modelo de processo já existente usando informações do próprio processo contidas no conjunto de *logs*. Enquanto a conformidade busca medir o nível de alinhamento entre o modelo de processo e a realidade do mesmo, a melhoria tem por fim mudar ou ampliar o modelo *a priori* do processo. Existem dois tipos de melhoria: reparo e ampliação (AALST, 2016).

No reparo, visa-se modificar o modelo de processo já conhecido para que esse reflita melhor a realidade do processo. Por exemplo, quando duas atividades do processo foram modeladas como sendo sequenciais, mas no comportamento observado elas não ocorrem em ordem determinada, o modelo pode ser modificado para refletir esse comportamento paralelo (AALST, 2016).

Na ampliação, o objetivo é adicionar uma nova perspectiva ao modelo de processo a priori, inserindo informações observáveis por meio do cruzamento entre o modelo e os *logs*. Por exemplo, pode-se inserir no modelo informações a respeito do desempenho de execução do processo, mostrando os possíveis gargalos, níveis de serviço, frequência e tempo de processamento das atividades, ou ainda informações refletindo os recursos usados, métricas de qualidade, regras de decisão, entre outras informações (AALST, 2016).

2.5 Problema da descoberta de processos

A descoberta de processos de negócio é uma tarefa computacional não trivial, que tem sido alvo de crescentes estudos visando propor técnicas sofisticadas de análise que viabilizem essa tarefa, uma vez que é a mais fundamental das atividades de mineração de processos. A não trivialidade desse problema ocorre pelos mesmos motivos comuns às outras tarefas de mineração de dados, tal como a presença de ruído nos dados, espaços de busca extensos e complexos, etc. Aalst (2016) lista alguns problemas específicos da descoberta de processos que tornam o problema particularmente difícil:

- Não existem exemplos de execução negativos, ou seja, os *logs* representam apenas o que pôde acontecer no processo, ignorando as restrições de eventos que não puderam ocorrer.
- Devido à concorrência entre atividades, estruturas de *loop* e decisão, o espaço de busca de soluções tem uma superfície demasiadamente complexa. Além disso, o *log* de eventos costuma conter apenas uma fração de todos os comportamentos possíveis no processo de negócio.
- Não há relação direta entre o tamanho de um modelo de processo e o comportamento possível neste, ou seja, mesmo um modelo de tamanho pequeno pode permitir instâncias de execução extensas.

2.5.1 Descrição geral do problema de descoberta de processos

Os três principais tipos de mineração de processos são a descoberta, conformidade e melhoria de processos de negócio. Dentro desses tipos de mineração, é possível assumir variadas perspectivas de mineração de processos, nomeadamente: perspectiva de fluxo de

controle, organizacional ou de recursos, de dados ou de intervalo de tempo. Assim, este trabalho foca na descoberta de processos, assumindo nessa mineração uma perspectiva de fluxo de controle. A combinação da descoberta com a perspectiva de fluxo de controle é chamada de descoberta de processos. O problema geral da descoberta de processo pode ser descrito da seguinte forma: Dado um *log* de eventos L (como o observado na tabela 1), um algoritmo de descoberta de processos é uma função que mapeia L em um modelo de processo (como o da figura 4), de forma que este modelo seja representativo do comportamento observado no *log* de eventos. Como pode-se observar na tabela 1, o conjunto de *log* é composto de casos (que identificam as diferentes execuções de um processo de negócio) e de traços de eventos (que identificam quais eventos ocorreram em determinada instância do processo) (AALST, 2016).

Tabela 1 – Log compacto de eventos relativos a $L1 = [(a,b,c,d)^3, (a,c,b,d)^2, (a,e,d)]$

Caso	Traço de eventos
Caso 1	a, c, b, d
Caso 2	a, b, c, d
Caso 3	a, e, d
Caso 4	a, c, b, d
Caso 5	a, b, c, d
Caso 6	a, b, c, d

Não há uma restrição de qual tipo de modelagem deve ser usada para representar o modelo de processo, sendo que o *log* pode conter atributos que permitam compor diferentes modelagens, das mais simples às mais ricas em detalhes. Neste trabalho, foi considerado o conceito de redes de Petri (explorado na subseção 2.1.1) para a modelagem interna das semânticas de execução dos modelos de processo no GA. Para a modelagem do resultado, para fins de maior clareza visual, foi utilizado BPMN⁹.

Assim, a entrada do algoritmo em questão deve ser um *log* de eventos L , composto de múltiplos traços de eventos envolvendo um conjunto de atividades (como o visto na tabela 1). O objetivo do algoritmo de descoberta é fazer o mapeamento entre o conjunto de *logs* e o processo de negócio, por exemplo, descobrindo a partir de $L1$ um modelo de processo que consiga reproduzir cada traço contido neste, como observado entre o *log* $L1$ e a *WF-net* na figura 4. O modelo de processo descoberto deve ser idealmente uma *WF-net* estruturalmente consistente, conforme o conceito apresentado na seção 2.1.2. Apesar da

⁹ Do inglês: *Business Process Model and Notation*

WF-net descoberta deve representar todas as sequências de atividades observadas nos *logs*, é aceitável que o modelo permita sequências as quais não aparecem no conjunto de dados (AALST, 2016).

Figura 4 – Exemplo de processo descoberto a partir de L1, modelado usando BPMN

Fonte: Adaptado de Aalst (2016)

A representatividade do modelo (a qual foca principalmente na reprodução dos traços contidos nos *logs*) é mensurada de acordo com quatro critérios: *replay*, precisão, generalização e simplicidade (AALST, 2016):

- **Replay (ou fitness):** o modelo descoberto deve permitir executar o comportamento contido no *log* de eventos. Está ligado à reprodução dos traços de eventos no log.
- **Precisão:** o modelo não deve permitir a ocorrência de comportamentos não relacionados com o observado no conjunto de *logs*.
- **Generalização:** o modelo deve poder generalizar os exemplos de comportamento observados no conjunto de dados.
- **Simplicidade:** o modelo deve ser a solução mais simples possível capaz de explicar os comportamentos contidos no *log* de eventos.

Um modelo de processo com uma qualidade boa deve conter um bom balanceamento entre esses quatro critérios. Obter tal balanceamento é um dos desafios da descoberta de processos, uma vez que a melhoria em um dos critérios pode influenciar direta ou inversamente as outras qualidades do modelo. Nesse quesito, os algoritmos evolutivos são ideais para tal atividade, uma vez que possuem uma forma de operar baseada na otimização de uma função objetiva em que se pode considerar mais de um critério de avaliação dos modelos, guiando assim a otimização com base nos valores em cada critério (AALST, 2016).

2.5.2 Desafios da descoberta de processos

Em Aalst e Weijters (2004) foi levantada uma série de desafios da atividade de mineração de processos os quais, apesar de serem tratados em diferentes níveis de satisfação pelas abordagens da literatura, continuam sendo uma dificuldade tanto para a melhoria nas atuais abordagens quanto para a sua satisfação em novas abordagens a serem propostas. Esses desafios são:

- **Tarefas ocultas:** na descoberta de processos, assume-se que todas as atividades relevantes à execução do processo de negócio em questão foram devidamente registradas nos *logs*. No entanto, é possível que determinadas tomadas de decisão adotadas no processo não tenham sido registradas no log, o que dificulta a sua representação no processo minerado. Apesar de não registradas, as consequências dessas decisões ainda são expressas nos *logs* por meio do comportamento de execução do processo, por exemplo quando há a execução paralela de um conjunto de atividades, mas a decisão de executá-las paralelamente (uma estrutura lógica do tipo *AND*) não é informada no log. Essa dificuldade se torna ainda menos trivial em processos de estrutura mais complexa, tornando a detecção de tarefas ocultas ainda mais difícil.
- **Tarefas duplicadas:** podem existir casos onde um modelo de processo possua dois nós referentes a uma mesma tarefa do processo, sendo difícil diferenciar a ocorrência de tais tarefas duplicadas nas diferentes execuções do processo, por conseguinte dificultando a descoberta do modelo. A existência de tarefas duplicadas é diretamente relacionada à existência de tarefas ocultas no processo, sendo que modelos possuindo tarefas ocultas e nenhuma tarefa duplicada podem ser convertidos em modelos com tarefas duplicadas e sem a existência de tarefas ocultas.
- **Estruturas de escolha não livre:** estruturas de escolha não livre são aquelas onde uma determinada decisão no processo precisa ser executada de forma sincronizada, sendo tal decisão controlada por outras decisões tomadas em partes diferentes do processo. Tal comportamento global no processo dificulta a mineração de tais estruturas, uma vez que é necessário ao algoritmo empregado minerar o processo sob uma perspectiva global.
- **Estruturas de *loop*:** num processo de negócio, podem existir tarefas que são executadas de forma repetida, sendo esse tipo de comportamento representado como

uma estrutura de laço (*loop*). Tal caso de *loop*, partindo de uma tarefa e retornando à mesma, é trivialmente descoberto na mineração. No entanto, quando o *loop* no processo apresenta um comportamento mais complexo, como o de retornar para um ponto anterior qualquer do processo, tal tarefa se torna consideravelmente complexa uma vez que podem ocorrer sequências múltiplas de uma mesma tarefa dentro de uma instância do processo.

- **Diferentes perspectivas:** um desafio na mineração de modelos de processo é a inclusão neste de informações extras presentes nos *logs*, de forma a enriquecer o modelo de processo descoberto. Apesar de o fluxo de controle do processo ser o foco da maioria das abordagens de descoberta de processo, a adição de outras perspectivas como a organizacional, temporal, de fluxo de dados, de aplicação, etc., é uma tarefa de interesse na descoberta de modelos de processo.
- **Ruído:** apesar de ruído (no contexto de logs de processo) também ser definido como tarefas incorretamente registradas nos *logs* (como na ocorrência de atrasos ou erros na atividade de log), o conceito mais adotado de ruído na descoberta de processos é o equivalente ao de *outliers* na mineração de dados. Uma vez que é impossível ao algoritmo de descoberta diferenciar os potenciais erros de registro dos comportamentos infrequentes ou raros (*outliers*), toma-se este último como conceito de ruído na descoberta de processos. Alguns algoritmos de descoberta são especializados em lidar com esse problema da descoberta, dentre eles os algoritmos evolutivos.
- **Dados incompletos:** esse desafio está diretamente relacionado aos ruídos no log. Tratam-se de casos onde o conjunto de dados de *log* não possui informações suficientes sobre os traços de execução do processo. Devido à grande quantidade de traços possíveis em um processo de negócio complexo, a infrequência de certos padrões impossibilita ao algoritmo derivar dos *logs* o modelo completamente correto.

Turner et al. (2012) lista alguns desses desafios, conforme pode-se observar na tabela 2, colocando quais abordagens focam na solução desses desafios estruturais. Como pode-se observar, as técnicas computacionais inteligentes são as mais usadas na literatura para a solução desses obstáculos da descoberta de processos. Tanto Turner et al. (2012) quanto Maita (2015) indicam que tal categoria de técnicas, em que se incluem algoritmos

evolutivos, redes neurais, técnicas *fuzzy*, dentre outras, têm sido o foco de pesquisa da área de mineração de processos nos anos recentes.

Tabela 2 – Relação entre desafios e técnicas empregadas na descoberta de processos

	Baseadas em mineração de dados	Heurísticas	Computação inteligente	Abordagem Markoviana
Ruído	x	x	x	x
Tarefas ocultas			x	
Tarefas duplicadas	x		x	
Escolha não livre		x	x	
<i>Loops</i>	x	x	x	
Diferentes perspectivas			x	
Concorrência	x	x	x	x

Fonte: Adaptado de Turner et al. (2012)

2.6 Definição do problema da descoberta de processos não estruturados

A descoberta de processos de negócio é uma tarefa computacional difícil que vem sendo alvo de crescentes estudos, os quais visam propor técnicas sofisticadas de análise para a viabilização dessa tarefa. A mineração de processos funciona com a premissa de, a partir do comportamento observado no conjunto de *logs* de um processo, derivar o modelo do processo até então desconhecido. Inicialmente, a maioria dos conjuntos de dados usados na validação de técnicas de mineração de processos na literatura foi composta de *logs* artificialmente gerados. Dessa forma, os primeiros algoritmos propostos eram capazes de minerar processos bem estruturados, requerendo um alto nível de qualidade dos *logs* (conjuntos completos e possuindo apenas os eventos de interesse) e com algum esforço mínimo de filtragem dos *logs* em casos mais problemáticos.

No entanto, processos de negócio reais não pertencem a contextos restritivos e inflexíveis de execução, os quais proporcionariam essas condições de qualidade de *logs*. As soluções mais populares de apoio à execução de processos fornecem um ambiente altamente adaptável e irrestrito de execução das tarefas de processo, não especificando um comportamento definido para o processo. A execução de um processo em tal contexto resulta em um comportamento altamente diverso e não estruturado, o que é refletido nos *logs* do processo. Tal característica nos *logs* destaca a fragilidade dos algoritmos até então

propostos, resultando na mineração de processos incompreensíveis devido ao excesso de comportamentos observados. Apesar de estarem corretamente refletindo a realidade, tais modelos de processo *espagete* não fornecem um apoio satisfatório à análise do processo em si, como se pode observar no na Figura 5.

Figura 5 – Exemplo de um modelo de processo minerado a partir do *log* de processo não estruturado

Fonte: Gabriel Lucas Cantanhede da Silva, 2018

A quantidade expressiva de eventos, aliada ao excesso de arestas, impossibilita a interpretação intuitiva do processo, sendo difícil analisar o fluxo de controle do processo. A causa do problema de mineração de processos não estruturados está em certas premissas implicitamente estabelecidas ao decorrer do desenvolvimento da área de pesquisa, as quais não se mantêm válidas em um contexto de processos realistas e não estruturados:

- **Premissa #1:** todos os *logs* de eventos eram considerados confiáveis, ou seja, qualquer evento presente nos *logs* eram considerados como sendo relacionados à uma atividade lógica no processo. No entanto, a realidade é que muitas tarefas não são registradas no log, enquanto que outros eventos não correspondem a nenhuma atividade do processo. Além disso, os *logs* não possuem uma estrutura homogênea, contendo eventos de diferentes níveis de abstração reunidos no mesmo log, dentre outros problemas.
- **Premissa #2:** existe um processo perfeitamente delimitado, que é refletido nos *logs*. Essa premissa assume a existência de uma solução perfeita para o problema da

descoberta de processos, sendo que o resultado da mineração deve ser um modelo de processo que explique perfeita, precisa e completamente o comportamento observado nos *logs*. No entanto, em processos não estruturados essa tarefa é inviável uma vez que seus *logs* refletem um nível alto de comportamento não estruturado, sendo necessário abstrair certos detalhes menos relevantes do processo. Além disso, a mineração de processos tradicional foca apenas em uma perspectiva do processo, comumente no fluxo de controle. Porém a combinação de diferentes perspectivas de processo com o fluxo de controle é frequente em ambientes não estruturados, sendo necessário levar as diferentes perspectivas do processo em consideração.

Para simplificar e visualizar os modelos de processo não estruturados minerados, algumas abordagens estabeleceram métricas de significância e correlação, sob as quais embasam critérios para a manipulação dos modelos. Embora essa forma de descoberta de modelos de processo não estruturados já viabilize a análise em um ambiente caracterizado pela presença significativa de ruído nos *logs*, a mineração realizada pelas soluções que implementam essa estratégia (como o plugin Fuzzy Miner do ProM) resulta em um mapa do processo (que se assemelha a uma máquina de estados finitos) onde é possível observar apenas a relação de causalidade entre as atividades ou os agrupamentos de atividades, sem ser no entanto possível saber claramente qual é o fluxo de controle do processo ou as relações explícitas de paralelismo entre as atividades no processo.

Devido à pouca exploração deste problema na literatura, além do grande potencial presente na mineração de processos não estruturados, este trabalho teve como objetivo contribuir para a solução do problema da descoberta de modelos de processo não estruturados.

3 Revisão da literatura

A revisão de literatura é um método que propõe a identificação, avaliação e interpretação de estudos relevantes associados a uma questão de pesquisa específica, a um tópico ou mesmo a um fenômeno de interesse (KITCHENHAM, 2007). Nesta, os estudos que contribuem para o processo realizado são chamados estudos primários, sendo a revisão caracterizada como um estudo secundário.

A revisão de literatura deste trabalho fora realizada seguindo as diretrizes e técnicas de revisão sistemática da literatura estabelecidas por Kitchenham (2007), a fim de aumentar a probabilidade de encontrar trabalhos relacionados relevantes ao tema deste trabalho. A diferença entre a revisão sistemática de literatura e os outros tipos de revisão de literatura não sistemáticos é que esta usa um procedimento bem definido, o qual visa garantir a reprodutividade do estudo, além de buscar evitar vieses (BIOLCHINI et al., 2005). Os passos desta revisão podem ser agrupados em três fases principais: o planejamento, a condução e o relatório da revisão. A seção 3.1 descreve os detalhes do planejamento, a 3.2 detalha a condução da mesma, e a seção 4 contém os resultados da revisão de literatura.

3.1 Planejamento da revisão de literatura

O planejamento da revisão de literatura fora estruturado da seguinte forma: identificação da necessidade de uma revisão de literatura; especificação das questões de pesquisa; e desenvolvimento de um protocolo de revisão.

3.1.1 Necessidade de uma revisão de literatura

O autor deste trabalho faz parte de um grupo de pesquisa interdisciplinar, pertencente à área de sistemas de informação. As atividades desenvolvidas por esse grupo envolvem majoritariamente a área de mineração de dados, associada ao uso de técnicas de inteligência computacional e de BPM. Dessa forma, a área de pesquisa de mineração de processos é de grande interesse deste grupo, despertando a necessidade de realizar uma análise objetiva dos estudos primários publicados relativos a esse contexto. A realização de uma revisão de literatura é ainda mais importante diante do fato de que, até este instante,

nenhuma revisão de literatura relacionada ao uso de computação evolutiva na mineração de processos fora publicada.

Usando esta revisão de literatura como subsídio, o presente grupo de pesquisa, além de possíveis outros pesquisadores interessados, poderão realizar trabalhos na área, evitando a perda de tempo e de recursos de pesquisa e consequentemente focando apenas na proposição de novas abordagens, usando a literatura levantada como base para seus trabalhos. Optou-se por seguir a metodologia proposta por Kitchenham (2007) devido a sua característica de minimizar a ocorrência de viés, além de propiciar a recuperação dos conjuntos de dados usados.

3.1.2 Questões de pesquisa

Para a realização desta revisão de literatura, foram propostas questões de pesquisa específicas direcionadas para a área de interesse, visando descobrir como as técnicas de inteligência computacional em questão estão sendo aplicadas na área de mineração de processos. A principal intenção na realização desta revisão de literatura é a de avaliar os artigos da literatura que empregam técnicas como algoritmos genéticos, programação genética e estratégias evolutivas nos tipos de mineração de processos.

QP1. Quais as técnicas evolutivas que estão sendo usadas na área de mineração de processos?

Para essa questão, foram consideradas apenas as técnicas que envolvem o conceito de evolução populacional por meio de gerações de indivíduos, nomeadamente algoritmos genéticos, programação genética, estratégias evolutivas e programação evolutiva.

QP2. Que tarefas de mineração de dados estão sendo executadas com o uso dessas técnicas?

Para essa questão, foram consideradas as seguintes tarefas de mineração de dados: predição categórica; regressão; identificação de tendências; clusterização; descoberta de padrões frequentes e de regras de correlação e associação (HAN; KAMBER; PEI, 2012).

QP3. Qual tipo de mineração de processos está sendo feito com o uso dessas técnicas evolutivas?

Para essa questão, foram considerados os três principais tipos de mineração de processos: descoberta de modelos de processo a partir de um conjunto de *logs*; análise da conformidade de um modelo de processo em relação aos traços de eventos presentes nos *logs*; melhoria do modelo de processo com base em *logs* anteriores do mesmo processo (AALST, 2016).

3.2 Protocolo

O protocolo é onde são especificados os métodos que serão usados durante a realização de uma revisão de literatura (KITCHENHAM, 2007). Os componentes principais a serem especificados no protocolo de uma revisão de literatura são: fontes de dados; estratégia de busca; estratégia de seleção dos estudos primários; método de extração; e resumo dos dados. O protocolo detalhado da revisão de literatura encontra-se no apêndice A deste documento.

3.3 Resultados

A distribuição temporal dos artigos classificados como estudos primários pode ser observada na figura 6:

Figura 6 – Distribuição de estudos primários por ano de publicação

Fonte: Gabriel Lucas Cantanhede da Silva, 2018

Como pode ser observado, o número de pesquisas envolvendo técnicas evolutivas para a mineração de processos aumentou consideravelmente nos últimos anos, revelando

o interesse da comunidade acadêmica pelo tema de pesquisa em questão. Nos passos a seguir são respondidas as questões de pesquisa, e na seção seguinte é apresentada a análise dos estudos primários selecionados.

QP1. Quais as técnicas evolutivas que estão sendo usadas na área de mineração de processos?

Para essa questão, foram consideradas apenas as técnicas que envolvem o conceito de evolução populacional por meio de gerações de indivíduos, nomeadamente algoritmos genéticos (GA), programação genética (GP), estratégias evolutivas (ES) e programação evolutiva (EP). Observou-se uma predominância no uso de técnicas de GA, utilizada em vinte e três estudos, seguida de GP em quatro estudos e duas ocorrências de ES baseada em mutação. Não fora identificada nenhuma ocorrência do uso de EP.

QP2. Que tarefas de mineração de dados estão sendo executadas com o uso dessas técnicas?

Para essa questão, foram consideradas as seguintes tarefas de mineração de dados: predição categórica; regressão; identificação de tendências; clusterização; descoberta de padrões frequentes e de regras de correlação e associação (HAN; KAMBER; PEI, 2012). Observou-se nos estudos analisados que não há uma ligação explícita entre as tarefas de mineração de dados e as de mineração de processos. Apenas em Lee et al. (2016) usou-se tarefas de mineração de dados, em que foram empregadas as tarefas de descoberta de padrões frequentes e de regras de associação.

QP3. Qual tipo de mineração de processos está sendo feito com o uso dessas técnicas evolutivas?

Para essa questão, foram considerados os três principais tipos de mineração de processos: descoberta de modelos de processo; análise de conformidade; melhoria do modelo de processo (AALST, 2011). Com exceção de Lee et al. (2016), o qual realizou a descoberta de regras de associação dentro do contexto de processos de negócio, todos os artigos analisados empregaram a descoberta de modelos de processo.

3.3.1 Tipos de mineração de processos

O principal tipo de mineração de processos empregado nos estudos analisados foi a descoberta de modelos de processo, com a exceção de apenas um estudo primário (LEE et al., 2016) onde foi feita a descoberta de regras de associação para a melhoria de processos industriais. A escolha pelo tipo descoberta é justificada pelo fato de que esta é uma das tarefas mais complexas e desafiadoras no campo de mineração de processos, além de servir de suporte para a execução da conformidade e melhoria de processos (AALST, 2016). A maioria dos artigos que fazem a descoberta focam em modelos de processo imperativos (onde a ordem de execução das tarefas é explícita). A única exceção é Broucke, Vanthienen e Baesens (2014), onde o objetivo foi obter processos declarativos no qual se descobre as condições que devem ser satisfeitas para que determinada tarefa seja executada. Dos vinte e nove estudos analisados, apenas Tsai e Chen (2009) e Tsai, Jen e Chen (2010) abordaram a mineração de processos não estruturados. Pode-se concluir que a descoberta de processos não estruturados ainda demanda considerável interesse dos pesquisadores, indicando que ainda há espaço para maiores melhorias nas abordagens propostas.

3.3.2 Utilização dos *logs* e criação da população inicial

A maioria dos estudos analisados considerou a totalidade do conjunto de *logs* do processo no funcionamento do algoritmo, com exceção de Lee et al. (2016) (onde não foi possível identificar como os logs são utilizados) e Yang et al. (2015) (onde segmentou-se o conjunto de logs). Em Tsai e Chen (2009) e Tsai, Jen e Chen (2010) calcula-se o *fitness* de um indivíduo com base em um segmento específico do log e no conjunto completo. Em Bratosin, Sidorova e Aalst (2010) e Bratosin, Sidorova e Aalst (2011) foi proposta uma abordagem que reduz a quantidade de processamento necessária para o algoritmo, calculando o fitness com base em uma amostra do conjunto de logs e testando a qualidade dos indivíduos para o conjunto completo apenas quando o *fitness* ótimo em relação à amostra for alcançado. Yang et al. (2015) apresentou uma abordagem onde se segmenta os *logs* em subconjuntos antes de dar início à descoberta de modelos de processo. Yang et al. (2015) levou em consideração conhecimentos a priori de especialistas nessa segmentação,

visando aumentar a capacidade de detecção de padrões locais nos *logs* para a criação de indivíduos de melhor qualidade na população inicial do GA.

Quanto à inicialização do algoritmo, a maioria dos estudos emprega uma heurística que leva em conta a dependência entre as atividades no log para criar a população inicial. Em Medeiros, Weijters e Aalst (2007) é apresentada uma abordagem que inclui uma heurística de criação dos indivíduos da população inicial. Nessa heurística, quanto mais frequente uma transição entre duas atividades aparece no log, mais provável é de essa transição ser utilizada na criação dos indivíduos, em contraste com a típica inicialização aleatória da população inicial. Buijs, Dongen e Aalst (2012), Seleem, Mohamed e Hussain (2013), Jian e Li (2013) e Broucke, Vanthienen e Baesens (2014) construíram a população inicial de forma aleatória. Em Lee et al. (2016) criou-se a população com base na detecção de padrões frequentes de atividades nos *logs*. Em Molka et al. (2015a) e Molka et al. (2015b) a população foi gerada com a mutação de modelos sequenciais de atividades obtidos de acordo com a ordem em que as atividades aparecem no *log*. Em Eck, Buijs e Dongen (2015) criou-se a população com a junção de diferentes traços de atividades do log. Em Khezerlou e Alizadeh (2014) usou-se a técnica de recozimento simulado, e em Yang et al. (2015) foram usados plug-ins da ferramenta ProM para gerar os indivíduos iniciais.

3.3.3 Representação estrutural e cromossômica dos modelos de processo

Todos os estudos analisados usaram redes de Petri na notação conceitual dos processos, o que se justifica devido a esta abordagem ser matematicamente formalizada e amplamente usada na literatura. Foram observadas três abordagens de representação estrutural dos modelos de processo: matrizes causais, árvores de processos e vetores. Dos vinte e nove estudos primários, vinte e dois usaram matrizes causais e cinco usaram árvores de processo como representação interna de processo no algoritmo. Lee et al. (2016) foi o único estudo que utilizou a representação de vetor, visto que seu objetivo foi representar regras de associação. Em Seleem, Mohamed e Hussain (2013) não foi possível determinar a representação interna utilizada. As duas principais representações (matrizes e árvores) apresentam vantagens e desvantagens na sua utilização: Matrizes causais permitem a existência de inconsistências estruturais nos modelos de processo (tais como impasses na execução, finalização imprópria do processo, etc.) (BUIJS; DONGEN; AALST,

2012) e requerem o uso de operações de reparação do cromossomo durante as operações genéticas (AALST; MEDEIROS; WEIJTERS, 2005). Porém essas possuem tamanho fixo (sem repetição de tarefas no cromossomo) (MEDEIROS; WEIJTERS; AALST, 2006) e conseguem expressar operações lógicas complexas (YANG et al., 2015). As árvores de processos evitam a criação de modelos de processo inconsistentes, reduzindo o espaço de busca de soluções a apenas modelos factíveis (AALST; BUIJS; DONGEN, 2012). No entanto, algumas estruturas de processo não podem ser representadas por essa estrutura (YANG et al., 2015).

3.3.4 Técnicas evolutivas empregadas

Vinte e três estudos primários usaram a técnica de GA, implementando todos os mecanismos relacionados a essa técnica evolutiva. Embora apenas quatro artigos tenham indicado explicitamente usar a técnica de GP, pode-se argumentar que todas as abordagens de GA apresentadas poderiam ser classificadas como GP uma vez que os indivíduos (matrizes causais, árvores de processo ou regras de associação) dos algoritmos propostos possuem uma estrutura que se assemelha à de um programa computacional (o qual pode ser entendido como uma regra ou processo de negócio). Observou-se duas adaptações da técnica de GA, incluindo mecanismos de vírus (JIAN; LI, 2013) e de “*slippage*” (LEE et al., 2016), representando fenômenos observados no campo da genética. Em Molka et al. (2015a) e Molka et al. (2015b) empregou-se a técnica ES baseada na operação de mutação, tendo o objetivo de alcançar a solução ótima global sem dar saltos no processo evolutivo que descaracterizassem as semelhanças entre uma geração e as suas antecessoras, obtendo assim uma boa cobertura do espaço de busca. Na solução proposta, monitora-se a diversidade genética da população do algoritmo de forma objetiva, visando direcionar as estratégias de atuação do operador de mutação para evitar uma convergência precoce.

3.3.5 Técnicas auxiliares

Durante a análise dos estudos primários, observou-se o uso de outras técnicas de otimização para auxiliar o processo de busca dos algoritmos evolutivos. Em Cheng, Ou-Yang e Juan (2012) empregou-se a técnica de otimização por enxame de partículas para

auxiliar o mecanismo de elitismo do GA. Em Cheng, Ou-Yang e Juan (2015a) e Cheng, Ou-Yang e Juan (2015b) continuou-se a abordagem desenvolvida em Cheng, Ou-Yang e Juan (2012), usando conjuntamente as técnicas de otimização por enxame de partículas e evolução diferencial, afirmando que estas técnicas são ideais para auxiliar os mecanismos de elitismo e mutação do GA, respectivamente. Khezerlou e Alizadeh (2014) usou a técnica de recozimento simulado para obter uma população inicial de maior qualidade, em contraste com a abordagem de inicialização aleatória da população do GA. Seleem, Mohamed e Hussain (2013) empregou a técnica de colônia de abelhas para auxiliar o GA, limitando o processo de busca aos melhores pontos do espaço de soluções. Vázquez-Barreiros, Mucientes e Lama (2014) e Yang et al. (2015) usaram plug-ins da ferramenta ProM para processar os conjuntos de *log* usados, cujos modelos de processo resultantes serviram de apoio para a criação da população inicial do GA.

3.3.6 Operadores de correção

Foram identificados dois tipos de operadores de correção dos indivíduos, um relativo ao domínio específico da aplicação em Lee et al. (2016) e outro associado ao uso de matrizes causais em Vázquez-Barreiros, Mucientes e Lama (2014). O operador de reparação em Vázquez-Barreiros, Mucientes e Lama (2014) tem por fim manter a consistência das relações causais nos indivíduos após as operações genéticas. Em Aalst, Medeiros e Weijters (2005) são reforçadas restrições visando manter a consistência do cromossomo no momento da criação dos indivíduos, porém não é possível concluir que essas restrições são usadas como operador de correção durante a execução da abordagem proposta.

3.3.7 Operadores genéticos

Nos artigos analisados, observou-se a implementação de quatro tipos diferentes de operadores de *crossover*: *crossover* de um ponto, heurístico, baseado em árvores e uniforme. Dezoito artigos implementaram o *crossover* de um ponto, conforme introduzido por Aalst, Medeiros e Weijters (2005). Neste, escolhe-se uma tarefa em comum nos pais selecionados para a reprodução, segmentando em seguida os conjuntos de entrada e saída da tarefa com base em um ponto de corte escolhido nesses conjuntos, por fim recombinando os

segmentos nos filhos gerados. O *crossover* heurístico funciona de forma semelhante ao de um ponto, usando heurísticas para a escolha do ponto de corte ou dos indivíduos usados na operação. Em Feng, OuYang e Li (2011) o ponto de corte é escolhido com base na métrica de *replay* dos indivíduos selecionados. Em Vázquez-Barreiros, Mucientes e Lama (2014) a escolha do ponto é baseada em uma função de densidade probabilística, que associa às atividades incorretamente acionadas nos indivíduos candidatos uma probabilidade uniforme inversamente proporcional ao número de erros, direcionando o *crossover* às regiões incorretas do modelo. Em Cheng, Ou-Yang e Juan (2015b) o *crossover* é feito entre um indivíduo selecionado e o melhor indivíduo encontrado até o momento. O *crossover* baseado em árvores é implementado em Buijs, Dongen e Aalst (2012), Khezerlou e Alizadeh (2014), Eck, Buijs e Dongen (2015) e Yang et al. (2015). Nesse, escolhe-se um nó na árvore de cada pai selecionado, trocando essas sub-árvores nos filhos gerados. O *crossover* uniforme (implementado apenas por Lee et al. (2016)) é semelhante ao de um ponto, assumindo a quantidade máxima de pontos de corte durante a operação de *crossover*. Em Aalst, Buijs e Dongen (2012), Molka et al. (2015b) e Molka et al. (2015a) não foram implementados operadores de *crossover*. A respeito dos operadores de mutação, pôde-se observar seis tipos diferentes: mutação simples, heurística, de relação entre atividades, baseada em árvores, estrutural e “*slippage*”. Dezesseis artigos implementaram a mutação simples, como proposto inicialmente em Aalst, Medeiros e Weijters (2005). Nessa, é feita a adição, remoção ou recombinação das atividades presentes nos conjuntos de entrada e saída da atividade a ser mutada. A mutação heurística se assemelha ao funcionamento da simples. Em Feng, OuYang e Li (2011) a tarefa a ser mutada é escolhida com base na métrica de “*replay*” do indivíduo a ser mutado. Em Vázquez-Barreiros, Mucientes e Lama (2014) força-se a mudança nos conjuntos de entrada e saída da atividade até que o indivíduo mutado seja diferente do inicial. Além disso, as mutações devem ocorrer de acordo com as dependências anteriores e posteriores entre as atividades nos *logs*. A mutação na relação entre as atividades (implementada em Cheng, Ou-Yang e Juan (2012), Cheng, Ou-Yang e Juan (2015a), Cheng, Ou-Yang e Juan (2015b)) ocorre com a mudança dos operadores lógicos que relacionam as atividades nos conjuntos de entrada e saída (“Ou” exclusivo, “E”, sequência). Na mutação com base em árvores (implementada em Aalst, Buijs e Dongen (2012), Buijs, Dongen e Aalst (2012), Khezerlou e Alizadeh (2014), Eck, Buijs e Dongen (2015) e Yang et al. (2015)) pode-se adicionar ou remover um nó no indivíduo, modificar o conteúdo de um nó ou trocar sub-árvores inteiras de lugar em sua estrutura. Na mutação

estrutural (implementada em Molka et al. (2015a) e Molka et al. (2015b)) podem ocorrer três tipos de operação: deslocamento de partes do indivíduo de lugar no modelo (operação “*move*”), reestruturação de atividades de forma paralela, alternada ou em laço (operação “*complexify*”) ou de forma sequencial (operação “*simplify*”) no modelo de processo. A mutação “*slippage*” foi desenvolvida em Lee et al. (2016) especificamente para o contexto da mineração de regras de associação nos conjuntos de *logs*. Nesta, altera-se o comprimento do vetor que representa a regra através da replicação de um trecho do cromossomo (indivíduo do GA). Quanto aos operadores de elitismo, foi observado o uso simples de elitismo em vinte artigos. O elitismo simples garante que o melhor indivíduo da população do algoritmo seja sempre mantido durante o processo evolutivo. Em dez estudos analisados o elitismo não foi empregado (ou não foi especificado o seu uso). Foram identificados três tipos de operadores de seleção: seleção por torneio, probabilística e aleatória. A seleção por torneio foi implementada por doze artigos, como proposto inicialmente por Aalst, Medeiros e Weijters (2005). Na seleção por torneio, escolhe-se aleatoriamente cinco indivíduos da população para participar do torneio, e o indivíduo que possuir maior *fitness* será selecionado. A seleção probabilística foi implementada por Esgin, Senkul e Cimenbicer (2010), Jian e Li (2013) e Broucke, Vanthienen e Baesens (2014). Nessa seleção, a probabilidade de um indivíduo ser selecionado é diretamente associada ao seu valor de *fitness*. A seleção aleatória (empregada em Molka et al. (2015b) e Molka et al. (2015a)) não utiliza critérios para diferenciar os indivíduos durante a seleção, de forma que todos os indivíduos da população têm a mesma probabilidade de serem selecionados. Dos vinte e nove artigos analisados, doze não especificaram o operador de seleção implementado.

3.3.8 Critérios de avaliação (*fitness*)

Na literatura avaliada, observou-se o uso de quatro métricas principais para o cálculo do *fitness* dos indivíduos: *replay*, precisão, simplicidade e generalização. Vinte e quatro estudos propuseram funções *fitness* que englobam essas características, associando um peso a cada critério que permite dar uma prioridade diferenciada a cada métrica. Em (MEDEIROS; WEIJTERS; AALST, 2007) é introduzida uma fórmula para o cálculo do critério de precisão, objetivando penalizar o excesso de comportamento extra nos indivíduos do GA, sendo necessário conhecer o modelo do processo a priori para o cálculo

da precisão estabelecido nesta. Em Esgin, Senkul e Cimenbicer (2010) usou-se uma métrica específica baseada na estrutura de matriz empregada na solução. Em Zhao et al. (2012) e Zhao, Liu e Dai (2014) o critério empregado para o cálculo do *fitness* foi uma métrica de dependência e relação entre as atividades/atores do processo. Em Lee et al. (2016) o cálculo foi feito com base na capacidade de cada regra de associação minerada de prever a qualidade na produção do vestuário. Em Seleem, Mohamed e Hussain (2013) não foi possível determinar qual o critério usado no cálculo.

3.3.9 Critérios de parada

Foram identificados três critérios de parada nos artigos analisados: número de gerações, alcance da solução ótima e convergência genética. Vinte estudos empregaram o critério de número de gerações, onde o algoritmo executa até que o processo evolutivo ultrapasse um limite de gerações estabelecido. O critério de alcance da solução ótima foi empregado por dezessete dos estudos primários analisados, onde o algoritmo cessa a busca assim que um indivíduo com *fitness* 1 (perfeito) é encontrado. A convergência genética foi usada por nove estudos. Nela, monitora-se a diversidade entre os indivíduos na população atual do algoritmo, cessando a busca caso os indivíduos se tornem muito semelhantes (perda de diversidade).

3.3.10 Parametrização

Apenas dez dos vinte e nove artigos avaliados apresentaram um estudo da parametrização das técnicas empregadas. O estudo da parametrização visa determinar o valor ideal para cada parâmetro da técnica (percentual de crossover, mutação, elitismo, etc.) que proporcione o melhor funcionamento dos operadores genéticos, chegando mais facilmente em soluções ótimas.

3.4 Considerações finais

Por meio dessa revisão de literatura foi possível concluir que a área de mineração de processos desperta grande interesse da comunidade acadêmica. Pode-se observar um

crescimento significativo na quantidade de estudos da área, onde vê-se também o aumento na quantidade de pesquisas que empregam técnicas bem estabelecidas de inteligência computacional ao domínio da mineração de processos.

Esta revisão de literatura é um aprofundamento consequente de um mapeamento da área de mineração de processos, realizado pelo grupo de pesquisa em BPM da EACH-USP, que revelou a necessidade de estudar a atual aplicação das técnicas de inteligência computacional nesse contexto, dentre elas as técnicas evolutivas. A intenção desse mapeamento fora obter uma visão das abordagens já empregadas na literatura, com o objetivo de guiar a proposta de uma nova solução para a descoberta de modelos de processo, de forma a identificar eventuais pontos a serem evitados e contribuir para o processo de amadurecimento da área por meio da proposição de novas estratégias ou combinações de técnicas. Por meio da mesma pôde-se observar que o uso de técnicas evolutivas no contexto de aplicação de processos não estruturados ainda não foi demasiadamente explorado. Foi possível identificar o uso apenas de operadores genéticos simples, o que foi considerada uma lacuna para a melhoria da descoberta de processos tanto estruturados como não estruturados.

Na elaboração da proposta de GA apresentada na seção 4, foram levados em considerações os pontos positivos e negativos detectados principalmente nos artigos em questão. A representação cromossômica adotada na nossa proposta foi adaptada de Aalst, Medeiros e Weijters (2005), com algumas modificações que são explicadas na seção 4. Essa representação foi escolhida pelo seu maior poder de expressão de estruturas lógicas complexas, segundo a literatura.

Novas fórmulas para completude e precisão foram criadas para esta proposta, baseadas no conceito de transições entre atividades. Isso se fez necessário visto que a fórmula para completude proposta por Medeiros, Weijters e Aalst (2006), baseada na execução de traços de eventos, permite que vários resultados ruins, assim como o resultado ótimo, tenham completude igual a 1 (perfeito). Além disso, a fórmula de precisão proposta em Medeiros, Weijters e Aalst (2007) requer conhecimento do modelo de processo a priori e a fórmula de Buijs, Dongen e Aalst (2012) se aplica apenas a árvores de processo. A fórmula de completude proposta por Medeiros, Weijters e Aalst (2006) foi implementada como uma forma de validar o resultado final da nossa proposta de GA.

Nenhum dos artigos avaliados tentou validar suas abordagens com o uso de logs de processos não estruturados, enquanto que a nossa abordagem foi projetada especificamente

objetivando a mineração de processos não estruturados, preservando porém a capacidade de minerar processos estruturados. A nossa abordagem propôs operadores de reparação da matriz causal, o que foi identificado em apenas um dos estudos da literatura que usam essa representação (VÁZQUEZ-BARREIROS; MUCIENTES; LAMA, 2014). Foram implementados operadores genéticos que não explorados pelas abordagens na literatura: crossover uniforme, crossover de dois pontos, mutação direcionada e estratégias de troca de população.

O GA proposto não leva em consideração os critérios da generalização e simplicidade, no entanto a utilização desses critérios ainda não é consenso na literatura referente a GA. Nossa proposta de GA não teve como objetivo propor uma solução para o problema do tempo de execução, conforme apresentado em Bratosin, Sidorova e Aalst (2010), Bratosin, Sidorova e Aalst (2010) e Bratosin, Sidorova e Aalst (2011).

4 Estrutura do GA

Este capítulo visa apresentar a estrutura da nossa abordagem proposta de GA para a descoberta de modelos de processo não estruturado. Este capítulo apresenta a representação cromossômica utilizada no GA, a função fitness proposta para o cálculo da completude e precisão com base nas transições entre atividades presentes no log, os operadores genéticos e de correção do cromossomo e as estratégias de fluxo evolutivo.

4.1 Representação cromossômica

A representação cromossômica utilizada na abordagem proposta é a de matriz causal, adaptada a partir do conceito de matriz causal apresentado em Aalst, Medeiros e Weijters (2005), Medeiros, Weijters e Aalst (2006). Como pode-se observar no cromossomo apresentado na Tabela 3, uma matriz causal é uma estrutura de dados bidimensional com $n + 2$ linhas e três colunas, onde n representa o número de atividades distintas no processo. Cada índice na coluna atividade da matriz causal possui dois conjuntos de atividades nas colunas *input* e *output*, representando respectivamente as pré-condições e pós-condições da atividade no índice em questão. Os conjuntos de pré-requisitos e pós-requisitos possuem a mesma estrutura, onde a primeira posição do conjunto representa o operador lógico a ser aplicado (*AND* ou *xOR*) e a segunda posição representa o grupo de atividades do *input* ou *output*. Por exemplo, no cromossomo da Tabela 3 a atividade *A5* só pode ser executada se ambas *A3* e *A4* já tiverem sido executadas. Já após a atividade *A6* ter sido executada, *A7* ou *A8* poderão ser executadas, mas nunca as duas ao mesmo tempo.

Na nossa representação cromossômica, conjuntos de *input* e *output* que possuam apenas uma atividade são tratados como uma sequência simples, e nesses casos o operador lógico é ignorado. Por exemplo, as estruturas $\{\{ 'xOR' \}, \{A2\}\}$ e $\{\{ 'AND' \}, \{A2\}\}$ são equivalentes, significando que a atividade *A2* deve ser a única entrada ou saída da atividade no índice da matriz em questão.

A existência de conjuntos vazios no *input* e *output* denota que a atividade em questão faz parte, respectivamente, do início ou fim do modelo de processo. Dessa forma, uma atividade que faz parte do início do processo não possui pré-requisitos, e uma atividade do fim não possui pós-requisitos. Os conjuntos de atividades que fazem parte do início e

Tabela 3 – Representação cromossômica de um modelo de processo

Atividade	Input	Output
A1	{}, {}	{AND}, {A2}
A2	{xOR}, {A1}	{AND}, {A3, A4}
A3	{AND}, {A2}	{AND}, {A5}
A4	{xOR}, {A2}	{AND}, {A5}
A5	{AND}, {A3, A4}	{AND}, {A6}
A6	{AND}, {A5}	{xOR}, {A7, A8}
A7	{AND}, {A6}	{AND}, {A9}
A8	{AND}, {A6}	{xOR}, {A9}
A9	{xOR}, {A7, A8}	{}, {}
Início		{AND}, {A1}
Fim	{xOR}, {A9}	

Fonte: Gabriel Lucas Cantanhede da Silva, 2018

fim do modelo de processo são representados nos índices *Início* e *Fim* da matriz. Esses dois índices não estão na proposta original de matriz causal, e foram incluídos na nossa abordagem de forma a permitir expressar o início ou fim do modelo de processo com o uso dos operadores lógicos *AND* e *xOR*. Dessa forma o modelo de processo pode começar ou terminar através de uma ou mais atividades, o que pode ser útil na representação de processos não estruturados.

O uso de matriz causal como representação cromossômica proporciona alguns benefícios se comparado à árvore de processo, outra representação utilizada na literatura recente: (i) a matriz causal representa o cromossomo de forma menos esparsa, uma vez que cada atividade aparece apenas uma vez no índice da matriz, o que beneficia tanto a manipulação da estrutura de dados em baixo nível quanto a interpretação do modelo de processo em alto nível; (ii) ela permite denotar explicitamente a semântica dos operadores lógicos nos conjuntos de *input* e *output* de cada atividade no modelo, o que também beneficia o uso dessa estrutura em baixo e alto nível; e (iii) ela permite expressar laços e transições entre diferentes regiões do processo, permitindo descobrir modelos de processos com estruturas complexas como as presentes em processos não estruturados.

4.2 Função fitness

A função fitness proposta na nossa abordagem avalia a qualidade dos indivíduos do GA de acordo com as duas principais métricas de um modelo de processo: *completude* e *precisão*. Nesta abordagem, no entanto, *completude* e *precisão* não são calculadas em

função dos traços de eventos presentes no log, como originalmente proposto. Em vez disso, utiliza-se as transições entre as atividades no log e os operadores lógicos implicitamente contidos no mesmo. Para estes cálculos, são utilizadas tanto as transições de chegada quanto de saída para cada atividade no log.

Dessa forma, um modelo de processo é considerado completo se todas as transições entre atividades existentes no log existirem também no modelo de processo, junto com todos os operadores lógicos certos implicitamente presentes no log. Assume-se que se cada atividade no modelo de processo possui as transições de entrada e saída corretas, assim como os operadores lógicos corretos, então o modelo de processo formado pela junção de todas essas atividades e transições também está correto. Por outro lado, um modelo de processo é considerado preciso se ele possui apenas transições que aparecem no log.

Além da matriz causal do indivíduo sendo avaliado, o cálculo do fitness proposto neste trabalho requer o uso de uma estrutura secundária denominada por nós de matriz *Before/After* (B/A). A matriz B/A tem uma estrutura semelhante à da matriz causal utilizada na representação cromossômica, no entanto, nessa estrutura secundária não tem como objetivo representar o modelo do processo. Ela é utilizada apenas para representar o log de forma resumida, em uma estrutura que facilite a consulta do mesmo. Levando em consideração o log exemplo da Tabela 4, um exemplo de matriz B/A montada a partir do mesmo pode ser observado na Tabela 5.

Tabela 4 – Log de um processo estruturado simples, referente ao modelo de processo da Tabela 3

Instância	Traço de eventos
1	$\langle A1, A2, A3, A4, A5, A6, A7, A9 \rangle$
2	$\langle A1, A2, A4, A3, A5, A6, A8, A9 \rangle$
3	$\langle A1, A2, A3, A4, A5, A6, A8, A9 \rangle$
4	$\langle A1, A2, A4, A3, A5, A6, A7, A9 \rangle$

Fonte: Gabriel Lucas Cantanhede da Silva, 2018

A matriz B/A não leva em consideração os operadores lógicos ao descrever as transições de entrada e saída de cada atividade no log. Na matriz B/A, a coluna *Before* descreve as atividades que ocorrem imediatamente antes de uma determinada atividade no log, levando em consideração todos os traços presentes no log. Já a coluna *After* considera todas as atividades que ocorrem imediatamente depois da determinada atividade. Ao comparar-se a matriz B/A da Tabela 5 com o cromossomo da Tabela 3, pode-se observar que a matriz B/A possui algumas transições extras que não estão no modelo de processo.

Tabela 5 – Matriz B/A (para o log na Tabela 4)

Atividade	Before	After
A1	{}	A2
A2	A1	A3, A4
A3	A2, A4	A4, A5
A4	A2, A3	A3, A5
A5	A3, A4	A6
A6	A5	A7, A8
A7	A6	A9
A8	A6	A9
A9	A7, A8	{}
Início		A1
Fim	A9	

Fonte: Gabriel Lucas Cantanhede da Silva, 2018

Na Tabela 5, a atividade $A3$ é precedida pelas atividades $A2$ e $A4$ e seguida por $A4$ e $A5$, enquanto que na Tabela 3 $A3$ é precedida apenas por $A2$ e seguida apenas por $A5$. O que aparenta ser apenas uma sequência de atividades segundo o log é na verdade o resultado da execução de duas atividades em paralelo, onde uma delas é registrada no log imediatamente após a outra. Dessa forma, a nossa abordagem busca explorar as transições existentes no log de forma a montar um modelo de processo com as transições corretas, que conseguem reproduzir exatamente o comportamento do log.

Com base no cromossomo e na matriz B/A, a função *fitness* avalia os indivíduos do GA considerando as transições de atividades presentes no indivíduo e no log em questão. Os cálculos são executados por meio de duas funções complementares, *Transitional Precision* (TP) e *Logical Completeness and Precision* (LCP), definidas pelas Equações 1 e 2.

$$TP(Ind, Log) = \frac{1}{2n+2} * \left(\sum_{k=1}^{n+1} \frac{\#\{\{Ind[k, I]\} \cap \{Log[k, B]\}\}}{\#\{Ind[k, I]\}} + \sum_{k=1}^{n+1} \frac{\#\{\{Ind[k, O]\} \cap \{Log[k, A]\}\}}{\#\{Ind[k, O]\}} \right) \quad (1)$$

$$LCP(Log_{Ind}, Log) = \frac{n'}{n} * \left(\frac{1}{2n'+2} * \left(\sum_{k=1}^{n'+1} \frac{\#\{\{Log_{Ind}[k, B]\} \cap \{Log'[k, B]\}\}}{\#\{Log_{Ind}[k, B]\} + \#\{Log'[k, B]\} - \#\{\{Log_{Ind}[k, B]\} \cap \{Log'[k, B]\}\}} + \sum_{k=1}^{n'+1} \frac{\#\{\{Log_{Ind}[k, A]\} \cap \{Log'[k, A]\}\}}{\#\{Log_{Ind}[k, B]\} + \#\{Log'[k, B]\} - \#\{\{Log_{Ind}[k, B]\} \cap \{Log'[k, B]\}\}} \right) \right) \quad (2)$$

Na Equação 1, Ind é um indivíduo do GA representado por uma matriz causal, Log é a matriz B/A do log em questão, n é o número de atividades presentes no log, $n + 1$ é utilizado nos somatórios devido as linhas extras *Início* ou *Fim* nas duas matrizes, I e O são as colunas *Input* e *Output* no cromossomo de Ind , B e A são as colunas *Before* e *After* na matriz B/A, k é o índice das atividades, $\{X[k, c]\}$ é o conjunto de atividades na linha k e coluna c (onde $c = I, O, B$ ou A) da matriz X , sendo que X pode ser a matriz causal (Ind) ou a matriz B/A (Log), e $\#$ representa a cardinalidade de um determinado conjunto de atividades.

Na Equação 2, Log_{Ind} é um log artificial gerado aleatoriamente por meio da criação de traços de eventos a partir de um indivíduo Ind , Log' é uma versão reduzida de Log considerando apenas as atividades que aparecem em Log_{Ind} e na mesma ordem em que aparecem, n' é o número de atividades em Log_{Ind} , e os demais símbolos possuem a mesma definição que na Equação 1.

A equação $TP(Ind, Log)$ avalia a *precisão* levando em consideração o nível de similaridade entre as transições de atividades em um indivíduo Ind (uma matriz causal) e as transições presentes no Log (representado pela sua matriz B/A). Para cada atividade em Ind , o percentual de transições de atividades que estão também em Log é calculado. Esse cálculo é feito separadamente para as transições de entrada e saída, sendo feito uma média entre os dois percentuais resultantes. Dessa forma, TP penaliza apenas os comportamentos extras no indivíduo tendo como referência o Log , avaliando somente as transições de atividades. Os operadores lógicos presentes na matriz causal do indivíduo não são levados em consideração no cálculo de TP , visto que não é possível compará-los diretamente com os operadores lógicos na matriz B/A, uma vez que estes não estão explicitamente presentes no Log .

A equação $LCP(Log_{Ind}, Log)$ avalia conjuntamente *completude* e *precisão* em seus cálculos, levando em consideração o nível de similaridade entre as transições de atividades em Log_{Ind} e as transições em Log . Neste caso, para ambos os logs utiliza-se a estrutura de matriz B/A, sendo que o cromossomo do indivíduo é utilizado apenas para gerar o log artificial Log_{Ind} a ser comparado com Log . De forma simplificada, LCP mede o percentual de transições comuns entre ambos os logs em relação às transições não compartilhadas por ambos. Dessa forma, tanto *completude* quanto *precisão* são medidas juntamente. Conforme o número de transições presentes apenas em Log diminui, a *completude* aumenta; por outro

lado, conforme o número de transições presentes apenas em Log_{Ind} diminui, a *precisão* aumenta.

No pior caso, não haverá transições em comum, resultando em 0; no melhor caso, quando ambos os logs forem iguais, o resultado será 1. Uma vez que este cálculo é realizado em uma versão reduzida de Log , uma penalização deve ser aplicada de acordo com o tamanho dessa redução, definida por $\frac{n'}{n}$. A ocorrência dessa penalização significa que o indivíduo em questão não cobre todas as atividades presentes em Log .

Embora os operadores lógicos não sejam considerados diretamente no cálculo de LCP , uma vez que as matrizes B/A não os levam em consideração explicitamente, eles são indiretamente importantes para o cálculo. No processo de geração do log artificial a partir do indivíduo (matriz causal), os operadores lógicos existentes no indivíduo são levados em consideração na construção de Log_{Ind} . Por exemplo, quando um operador xOR é encontrado durante a construção de um traço, uma decisão aleatória precisa ser tomada, resultando na geração de diferentes traços de eventos. Essas decisões irão refletir diretamente no conteúdo da matriz B/A gerada para o Log_{Ind} , e dessa forma a avaliação da similaridade acaba levando em consideração a lógica por trás do indivíduo sendo avaliado, o que não ocorre durante o cálculo de TP .

O objetivo de LCP é contribuir para a evolução de um indivíduo que seja capaz de gerar um log (Log_{Ind}) contendo todas as transições da matriz B/A que descreve o log de referência (Log), apresentando assim similaridade total entre ambas as matrizes B/A de Log_{Ind} e Log . A função *fitness* completa é definida na Equação 3, que é uma função de maximização limitada a 1. Nessa equação, α e β são pesos para as funções LCP e TP , os quais são definidos para balancear a influência das duas funções na avaliação.

$$Fitness = (\alpha * TP) + (\beta * LCP) \text{ com } \alpha + \beta = 1 \quad (3)$$

4.3 Operadores genéticos

Esta seção descreve os operadores genéticos implementados para nossa abordagem, conforme colocado a seguir:

4.3.1 Crossover

Dois operadores de *crossover* foram implementados: *crossover* de um ponto e *crossover* uniforme (DE JONG, 2006). No *crossover* de um ponto, um ponto de corte é definido na estrutura do cromossomo, dividindo o índice das atividades em dois grupos. Conforme o exemplo na Tabela 6, um dos filhos resultantes dessa operação receberia o conteúdo das atividades *A1*, *A2* e *A3* do primeiro pai, e o conteúdo das atividades *A4*, *A5*, *A6*, *A7*, *A8* e *A9* do segundo pai. Para o outro filho, o processo oposto é realizado.

Tabela 6 – Exemplo de *crossover* de um ponto

Atividade	Input	Output
A1	{}, {}	{AND}, {A2}
A2	{xOR}, {A1}	{AND}, {A3, A4}
A3	{AND}, {A2}	{AND}, {A5}
A4	{xOR}, {A2}	{AND}, {A5}
A5	{AND}, {A3, A4}	{AND}, {A6}
A6	{AND}, {A5}	{xOR}, {A7, A8}
A7	{AND}, {A6}	{AND}, {A9}
A8	{AND}, {A6}	{AND}, {A9}
A9	{xOR}, {A7, A8}	{}, {}
Início		{AND}, {A1}
Fim	{AND}, {A9}	

Fonte: Gabriel Lucas Cantanhede da Silva, 2018

No *crossover* uniforme, são estabelecidos vários pontos de corte na estrutura do cromossomo, formando uma espécie de ‘máscara’ que guiará a recombinação dos cromossomos. Seguindo o exemplo da Tabela 7, um dos filhos resultantes dessa operação receberia o conteúdo das atividades *A1*, *A2*, *A5*, *A6* e *A9* do primeiro pai e o conteúdo de *A3*, *A4*, *A7* e *A8* do segundo pai, enquanto que para o outro filho gerado o processo oposto é realizado. As linhas *Início* e *Fim* participam apenas indiretamente do *crossover*, tanto no de um ponto quanto uniforme, uma vez que as atividades de início e fim derivam do conteúdo das outras atividades no cromossomo.

4.3.2 Correção do cromossomo

Ao decorrer das operações de *crossover*, cromossomos não factíveis são gerados uma vez que as transições de atividades são representadas em mais de um lugar no cromossomo. Por exemplo, na Tabela 3, na saída da atividade *A2* a atividade *A3* aparece, e por sua vez

Tabela 7 – Exemplo de *crossover* uniforme

Atividade	Input	Output
A1	{}, {}	{AND}, {A2}
A2	{xOR}, {A1}	{AND}, {A3, A4}
A3	{AND}, {A2}	{AND}, {A5}
A4	{xOR}, {A2}	{AND}, {A5}
A5	{AND}, {A3, A4}	{AND}, {A6}
A6	{AND}, {A5}	{xOR}, {A7, A8}
A7	{AND}, {A6}	{AND}, {A9}
A8	{AND}, {A6}	{AND}, {A9}
A9	{xOR}, {A7, A8}	{}, {}
Início		{AND}, {A1}
Fim	{AND}, {A9}	

Fonte: Gabriel Lucas Cantanhede da Silva, 2018

na entrada da atividade *A3* também aparece a atividade *A2*. Essa representação dupla entre as atividades *A2* e *A3* caracteriza a transição $A2 \rightarrow A3$

As operações de *crossover* acabam por quebrar essa representação múltipla das transições ao recombinar as estruturas no cromossomo, a qual precisa ser reconstituída para preservar a factibilidade dos indivíduos. Para isso, a operação de correção do cromossomo ocorre após o crossover, em duas etapas: na primeira etapa, representada pela Equação 4, apaga-se todas as antigas informações de pré-requisitos (conjuntos de *input*) das atividades do indivíduo, preservando apenas os operadores lógicos; na segunda etapa, representada pelas Equações 5 e 6, as informações de pré-requisitos são preenchidas tendo como base as novas informações de pós-requisitos (conjuntos de *output*), reconstruindo as transições.

Uma vez que essa correção opera apenas nas transições, se os conjuntos de *input* possuírem um operador lógico, este é preservado. Caso um desses conjuntos não possua operador lógico, um operador é sorteado e associado ao conjunto de *input*. Por fim, as informações de início e fim do modelo de processo são atualizadas. Caso o indivíduo não possua um início ou fim, atividades devem ser sorteadas para cumprir essa função, e em seguida os conjuntos de *input* ou *output* das atividades sorteadas são ajustados para manter a factibilidade do indivíduo. Para as três equações, *Ind* é o indivíduo sendo corrigido; *k* e *i* são o índice das atividades no cromossomo, *n* é o total de atividades no índice de atividades; *I* e *O* representam os conjuntos de *input* e *output* e *Activ* é cada transição de atividades nos respectivos conjuntos.

$$\forall k, k = 1..n + 1, \{Ind[k, I]\} \leftarrow \emptyset \quad (4)$$

$$\begin{aligned} \forall k, k = 1..n, \exists i \in \{1..n\} \mid \{Ind[k, O]\} \supset Activ_i \\ \wedge \{Ind[i, I]\} \leftarrow \{Ind[i, I]\} \cup \{Activ_k\} \end{aligned} \quad (5)$$

$$\begin{aligned} \exists i, k \in \{1..n\} \mid \{Ind[i, I]\} = \emptyset \wedge \{Ind[k, O]\} = \emptyset \\ \wedge \{Ind[Begin, I]\} \leftarrow Activ_i \wedge \{Ind[End, O]\} \leftarrow Activ_k \end{aligned} \quad (6)$$

4.3.3 Mutação

Dois operadores de mutação foram implementados, mutação simples (DE JONG, 2006) e uma variação de mutação dirigida (BHANDARI; PAL, 1994). A mutação simples pode ocorrer de duas formas: nas transições de pré-condição ou pós-condição de uma atividade, ou nos respectivos operadores lógicos. Na primeira forma, escolhe-se um dos conjuntos de *input* ou *output* da atividade sorteada para a mutação, e então é adicionada ou retirada uma das atividades ao conjunto escolhido. Durante essa forma de mutação simples, é necessário atualizar as informações da transição modificada, para que não se perca a factibilidade do cromossomo. Na segunda forma, o operador lógico do conjunto escolhido é modificado de *AND* para *xOR* e vice-versa. Na mutação dirigida, toma-se uma amostra dos melhores indivíduos de uma população é feita então a comparação entre os cromossomos desses indivíduos para determinar que genes todos eles têm em comum. Todos os genes em comum na estrutura de todos os indivíduos são então modificados por meio da mutação simples.

4.3.4 Fluxo evolutivo

Para guiar a seleção de indivíduos para a aplicação dos operadores genéticos, o operador clássico de seleção por roleta viciada foi implementado, conforme proposto por DE JONG (2006).

Para a troca de população durante o processo evolutivo, foram implementadas duas estratégias de troca: Na primeira estratégia, os pais selecionados para as operações de *crossover* são passados para a próxima geração de indivíduos junto com os filhos gerados. Na segunda estratégia, apenas os filhos gerados são passados para a próxima geração de indivíduos.

Foi implementado também um mecanismo de elitismo, onde o melhor indivíduo de uma geração tem um percentual de chance de ser preservado, sendo passado diretamente para a próxima geração de indivíduos.

5 Experimentos e Resultados

Este capítulo visa apresentar as configurações e resultados das experimentações realizadas com a abordagem proposta de GA para a mineração de processos não estruturados, comparando os resultados com a principal abordagem genética da literatura. Este capítulo apresenta o conjunto de dados a ser usado; apresenta os procedimentos de pré-processamento que foram aplicados ao conjunto de dados do EVC; apresenta os experimentos executados e a estratégia de parametrização adotada na experimentação realizada sobre o conjunto de dados, e apresenta os melhores resultados dos experimentos.

5.1 *Conjunto de dados*

Esta seção apresenta o conjunto de dados utilizado nos experimentos realizados. O conjunto de logs do EVC é o principal conjunto de dados que foi usado nos experimentos preliminares realizados para analisar o problema da descoberta de processos não estruturados com a mineração de processos de negócio.

5.1.1 *Logs de um ambiente virtual de aprendizado*

O conjunto de dados usado no problema da mineração de processos não estruturados é referente a um ambiente virtual de aprendizado, desenvolvido para apoiar o processo de ensino e aprendizagem de um curso semipresencial, em nível de pós-graduação lato sensu (na modalidade especialização), chamado “Ética, Valores e Cidadania na Escola” (EVC), desenvolvido pela Universidade Virtual do Estado de São Paulo (Univesp) com o apoio da ferramenta Moodle. O conjunto de dados é composto de registros de eventos referentes à navegação realizada pelos alunos nos cursos de especialização no sistema de ensino à distância. O sistema armazena nativamente as interações dos usuários com os objetos de aprendizagem do sistema, cujas informações se encontram nos *logs* extraídos do sistema.

Durante toda a extensão dos cursos, aproximadamente 1.500 usuários (dentre os quais se encontram alunos, tutores e coordenadores pedagógicos) usaram o sistema na plataforma Moodle. Para o desempenho dos experimentos práticos de mineração de processos nesse trabalho, foram considerados 291 alunos matriculados na terceira oferta

do curso, realizada no intervalo de 23/08/2013 a 09/12/2014. As informações relativas à interação dos usuários com o sistema durante esse período compõem cerca de 340.000 registros em *log* de eventos. A estrutura dos registros do *log* de eventos armazenados pela plataforma Moodle para o curso EVC é descrita na tabela 8, e um trecho do arquivo de *log* de eventos é apresentado na tabela 9.

Tabela 8 – Dados gerais sobre o conjunto de dados do curso EVC

Rótulo do campo	Descrição	Tipo de dado
Curso	Nomenclatura técnica usada pelos desenvolvedores do sistema	Texto
Data/hora	Momento em que o usuário executa cada ação no sistema	Data/hora
Endereço IP	Endereço na internet da máquina de acesso do usuário	Máscara IP
Nome completo	Nome completo do usuário logado no curso EVC	Texto
Ação	Identificação da ação do usuário na navegação no curso EVC. Trata-se do acesso a um objeto de aprendizagem no sistema	Texto
Informação	Informação adicional sobre a ação realizada pelo usuário	Texto

Fonte: Gabriel Lucas Cantanhede da Silva, 2018

Tabela 9 – Trecho do *log* de eventos do EVC

Curso	Data/hora	Endereço IP	Nome completo	Ação	Informação
PP	11/02/2014 15:07	123.456.78.999	Aluno X	url view	M1M2 RIBEIRÃO PRETO I TURMA A
PP	12/02/2014 15:07	123.456.78.999	Aluno X	course view	Projeto de pesquisa
EVC	13/02/2014 15:07	123.456.78.999	Aluno X	course view	Ética, Valores e Cidadania na Escola
REP2	14/02/2014 15:07	123.456.78.999	Aluno X	page view	Quadro de Notas
EVC	15/02/2014 15:07	123.456.78.999	Aluno X	course view	Ética, Valores e Cidadania na Escola
PP	16/02/2014 15:07	123.456.78.999	Aluno X	url view	M1M2 RIBEIRÃO PRETO I TURMA A
PP	17/02/2014 15:07	123.456.78.999	Aluno X	course view	Projeto de pesquisa
REP2	18/02/2014 15:07	123.456.78.999	Aluno X	page view	Quadro de Notas
EVC	19/02/2014 15:07	123.456.78.999	Aluno X	course view	Ética, Valores e Cidadania na Escola
REP2	20/02/2014 15:07	123.456.78.999	Aluno X	page view	Quadro de Notas
EVC	21/02/2014 15:07	987.654.32.111	Aluno X	course view	Ética, Valores e Cidadania na Escola
PP	22/02/2014 15:07	987.654.32.111	Aluno X	url view	M1M2 RIBEIRÃO PRETO I TURMA A
PP	23/02/2014 15:07	987.654.32.111	Aluno X	course view	Projeto de pesquisa
REP2	24/02/2014 15:07	987.654.32.111	Aluno X	page view	Quadro de Notas
EVC	25/02/2014 15:07	987.654.32.111	Aluno X	course view	Ética, Valores e Cidadania na Escola
REP2	26/02/2014 15:07	987.654.32.111	Aluno X	url view	Texto: Lugar de Vivência: a cidade e a aprendizagem
REP2	27/02/2014 15:07	987.654.32.111	Aluno X	resource view	EC Texto Educação em valores éticos
REP2	28/02/2014 15:07	987.654.32.111	Aluno X	url view	EC Texto A Escola Frente às novas demandas Sociais
REP2	01/03/2014 15:07	987.654.32.111	Aluno X	url view	EC Texto A Escola Frente às novas demandas Sociais
REP2	02/03/2014 15:07	987.654.32.111	Aluno X	url view	mod2sem2texto2
REP2	03/03/2014 15:07	987.654.32.111	Aluno X	url view	mod2sem2texto1

Fonte: Gabriel Lucas Cantanhede da Silva, 2018

5.1.2 Pré-processamento

Os *logs* de eventos originais extraídos do sistema apresentam problemas que dificultam a execução da mineração de processos. Por meio da análise dos registros, os seguintes problemas foram encontrados:

- **Inconsistência:** nomes de alunos e de atividades não padronizados;
- **Incompletude:** ausência de valores para alguns atributos;
- **Ruído:** *logs* relativos à navegação realizada pelos desenvolvedores do sistema na fase de testes do desenvolvimento;
- **Erros de usuário:** erros de autenticação de sessão.
- **Dados de tempo:** as informações de data/hora encontram-se em formato textual.

Tais detalhes foram tratados manualmente com a correção de valores ou exclusão de registros. Após essa limpeza, o conjunto de dados foi reduzido a 254.161 registros de *log* de eventos e, a partir deles, os registros foram transformados em uma nova organização, a qual se adequa melhor à mineração de processos. A nova estrutura do conjunto de dados obtida após o pré-processamento pode ser observada na tabela 10, e um exemplo do novo *log* de eventos pode ser visto na tabela 11

Tabela 10 – Dados gerais do novo conjunto de dados do curso EVC

Rótulo do campo	Descrição	Tipo de dado
Data/hora	Momento em que o usuário executa cada ação no sistema	Data/hora
Usuário	Identificação do usuário que executa a ação	Texto
Atividade	Descrição da ação executada pelo aluno no curso EVC	Texto

Fonte: Gabriel Lucas Cantanhede da Silva, 2018

Tabela 11 – Trecho do *log* de eventos pré-processado do EVC

Data/hora	Usuario	Atividade
11/02/2014 15:07	Aluno X	Projeto de pesquisa
12/02/2014 15:07	Aluno X	Portifolio
13/02/2014 15:07	Aluno X	Profissao docente e educacao especial/inclusiva
14/02/2014 15:07	Aluno X	Profissao docente e educacao especial/inclusiva
15/02/2014 15:07	Aluno X	Profissao docente e educacao especial/inclusiva
16/02/2014 15:07	Aluno X	Portifolio
17/02/2014 15:07	Aluno X	Profissao docente e educacao especial/inclusiva
18/02/2014 15:07	Aluno X	Legenda videoaula
19/02/2014 15:07	Aluno X	Legenda videoaula
20/02/2014 15:07	Aluno X	Profissao docente
21/02/2014 15:07	Aluno X	Profissao docente e educacao especial/inclusiva
22/02/2014 15:07	Aluno X	Educacao especial/inclusiva
23/02/2014 15:07	Aluno X	Legenda videoaula
24/02/2014 15:07	Aluno X	Legenda videoaula
25/02/2014 15:07	Aluno X	Profissao docente

Fonte: Gabriel Lucas Cantanhede da Silva, 2018

Para facilitar nossos experimentos, o log foi reorganizado de forma que as instâncias do processo fossem separadas umas das outras. Dessa forma, cada instância teve seus eventos colocados na ordem em que foram executados, formando os traços de eventos conforme o exemplo na Tabela 12. Além disso, o nome das informações no campo *Ação* no log foi simplificado, facilitando a expressão das respectivas atividades no cromossomo do GA.

Tabela 12 – Amostra do log de eventos do EVC após a reorganização

Id	Traço de eventos
1	⟨ E, D, D, D, C, D, A ⟩
2	⟨ E, D, D, D, C, D, D, D, D, C, D, D, D, D, C, D, D, D, D, D, C, D, D, D, D, C, D, A ⟩
3	⟨ E, M, P, P, P, D, D, C, D, C, D, A ⟩
4	⟨ E, D, D, D, B, B, B, B, B, B, A ⟩
5	⟨ E, D, D, D, C, D, D, A ⟩
6	⟨ E, D, D, D, C, D, D, D, D, D, D, D, B, A ⟩

Fonte: Gabriel Lucas Cantanhede da Silva, 2018

Devido ao volume massivo de informações e extrema quantidade de eventos presentes nos logs do EVC, foi realizada uma amostragem do mesmo visando avaliar primeiramente a capacidade da abordagem proposta na mineração de processos não estruturados com uma amostra menor do log. Isso foi necessário devido a constatação de que o tempo de processamento para as tentativas de minerar o log inteiro tornam a mineração do log

inteiro extremamente custosa computacionalmente, o que inviabilizaria o aprimoramento da abordagem. Além disso, a principal abordagem de genéticos da literatura também enfrentou dificuldades em minerar o log inteiro, não conseguindo até mesmo iniciar o processo evolutivo.

Dessa forma, foi criado um subconjunto do log contendo os eventos de interação de apenas um dos alunos do curso, referente ao primeiro módulo cursado pelo aluno (equivalente a dois meses do curso), contendo 1,377 eventos de interação. Desse subconjunto, foram feitas três amostragens aleatórias, com 10, 20 e 35 traços de eventos, contendo respectivamente 217, 405 e 616 eventos cada. Estas três amostragens do log real constituem o log de referência utilizado nos nossos experimentos, representados pela variável *Log* na função *fitness* apresentada na Seção 4.2. As três amostras do log foram utilizadas para validar a nossa abordagem, bem como para avaliar e comparar os resultados da abordagem do plugin *Evolutionary Tree Miner* (ETM) da ferramenta ProM.

5.2 Configuração dos experimentos e resultados

Esta seção apresenta a configuração de parametrização adotada nos experimentos com a abordagem proposta de GA e a principal ferramenta de mineração de processos baseada em genéticos, ETM-ProM. Para a ferramenta ETM-ProM, foi adotada a configuração padrão indicada pelos autores da ferramenta: número de gerações = 1.000 por execução; elitismo = 5 indivíduos por geração; pesos para a função fitness = 10 para completude, 5 para precisão, 1 para generalização e 1 para simplicidade; taxa de crossover = 0,25; criação aleatória de árvores = 0,25; adição, remoção e mutação de nós = 1; remoção de nós desnecessários = 0,1. A única modificação feita na configuração indicada foi no tamanho da população de indivíduos (cujo valor indicado é 20) para 100, de forma a se equiparar com a quantidade de indivíduos adotada na configuração da nossa abordagem. Dessa forma, ambas soluções dispõem da mesma quantidade de recursos para explorar a superfície de soluções possíveis.

Para a nossa abordagem de GA, adotou-se uma série de valores para os parâmetros configuráveis, visando avaliar o comportamento de cada mecanismo implementado bem como a combinação entre esses. A análise da parametrização é uma etapa fundamental de um algoritmo genético, que tem por objetivo definir a melhor combinação de parâmetros

e valores para o problema em questão. Os experimentos com a nossa abordagem foram executados levando em consideração as seguintes configurações: tamanho da população = 100 por geração; número de gerações = 1.000 por execução; $\alpha = 0.5$; $\beta = 0.5$; operadores de *crossover* = de um ponto e uniforme; taxa de *crossover* = 0,7 e 0,95; taxa da mutação simples = 0,3 e 0,05; taxa da mutação dirigida = 0,3 e 0,05; operador de seleção = roleta viciada; troca de população = só os filhos passam adiante e pais mais filhos passam adiante; taxa de elitismo = 0,5 e 1,0. Levando esses parâmetros em consideração, 32 combinações possíveis de parametrização foram criadas para serem executadas, variando as opções de valores e operadores onde mais de um valor/operador era aplicável.

Os experimentos de descoberta de modelos na mineração de processos não estruturados foram divididos em dois segmentos: (i) primeiramente, aplicando a ferramenta ETM-ProM na mineração das três amostras de log do EVC; e (ii) depois, por meio da análise da parametrização da nossa abordagem proposta de GA para as três amostras de log, apresentando o melhor resultado e parametrização para cada amostra.

5.2.1 Ferramenta *ETM-ProM*

Esse experimento visou usar a principal ferramenta de mineração de processos disponível para a descoberta de processos, aplicando-a no conjunto de logs do EVC. O objetivo desse experimento foi mostrar se de fato a principal abordagem evolutiva atual pode minerar um modelo de processo que dê suporte à análise de processos não estruturados como o do EVC.

A ferramenta ProM foi escolhida porque se trata de uma plataforma aberta para o desenvolvimento de algoritmos de mineração de processos, a qual centraliza vários grupos de pesquisa acadêmicos que colaboram com a proposição de diversos plug-ins de mineração. Por ser voltada para o público acadêmico e ter o código aberto para desenvolvimento, a ferramenta é uma das mais amplamente utilizadas na literatura para os diferentes tipos de mineração de processos. Para a mineração na ferramenta ProM, foi escolhido o plugin que implementa algoritmos genéticos *Evolutionary Tree Miner* (ETM).

As três amostras do log foram mineradas com o ETM-ProM, no entanto apenas o resultado para o log com 10 traços está sendo apresentado, devido as grandes proporções dos modelos de processo minerados para os logs de 20 e 35 traços, além da baixa qualidade

visual dos mesmos. Pode-se notar que no modelo minerado pelo ETM-ProM apresentado na figura 7 há abundância de atividades repetidas no modelo, a exemplo da atividade *Course View* no início do modelo. Avalia-se que essa abundância de atividades repetidas é favorecida pelo ETM-ProM para tentar reproduzir o comportamento complexo presente no log, mesmo com uma amostragem de log pequena com 10 traços. Além disso, o modelo apresenta estruturas lógicas e fluxos de sequência complexos, que tentam criar um grande número de possíveis transições entre atividades. Avalia-se que o grande número de loops presente no log (indicado pela constante repetição de atividades como a *Course View* nos traços) agrava o resultado da mineração do ETM-ProM, visto que a estrutura de cromossomo abordada não permite a criação de loops complexos no modelo de processo. Apesar de permitir a expressão de estruturas de loop no cromossomo, a dificuldade da estrutura de dados adotada pelo ETM-ProM em lidar com loops entre regiões diferentes do processo resulta que o modelo tem de incluir várias tarefas repetidas e estruturas complexas para representar o comportamento do log de forma adequada. Outro problema apresentado pelos modelos minerados pelo ETM-ProM é a presença de grandes sequências de operadores lógicos desnecessários, que não alteram o comportamento mas que poluem ainda mais visualmente o modelo de processo minerado. Todas essas características também foram observadas para os modelos minerados a partir dos logs com 20 e 35 traces. Além disso, observa-se um aumento na complexidade do modelo minerado conforme aumenta-se a quantidade de traços no log utilizado, resultando em modelos com várias vezes a extensão do modelo apresentado na figura 7.

Visto que a ferramenta permite o banceamento das quatro métricas de avaliação dos processos, dentre elas a simplicidade, um outro experimento foi realizado com a amostra de log com 10 traços. Nesse experimento, aumentou-se o peso da variável simplicidade de 1 para 5 (o mesmo peso permaneceu para os outros critérios) visando avaliar se a abordagem consegue minerar um modelo de processo sem as estruturas complexas observadas na Figura 7. O resultado dessa nova parametrização pode ser observado na Figura 8. Se por um lado o modelo de processo da Figura 7 possui excesso de comportamento e tarefas repetidas, ele cobre todas as atividades presentes na amostra. O modelo na Figura 8 possui também tarefas repetidas e é visualmente mais simples, no entanto ele não cobre todas as atividades presentes na amostra.

Figura 7 – Saída do plugin ETM-ProM para a amostra de 10 traços de eventos

Fonte: Gabriel Lucas Cantanhede da Silva, 2018

Figura 8 – Saída do plugin ETM-ProM para a amostra de 10 traços de eventos, com simplicidade = 5

Fonte: Gabriel Lucas Cantanhede da Silva, 2018

5.2.2 Abordagem proposta de GA

Este experimento utilizou a abordagem proposta neste trabalho para a mineração das amostras do log EVC. O experimento teve dois objetivos: *(i)* testar as combinações de parâmetros de forma a avaliar o desempenho dos operadores genéticos desenvolvidos na abordagem, descobrindo as melhores parametrizações para o GA; *(ii)* Apresentar o modelo de processo descoberto pela melhor configuração de parâmetros para cada amostra, analisando as características dos modelos minerados.

(i) Testes de parametrização

Para avaliar as combinações de parâmetros da abordagem proposta, foram executadas 32 combinações de parâmetros utilizando a amostra do log EVC com 10 traços de eventos, com 5 execuções para cada combinação de parâmetros. Essa avaliação inicial visou detectar as melhores combinações que seriam avaliadas para as amostras de 20 e 35 traços de eventos. A lista de configurações dos parâmetros pode ser observada na Tabela 13. Como pode-se observar na Tabela 14, 14 das 32 configurações chegaram ao fitness 1 ao menos uma vez, no entanto pode-se ver que os operadores tem um desempenho geral bom, uma vez que apenas 4 das configurações obtiveram fitness abaixo de 0,8 (configurações 12, 4, 28 e 6).

Para as execuções utilizando os logs com 20 e 35 traços de eventos, as configurações 1, 15, 7, 8, 18 e 20 foram escolhidas. Os resultados podem ser observados nas Tabelas 15 e 16, respectivamente, para os logs de 20 e 35 traços. Na Tabela 15, pode-se observar que todas as configurações alcançaram média acima de 0,8, sendo que apenas uma das configurações chegou ao menos uma vez ao fitness 1. Isso se dá devido ao aumento na complexidade da mineração conforme aumenta-se a quantidade de traços no log. Para todas as configurações, o desvio padrão ficou abaixo de 0,08, o que indica que as cinco execuções para cada uma dessas configurações estão próximas da média, não apresentando grandes oscilações no resultado final do fitness. Na Tabela 16, pode-se observar que apenas a configuração 18 ficou abaixo de 0,8 na média, sendo também a configuração com maior oscilação, como indicado pelo desvio padrão de 0,11.

Tabela 13 – Lista de configurações dos parâmetros

Id.	Indivíduos	Gerações	α	β	Crossover	% Mut. simples	% Cross.	Troca de pop.	Elitismo	% Mut. dirigida
1	100	1000	0.5	0.5	1pt.	5	95	só filhos	1	5
15	100	1000	0.5	0.5	1pt.	30	70	pais e filhos	1	5
7	100	1000	0.5	0.5	uniforme	5	95	só filhos	1	5
8	100	1000	0.5	0.5	uniforme	30	70	pais e filhos	1	30
18	100	1000	0.5	0.5	1pt.	5	95	só filhos	1	30
20	100	1000	0.5	0.5	uniforme	5	95	só filhos	1	30
24	100	1000	0.5	0.5	1pt.	5	95	só filhos	0.5	5
5	100	1000	0.5	0.5	1pt.	5	95	só filhos	0.5	30
9	100	1000	0.5	0.5	uniforme	30	70	pais e filhos	1	5
27	100	1000	0.5	0.5	uniforme	5	95	só filhos	0.5	5
26	100	1000	0.5	0.5	uniforme	30	70	só filhos	1	5
19	100	1000	0.5	0.5	1pt.	30	70	pais e filhos	1	30
30	100	1000	0.5	0.5	1pt.	30	70	só filhos	1	5
21	100	1000	0.5	0.5	uniforme	5	95	só filhos	0.5	30
16	100	1000	0.5	0.5	uniforme	5	95	pais e filhos	1	5
10	100	1000	0.5	0.5	uniforme	30	70	só filhos	1	30
23	100	1000	0.5	0.5	1pt.	30	70	só filhos	1	30
31	100	1000	0.5	0.5	uniforme	5	95	pais e filhos	0.5	30
14	100	1000	0.5	0.5	uniforme	5	95	pais e filhos	0.5	5
0	100	1000	0.5	0.5	1pt.	30	70	pais e filhos	0.5	5
22	100	1000	0.5	0.5	1pt.	30	70	pais e filhos	0.5	30
25	100	1000	0.5	0.5	uniforme	30	70	pais e filhos	0.5	30
3	100	1000	0.5	0.5	1pt.	5	95	pais e filhos	1	5
13	100	1000	0.5	0.5	1pt.	5	95	pais e filhos	1	30
29	100	1000	0.5	0.5	1pt.	5	95	pais e filhos	0.5	5
11	100	1000	0.5	0.5	uniforme	30	70	pais e filhos	0.5	5
2	100	1000	0.5	0.5	uniforme	5	95	pais e filhos	1	30
17	100	1000	0.5	0.5	1pt.	5	95	pais e filhos	0.5	5
12	100	1000	0.5	0.5	1pt.	30	70	só filhos	0.5	5
4	100	1000	0.5	0.5	1pt.	30	70	só filhos	0.5	30
28	100	1000	0.5	0.5	uniforme	30	70	só filhos	0.5	30
6	100	1000	0.5	0.5	uniforme	30	70	só filhos	0.5	5

Fonte: Gabriel Lucas Cantanhede da Silva, 2018

Tabela 14 – Resultados das execuções para a amostra com 10 traços de eventos

Id.	Fitness máximo	Mínimo	Média	Desv. Padrão
1	1	0.966666667	0.989333333	0.013727507
15	1	0.964166667	0.981	0.015858752
7	1	0.953809524	0.978857143	0.018414723
8	1	0.950952381	0.976952381	0.017038306
18	1	0.95026455	0.976505291	0.017411356
20	1	0.94047619	0.975357143	0.02537854
24	0.990833333	0.941211052	0.973527925	0.017902113
5	0.994642857	0.934166667	0.97352381	0.021428519
9	1	0.943809524	0.970952381	0.019293649
27	1	0.922751323	0.970478836	0.026721187
26	0.994642857	0.947619048	0.968047619	0.017622361
19	1	0.916507937	0.960420635	0.031309977
30	1	0.920634921	0.960238095	0.028731211
21	0.968333333	0.918636096	0.952822457	0.018564787
16	0.98	0.860199882	0.945635215	0.044248315
10	0.97	0.901352146	0.944539389	0.025421853
23	0.970833333	0.863078704	0.935862948	0.038284859
31	0.9575	0.894212963	0.934496326	0.022159331
14	0.970833333	0.841666667	0.925661376	0.044047525
0	1	0.892298648	0.921985303	0.039968414
22	0.964166667	0.868371546	0.919109347	0.032753555
25	0.931666667	0.900529101	0.915698413	0.012445574
3	1	0.799166667	0.907634921	0.069583655
13	0.930511464	0.866388889	0.897656085	0.023794107
29	1	0.688580247	0.896265726	0.108100987
11	0.954166667	0.820833333	0.893518519	0.045036976
2	0.923927102	0.698333333	0.86194027	0.083871349
17	1	0.692016461	0.85658517	0.113316289
12	0.785952381	0.642345679	0.700869195	0.052973789
4	0.733303939	0.657256236	0.692979256	0.025979508
28	0.687277778	0.614	0.647916259	0.032138062
6	0.674960317	0.599265138	0.629009168	0.026641268

Fonte: Gabriel Lucas Cantanhede da Silva, 2018

(ii) Apresentação dos modelos minerados

Os modelos a serem representados foram escolhidos conforme os resultados das execuções apresentadas nas Tabelas 14, 15 e 16, considerando as configurações que atingiram maior média de fitness para as 5 execuções realizadas. Para o log com 10 e 35 traços, a configuração 1 foi escolhida, enquanto que para o log com 20 traços, a configuração 7 foi a escolhida. Ambas configurações apresentam parâmetros quase idênticos, mudando

Tabela 15 – Resultados das melhores configurações para o log com 20 traços.

Id	Fitness máximo	Mínimo	Média	Desv. Padrão
7	1	0.817156085	0.944334055	0.066177165
18	0.967063492	0.831726354	0.913193034	0.050151328
20	0.934186016	0.889000722	0.908320128	0.018725968
1	0.961349206	0.840464165	0.904943723	0.038452548
15	0.959438541	0.848351371	0.897493299	0.03747747
8	0.950277778	0.765126263	0.856498159	0.074728466

Fonte: Gabriel Lucas Cantanhede da Silva, 2018

Tabela 16 – Resultados das melhores configurações para o log com 35 traços.

Id.	Fitness máximo	Mínimo	Média	Desv. Padrão
1	0.960634921	0.826013431	0.911289565	0.04781381
15	0.947222222	0.81460928	0.88764479	0.043364418
20	0.911045365	0.743197358	0.83001709	0.05572589
7	0.91468254	0.779615385	0.8207109	0.049237332
8	0.858083028	0.728181387	0.809957294	0.044089244
18	0.920872507	0.616745014	0.745277582	0.113032964

Fonte: Gabriel Lucas Cantanhede da Silva, 2018

apenas o operador de crossover escolhido. Os modelos de processo minerados podem ser observados na Figura 9.

Nos experimentos com nossa abordagem, pode-se observar que uma estrutura que permita a mineração de loops entre diferentes regiões do processo resulta em uma simplificação considerada do modelo final minerado, indicando que a estrutura de matriz causal é mais adequada para a mineração de um log não estruturado. A complexidade de cada modelo minerado aumenta conforme o número de traços no log cresce, uma vez que naturalmente esse incluirá mais informações de eventos e transições entre atividades. Nos modelos minerados, foram identificados operadores lógicos de sequência, disjunção exclusiva (*xOR*) e estruturas de laço entre atividades, porém não foi identificado nenhum operador de paralelismo (*AND*) nos logs. Essa ausência de operadores *AND* já era esperada, uma vez que o sistema do EVC impõe que os alunos executem as suas atividades sequencialmente.

Apesar da diferença de complexidade entre os três modelos de processo, algumas similaridades podem ser observadas entre eles: A primeira atividade associada ao conteúdo do curso é sempre *course view*, sendo a principal atividade que centraliza os modelos; *message write* está sempre diretamente associada com *course view*; a atividade *page view* é sempre seguida por *resource view*; *resource view* e *blog view* sempre possuem um laço de repetição para si mesmas. Algumas estruturas estão presentes apenas alguns dos modelos,

como o laço de repetição entre as atividades *assignment view* e *assignment upload*, presente apenas nos modelos da Figura 9(b) e 9(c).

Figura 9 – Modelos de processo minerados para os logs com 10, 20 e 35 traços, respectivamente (modelados em BPMN na ferramenta Signavio Academic).

Fonte: Gabriel Lucas Cantanhede da Silva, 2018

O gráfico de evolução da melhor configuração de parâmetros para os logs de 10, 20 e 35 traços é apresentado na Figura 10, levando em consideração a melhor das cinco execuções para a melhor configuração em cada um dos logs. Para cada uma das três execuções escolhidas, o gráfico mostra a progressão do melhor, pior e a média de *fitness* ao decorrer das gerações. Assim como nos modelos de processo minerados, pode-se observar na Figura 10 o aumento na complexidade da tarefa de mineração, necessitando mais gerações para alcançar bons resultados. Para o log com 10 traços, o *fitness* 1 foi atingido na 427ª geração. Para o log com 20 traços, *fitness* 1 foi atingido na 869ª geração. Já para o log com 35 traços, o limite de 1000 gerações foi atingido, terminando com *fitness* igual a 0,961. Para todas as linhas de evolução foi detectada uma tendência de crescimento.

Figura 10 – Evolução do *fitness* para o log com 10 traços em laranja, 20 traços em azul e 35 traços em verde.

Fonte: Gabriel Lucas Cantanhede da Silva, 2018

Como forma de validar a qualidade dos modelos minerados, a função *fitness* para completude proposta por Medeiros, Weijters e Aalst (2006) foi aplicada nos indivíduos gerados pela nossa abordagem. Essa fórmula leva em consideração os traços de eventos como um todo, em contraste à nossa abordagem que leva em consideração as transições entre atividades presentes nos traços de eventos no log. Para os modelos de processo mostrados na Figura 9, essa fórmula de completude resultou nos valores 1, 1 e -1,873,

respectivamente. O valor dessa fórmula para o último modelo de processo o grande grau de oscilação que essa fórmula proporciona, uma vez que um indivíduo que pela nossa fórmula está bem próximo de 1 é avaliado negativamente pela mesma.

6 Conclusão

A meta deste trabalho de pesquisa de mestrado foi contribuir para o crescente desenvolvimento da área de mineração de processos, apresentando e validando uma solução para a descoberta automática de modelos de processos não estruturados. A apresentação dessa abordagem tem por fim dar apoio ao trabalho de analistas de processos, permitindo identificar comportamentos de interesse no contexto de processos complexos. Tal solução teve como base a técnica de algoritmos genéticos para a descoberta de padrões no processo, o que requisitou estudo e experimentação aprofundados das representações cromossômicas de processos adotadas na literatura, da função *fitness* e suas métricas e dos operadores genéticos aplicáveis ao mesmo.

Para fundamentar as escolhas necessárias à proposição da abordagem de GA apresentada neste trabalho, foi necessária a realização de uma revisão da literatura na área. Essa revisão foi um aprofundamento feito a partir de um mapeamento da área de mineração de processos, realizado pelo grupo de pesquisa em BPM da EACH-USP do qual o autor faz parte, que apontou a necessidade de estudo e aprimoramento de várias técnicas de inteligência computacional aplicadas no contexto da mineração de processos, dentre elas os algoritmos genéticos.

O objetivo dessa revisão foi obter uma visão das abordagens de algoritmos genéticos já existentes na literatura, com o objetivo de orientar a proposição de uma nova solução para a descoberta de modelos de processo não estruturados, de forma a identificar pontos falhos a serem evitados e conseqüentemente contribuir para o amadurecimento da área com a proposição de novas estratégias e combinações de operadores. Por meio da mesma pôde-se observar que o uso de algoritmos genéticos no contexto de aplicação de processos não estruturados ainda não foi devidamente explorado. Além disso, foi possível identificar nas abordagens da literatura o uso de apenas operadores genéticos simples, o que foi considerada uma lacuna a ser trabalhada na nossa abordagem proposta.

Na elaboração da proposta de GA apresentada neste trabalho, foram levados em considerações os pontos positivos e negativos detectados principalmente nos artigos avaliados na revisão de literatura. A representação cromossômica adotada na nossa proposta foi a de matriz causal, adaptada de um dos trabalhos da literatura, que foi escolhida pelo seu maior poder de expressão de estruturas lógicas complexas, segundo a literatura. Foram

adicionadas algumas modificações à representação escolhida, que se fizeram necessárias devido à falta de expressividade da representação original quanto ao início e fim do modelo de processo, o que é fundamental em um problema caracterizado pela falta de estrutura.

Novas fórmulas para completude e precisão foram criadas para a abordagem proposta neste trabalho. Essas fórmulas são baseadas no conceito de transições entre atividades, o qual não foi explorado ainda na literatura relativa a algoritmos genéticos. Isso se fez necessário visto que as principais fórmulas utilizadas na literatura se mostraram insuficientes no contexto de processos não estruturados. A principal fórmula para completude na literatura, baseada na execução de traços de eventos, não consegue apresentar uma divisão satisfatória entre resultados ruins e bons dentro desse contexto, permitindo que indivíduos ruins atinjam bons valores de fitness ao passo que penaliza possíveis indivíduos bons. Além disso, a principal fórmula de precisão proposta na literatura requer que se possua o modelo do processo a priori, e a fórmula de precisão mais recentemente proposta se aplica apenas a representações cromossômicas de árvores de processo. Para possibilitar os cálculos das fórmulas propostas neste trabalho, foi necessária a criação de uma estrutura auxiliar ao log, nomeada matriz B/A. Essa estrutura se mostrou fundamental, uma vez que a consulta direta ao log durante os cálculos é um dos principais gargalos identificados nas abordagens da literatura, aumentando o tempo de processamento para cada execução do GA.

A abordagem proposta neste trabalho teve como principal objetivo apresentar uma solução para a mineração de processos não estruturados, em detrimento às abordagens da literatura que em sua maioria são validadas utilizando logs de processos estruturados. A nossa abordagem mostrou-se eficaz na descoberta de modelos de processos não estruturados se comparado com outra abordagem evolutiva na literatura, no entanto conclusões mais assertivas em relação à comparação com essa abordagem da literatura não podem ser feitas uma vez que tanto a sua representação cromossômica quanto os seus operadores e configurações possuem características diferentes das do nosso trabalho, não podendo ser diretamente comparadas. A abordagem deste trabalho também se aplica à mineração de processos estruturados, o que demonstra a capacidade da técnica de algoritmos genéticos na mineração de processos.

A nossa abordagem contribui para a aplicação de algoritmos genéticos na mineração de processos em quatro aspectos: Propõe-se uma nova versão da matriz causal utilizada na literatura; propõe-se um novo operador de reparação da matriz causal; estabelece-se uma nova fórmula de fitness que pode ser utilizada também em outras representações

de cromossomo, ajudada pela estrutura auxiliar de matriz B/A proposta; e implementa-se operadores genéticos clássicos que ainda não haviam sido explorados na literatura – crossover uniforme, mutação direcionada e estratégias de troca de população.

O GA proposto não leva em consideração os critérios da generalização e simplicidade, apesar de a utilização desses critérios ainda não é consenso na literatura referente a GA. No entanto, considerando apenas as métricas de completude e precisão, os resultados dos experimentos indicam que as métricas de generalização e simplicidade podem ter sido alcançadas indiretamente na nossa abordagem. Porém, análises mais extensivas precisam ser realizadas levando também em consideração amostras maiores de *log*.

As duas hipóteses levantadas para este trabalho foram provadas verdadeiras: O GA proposto neste trabalho se mostrou capaz de obter melhores resultados que a outra abordagem genética comparada, mesmo com o alto grau de ruído e falta de estrutura presentes nos logs; o uso de operadores genéticos mais específicos para o problema se mostraram essenciais para a obtenção de resultados melhores na mineração de processos não estruturados, colocando em uso estratégias e operações que levaram à mineração de modelos de processo melhores.

Nossa proposta de GA não teve como objetivo propor uma solução para o problema do tempo de execução do GA, o que é considerada uma oportunidade de trabalho futuro. Conforme o tamanho do log aumenta, os recursos computacionais e o tempo de execução crescem numa tendência exponencial, sendo necessário estudar formas de lidar com esse gargalo computacional. Outras oportunidades de trabalhos futuros são a implementação e avaliação de novos operadores genéticos, bem como a proposição de fórmulas e estratégias para considerar diretamente as métricas de generalização e simplicidade na abordagem.

Referências¹

- AALST, V. M. P. van der. *Process Mining – Discovery, Conformance and Enhancement of business processes*. [S.l.]: Springer-Verlag, 2011. Citado 2 vezes nas páginas 18 e 57.
- AALST, W. M. P.; MEDEIROS, A. K. A. de; WEIJTERS, A. J. M. M. Genetic process mining. In: *Proc. of the 26th Int. Conf. on Applications and Theory of Petri Nets (ICATPN)*. [S.l.]: Springer, 2005. p. 48–69. Citado 6 vezes nas páginas 60, 61, 62, 63, 65 e 67.
- AALST, W. M. P. van der. The application of petri nets to workflow management. *Journal of Circuits, Systems, and Computers*, v. 8, n. 1, p. 21–66, 1998. Citado 2 vezes nas páginas 28 e 29.
- AALST, W. M. P. van der. Business process management as the "killer app" for petri nets. *Software and System Modeling*, v. 14, n. 2, p. 685–691, 2015. Citado 2 vezes nas páginas 26 e 29.
- AALST, W. M. P. van der. *Process Mining: Data Science in Action*. 2. ed. Heidelberg: Springer, 2016. Citado 9 vezes nas páginas 20, 43, 44, 45, 46, 47, 48, 56 e 58.
- AALST, W. M. P. van der; GUNTHER, C. W. Finding structure in unstructured processes: The case for process mining. In: BASTEN, T.; JUHÁŠ, G.; SHUKLA, S. K. (Ed.). *Application of Concurrency to System Design, 2007. ACSD 2007. Seventh International Conference on*. [S.l.]: IEEE Computer Society, 2007. p. 3–12. Citado 3 vezes nas páginas 18, 19 e 21.
- AALST, W. M. P. van der; WEIJTERS, A. J. M. M. Process mining: A research agenda. *Comput. Ind.*, Elsevier Science Publishers B. V., v. 53, n. 3, p. 231–244, abr. 2004. ISSN 0166-3615. Citado na página 49.
- AALST, W. van der; BUIJS, J.; DONGEN, B. van. Towards improving the representational bias of process mining. In: _____. *Data-Driven Process Discovery and Analysis: First International Symposium, SIMPDA 2011, Campione d'Italia, Italy, June 29 – July 1, 2011, Revised Selected Papers*. [S.l.]: Springer Berlin Heidelberg, 2012. p. 39–54. Citado 2 vezes nas páginas 60 e 62.
- ABONYI, J.; FEIL, B.; ABRAHAM, A. Computational intelligence in data mining. *Informatica*, Slovene Society Informatika, v. 29, n. 1, p. 3–12, 5 2005. Citado na página 38.
- ANGELINE, P. J. Advances in genetic programming. In: ANGELINE, P. J.; KINNEAR JR., K. E. (Ed.). [S.l.]: MIT Press, 1996. cap. Genetic Programming's Continued Evolution, p. 1–20. Citado na página 42.
- ASHLOCK, D. A. *Evolutionary Computation for Modeling and Optimization*. [S.l.]: Springer, 2006. Citado 4 vezes nas páginas 22, 39, 40 e 44.
- BHANDARI, D.; PAL, N. R. Directed mutation in genetic algorithms. *Information Sciences*, Elsevier, v. 79, n. 3-4, p. 251–270, 1994. Citado na página 75.

¹ De acordo com a Associação Brasileira de Normas Técnicas. NBR 6023.

BIOLCHINI, J. et al. *Systematic Review in Software Engineering*. [S.l.], 2005. Citado na página 54.

BRATOSIN, C.; SIDOROVA, N.; AALST, W. M. P. van der. Distributed genetic process mining. In: *Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2010, Barcelona, Spain, 18-23 July 2010*. [S.l.: s.n.], 2010. p. 1–8. Citado na página 66.

BRATOSIN, C.; SIDOROVA, N.; AALST, W. M. P. van der. Distributed genetic process mining using sampling. In: *Parallel Computing Technologies - 11th International Conference, PaCT 2011, Kazan, Russia, September 19-23, 2011. Proceedings*. [S.l.: s.n.], 2011. p. 224–237. Citado 2 vezes nas páginas 58 e 66.

BRATOSIN, C.; SIDOROVA, N.; AALST, W. van der. Discovering process models with genetic algorithms using sampling. In: _____. *Knowledge-Based and Intelligent Information and Engineering Systems: 14th International Conference, KES 2010, Cardiff, UK, September 8-10, 2010, Proceedings, Part I*. [S.l.]: Springer Berlin Heidelberg, 2010. p. 41–50. Citado 2 vezes nas páginas 58 e 66.

BROUCKE, S. V.; VANTHIENEN, J.; BAESENS, B. Declarative process discovery with evolutionary computing. In: . [S.l.: s.n.], 2014. p. 2412–2419. Citado 3 vezes nas páginas 58, 59 e 63.

BUIJS, J. C. A. M.; DONGEN, B. F. van; AALST, W. M. P. van der. A genetic algorithm for discovering process trees. In: *Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2012, Brisbane, Australia, June 10-15, 2012*. [S.l.: s.n.], 2012. p. 1–8. Citado 4 vezes nas páginas 59, 60, 62 e 65.

CAMILO C. O.; SILVA, J. C. d. *Mineração de dados: Conceitos, Tarefas, Métodos e Ferramentas*. [S.l.], 2009. Relatório técnico. Citado na página 30.

CHENG, H. J.; OU-YANG, C.; JUAN, Y. C. A hybrid evolutionary-based process mining technology to discover parallelism structures. In: *2012 IEEE International Conference on Industrial Engineering and Engineering Management*. [S.l.: s.n.], 2012. p. 1573–1577. Citado 3 vezes nas páginas 60, 61 e 62.

CHENG, H.-J.; OU-YANG, C.; JUAN, Y.-C. A hybrid approach to extract business process models with high fitness and precision. *Journal of Industrial and Production Engineering*, v. 32, n. 6, p. 351–359, 2015. Citado 2 vezes nas páginas 61 e 62.

CHENG, H.-J.; OU-YANG, C.; JUAN, Y.-C. An integrated mining approach to discover business process models with parallel structures: Towards fitness improvement. *International Journal of Production Research*, v. 53, n. 13, p. 3888–3916, 2015. Citado 2 vezes nas páginas 61 e 62.

DE JONG, K. A. *Evolutionary Computation: A Unified Approach*. [S.l.]: MIT press, 2006. 250 p. Citado 4 vezes nas páginas 18, 21, 73 e 75.

DIAMANTINI, C.; GENGA, L.; POTENA, D. Behavioral process mining for unstructured processes. *J. Intell. Inf. Syst.*, Kluwer Academic Publishers, v. 47, n. 1, p. 5–32, ago. 2016. ISSN 0925-9902. Citado 2 vezes nas páginas 19 e 21.

DUMAS, M. et al. *Fundamentals of Business Process Management*. [S.l.]: Springer, 2013. Citado na página 26.

EBERHART, R.; SHI, Y. *Computational Intelligence: Concepts to Implementations*. [S.l.]: Elsevier, 2007. Citado 2 vezes nas páginas 34 e 35.

ECK, M. van; BUIJS, J.; DONGEN, B. van. Genetic process mining: Alignment-based process model mutation. *Lecture Notes in Business Information Processing*, v. 202, p. 291–303, 2015. Citado 2 vezes nas páginas 59 e 62.

ESGIN, E.; SENKUL, P.; CIMENBICER, C. A hybrid approach for process mining: Using from-to chart arranged by genetic algorithms. In: _____. *Hybrid Artificial Intelligence Systems: 5th International Conference, HAIS 2010, San Sebastián, Spain, June 23-25, 2010. Proceedings, Part I*. [S.l.]: Springer Berlin Heidelberg, 2010. p. 178–186. Citado 2 vezes nas páginas 63 e 64.

FENG, M.; OUYANG, J.; LI, J. A heuristic genetic process mining algorithm. *2011 Ninth International Conference on Computational Intelligence and Security*, IEEE Computer Society, v. 00, p. 15–19, 2011. Citado na página 62.

GRZYMALA-BUSSE, J. W. Three approaches to missing attribute values: A rough set perspective. In: _____. *Data Mining: Foundations and Practice*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008. p. 139–152. Citado na página 30.

HAN, J.; KAMBER, M.; PEI, J. *Data mining concepts and techniques, third edition*. Waltham, Mass.: Morgan Kaufmann Publishers, 2012. Citado 4 vezes nas páginas 17, 29, 55 e 57.

HAN J.; KAMBER, M. *Data Mining – Concepts and Techniques*. 2nd. ed. [S.l.]: Morgan Kaufmann, 2006. Citado na página 30.

JIAN, C.; LI, F. An improved virus evolutionary genetic algorithm for workflow mining. *Journal of Theoretical and Applied Information Technology*, v. 47, n. 1, p. 406–411, 2013. Citado 3 vezes nas páginas 59, 60 e 63.

KANTARDZIC, M. *Data Mining: Concepts, Models, Methods, and Algorithms*. [S.l.]: John Wiley & Sons, 2011. Citado 3 vezes nas páginas 30, 34 e 35.

KHEZERLOU, A. V.; ALIZADEH, S. A new model for discovering process trees from event logs. *Applied Intelligence*, v. 41, n. 3, p. 725–735, 2014. Citado 3 vezes nas páginas 59, 61 e 62.

KITCHENHAM, B. *Guidelines for Performing Systematic Literature Reviews*. [S.l.], 2007. Citado 4 vezes nas páginas 54, 55, 56 e 103.

LEE, C. et al. A slippery genetic algorithm-based process mining system for achieving better quality assurance in the garment industry. *Expert Systems with Applications*, v. 46, p. 236–248, 2016. Citado 8 vezes nas páginas 57, 58, 59, 60, 61, 62, 63 e 64.

MADEO, R. C. B. et al. A committee machine implementing the pattern recognition module for fingerspelling applications. In: *Proceedings of the 2010 ACM Symposium on Applied Computing*. [S.l.]: ACM, 2010. (SAC '10), p. 954–958. Citado na página 30.

MAITA, A. R. C. *Um estudo da aplicação de técnicas de inteligência computacional e de aprendizado em máquina de mineração de processos de negócio*. Dissertação (Mestrado) — Escola de Artes, Ciências e Humanidades, Universidade de São Paulo, São Paulo, 2015. Citado 5 vezes nas páginas 17, 20, 21, 43 e 50.

MEDEIROS, A. K. A. de; WEIJTERS, A. J. M. M.; AALST, W. M. P. Genetic process mining: A basic approach and its challenges. In: *Proc. of the 1st Works. on Bus. Proc. Intelligence (BPI)*. [S.l.: s.n.], 2006. p. 203–215. Citado 4 vezes nas páginas 60, 65, 67 e 92.

MEDEIROS, A. K. A. de; WEIJTERS, A. J. M. M.; AALST, W. M. P. Genetic process mining: An experimental evaluation. *Data Mining and Knowledge Discovery*, Springer, v. 14, n. 2, p. 245–304, 2007. Citado 3 vezes nas páginas 59, 63 e 65.

MOLKA, T. et al. Diversity guided evolutionary mining of hierarchical process models. In: . [S.l.: s.n.], 2015. p. 1247–1254. Citado 4 vezes nas páginas 59, 60, 62 e 63.

MOLKA, T. et al. Evolutionary computation based discovery of hierarchical business process models. *Lecture Notes in Business Information Processing*, v. 208, p. 191–204, 2015. Citado 4 vezes nas páginas 59, 60, 62 e 63.

REISIG, W. *Understanding Petri Nets: Modeling Techniques, Analysis Methods, Case Studies*. [S.l.]: Springer Publishing Company, Incorporated, 2013. Citado 2 vezes nas páginas 26 e 27.

SELEEM, Y. Z.; MOHAMED, M. H.; HUSSAIN, K. F. Improving genetic process mining using honey bee algorithm. In: *Informatics and Applications (ICIA), 2013 Second International Conference on*. [S.l.: s.n.], 2013. p. 59–65. Citado 3 vezes nas páginas 59, 61 e 64.

SIVANANDAM, S. N.; DEEPA, S. N. *Introduction to Genetic Algorithms*. 1st. ed. [S.l.]: Springer-Verlag, 2008. Citado 8 vezes nas páginas 17, 18, 22, 38, 40, 41, 42 e 43.

TAMPELINI, L. G. et al. Uma aplicação de redes de elman no tratamento e predição de séries temporais hidrológicas. *Anais do IX Congresso Brasileiro de Redes Neurais e Inteligência Computacional*, Elsevier Science Publishers B. V., Ouro Preto, p. 231–244, 2009. Citado na página 30.

THOMÉ, A. C. G. *Redes neurais: uma ferramenta para kdd e data mining*. [S.l.], 2002. Material didático. Citado na página 30.

TSAI, C.-Y.; CHEN, I.-C. Using genetic process mining technology to construct a time-interval process model. In: *Proceedings of the 22Nd International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems: Next-Generation Applied Intelligence*. [S.l.]: Springer-Verlag, 2009. (IEA/AIE '09), p. 113–122. Citado na página 58.

TSAI, C.-Y.; JEN, H.; CHEN, Y.-C. Time-interval process model discovery and validation—agenetic process mining approach. *Applied Intelligence*, v. 33, n. 1, p. 54–66, 2010. Citado na página 58.

TURNER, C. J. et al. Process mining: from theory to practice. *Business Proc. Manag. Journal*, v. 18, n. 3, p. 493–512, 2012. Citado 4 vezes nas páginas 20, 21, 50 e 51.

VÁZQUEZ-BARREIROS, B.; MUCIENTES, M.; LAMA, M. A genetic algorithm for process discovery guided by completeness, precision and simplicity. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, v. 8659 LNCS, p. 118–133, 2014. Citado 3 vezes nas páginas 61, 62 e 66.

WANG, L.; FU, X. *Data Mining with Computational Intelligence*. [S.l.]: Springer Berlin Heidelberg, 2006. Citado na página 30.

WESKE, M. *Business Process Management: Concepts, Languages, Architectures*. 2nd. ed. [S.l.]: Springer-Verlag, 2012. Citado 2 vezes nas páginas 17 e 26.

YANG, L. et al. An effective process mining approach against diverse logs based on case classification. In: . [S.l.: s.n.], 2015. p. 351–358. Citado 5 vezes nas páginas 58, 59, 60, 61 e 62.

ZHAO, W. et al. Mining the role-oriented process models based on genetic algorithm. In: _____. *Advances in Swarm Intelligence: Third International Conference, ICSI 2012, Shenzhen, China, June 17-20, 2012 Proceedings, Part I*. [S.l.]: Springer Berlin Heidelberg, 2012. p. 398–405. Citado na página 64.

ZHAO, W.; LIU, X.; DAI, W. Simplified process model discovery based on role-oriented genetic mining. *The Scientific World Journal*, v. 2014, 2014. Citado na página 64.

Apêndices

Apêndice A – Protocolo da revisão de literatura

O protocolo é onde são especificados os métodos que serão utilizados durante a realização de uma RSL (KITCHENHAM, 2007). Os componentes principais a serem especificados no protocolo de uma RSL são: Fontes de dados; estratégia de busca; estratégia de seleção dos estudos primários; método de extração; e resumo dos dados.

A.1 Fontes de dados e estratégia de busca

Como fontes de dados para esta RSL, foram escolhidos dois mecanismos de busca de artigos: Scopus e Web of Science (WoS). Essas ferramentas são mecanismos de busca populares no meio acadêmico, uma vez que indexam as principais bases digitais de artigos acadêmicos. A estratégia de busca foi construída a partir das questões de pesquisa previamente estabelecidas. Foram escolhidas palavras-chave que refletissem esses questionamentos, sendo essas palavras conectadas por meio de operadores lógicos AND e OR, de forma a abranger o máximo de combinações de palavras que fizessem sentido para essa pesquisa, visando ampliar a quantidade de estudos relacionados retornados. A primeira metade da string de busca reflete os termos de interesse relacionados à área de mineração de processos, já a última parte limita os estudos retornados a apenas aqueles que usam as técnicas de computação evolucionária de interesse para esta RSL. As palavras-chave utilizadas na string foram baseadas no conhecimento de especialistas das áreas de BPM e inteligência computacional, os quais são participantes deste grupo de pesquisa. Uma vez que as duas bases de dados utilizadas implementam sintaxes particulares para a realização de buscas avançadas nas mesmas, foi necessário adaptar a string de busca visando refletir essas sintaxes. Os mecanismos de busca utilizados permitem aplicar vários tipos de filtros nos resultados da busca, no entanto, esses filtros foram aplicados manualmente. Essa decisão foi tomada visando eliminar ao máximo as chances de excluir estudos erroneamente indexados nas bases, o que ocorre frequentemente com artigos de uma das bases indexadas.

A.1.1 Strings de busca - Scopus:

TITLE-ABS-KEY(("process mining"OR "processes mining"OR "workflow mining"OR "workflows mining"OR "mining of business processes"OR "mining of processes"OR "mining of workflows"OR (("business process"OR "business processes"OR "workflow"OR "workflows") AND "mining")) AND (genetic* OR evolution* OR evolutive*))

A.1.2 Strings de busca - WoS:

TS=(("process mining"OR "processes mining"OR "workflow mining"OR "workflows mining"OR "mining of business processes"OR "mining of processes"OR "mining of workflows"OR (("business process"OR "business processes"OR "workflow"OR "workflows") AND "mining")) AND (genetic* OR evolution* OR evolutive*)) A versão final das strings de busca fora aplicada pela primeira vez em abril e novamente em maio de 2016.

A.2 *Estratégia de seleção dos estudos primários*

Foram estabelecidos critérios de inclusão e exclusão para esta RSL, os quais visam garantir que os estudos primários selecionados estão dentro do contexto de interesse desta revisão, ou seja, estão situados nas áreas de mineração de processos e computação evolucionária. Um estudo é tido como primário se o mesmo satisfaz todos os critérios de inclusão, bem como não se encaixa em qualquer um dos critérios de exclusão. Desta forma, foram aplicados nessa RSL os seguintes critérios de inclusão/exclusão:

A.2.1 Critérios de inclusão

- **CI-01** O artigo aborda diretamente algum aspecto de data mining, ou seja, alguma tarefa de mineração é executada.
- **CI-02** Processos de negócio ou workflows são parte essencial do escopo do artigo, significando que elementos associados a essas áreas são tratados de forma significativa.
- **CI-03** Data mining e processos de negócios/workflows são abordados conjuntamente, de forma a tratar de algum dos tipos de mineração de processos: descoberta, confor-

midade ou melhoria de processos. No entanto, será considerada também qualquer tarefa, direta ou indireta, de mineração no contexto de processos de negócio ou de workflows.

- **CI-04** O artigo emprega, na tarefa de mineração, uma das seguintes técnicas de computação evolutiva: Algoritmos genéticos, estratégias evolutivas ou programação genética.

A.2.2 Critérios de exclusão

- **CE-01** O artigo não está disponível eletronicamente.
- **CE-02** O artigo não está escrito em inglês.
- **CE-03** O artigo é relacionado primordialmente a outras áreas de pesquisa, e não a sistemas de informação ou ciência da computação.
- **CE-04** O registro retornado pela string de busca não é um artigo científico revisado por pares (a exemplo de publicações como relatórios técnicos, livros, capítulos de livro, prefácios de anais ou editoriais de periódicos).
- **CE-05** O artigo é um estudo secundário.
- **CE-06** O artigo usa de forma isolada as seguintes técnicas de computação evolutiva: particle swarm, sistemas classificadores, ant colony, bee colony, ou mesmo computação evolutiva em geral, sem deixar claro qual tipo de técnica foi utilizada.
- **CE-07** O artigo não apresenta a modelagem do problema, ou seja, não mostra detalhes de como os conceitos das técnicas evolutivas foram aplicados ao problema (estrutura do GA, operadores implementados, modelagem do cromossomo, estratégias adotadas, etc.)

A.3 *Condução da revisão sistemática de literatura*

Esta sessão contém a forma como esta RSL fora conduzida, sendo estruturada em duas seções: Identificação e seleção dos estudos primários; e extração e síntese dos dados.

A.3.1 Identificação e seleção dos estudos primários

A identificação e seleção dos estudos primários fora feita em duas etapas, conforme a seguir:

- **Etapa 1.** Identificação dos estudos primários nas bases de dados: Para essa etapa, as strings de busca elaboradas para esta revisão foram aplicadas nas duas bases de dados determinadas. A busca retornou 392 registros no Scopus e 116 no WoS. Cruzando os registros das bases, apenas 19 dos 116 registros do WoS eram inéditos em relação aos do Scopus, totalizando 411 registros.
- **Etapa 2.** Aplicando os critérios de inclusão e exclusão: Nessa etapa, todos os 411 registros retornados pela etapa anterior foram analisados com base nos critérios de inclusão e exclusão elencados. Esses critérios foram aplicados através da leitura superficial dos registros, baseando-se nas informações contidas no título, resumo e nas palavras-chave do registro. Nos casos onde essa análise superficial não permitira excluir ou incluir o registro em questão, foram consideradas informações presentes no arquivo completo do artigo. Como resultado desta etapa, 47 estudos foram classificados como primários.

A.3.2 Extração e síntese dos dados

Nesta etapa, todos os estudos classificados como primários foram objeto de uma análise aprofundada, visando responder às questões de pesquisa previamente elencadas. Durante essa leitura detalhada, alguns estudos foram eliminados por se enquadrarem por não atenderem satisfatoriamente a um ou mais critérios de inclusão, ou mesmo por se encaixarem em algum dos critérios de exclusão. Para auxiliar no controle desse processo, foram utilizados dados exportados nas bases onde fora feita a pesquisa, manipulando essas informações em um software de edição de planilhas.