João José Jóia Gonçalves

Descoberta automática de práticas de trabalho com técnicas de mineração de processos

Universidade do Algarve

Faculdade de Ciências e Tecnologia

2021

João José Jóia Gonçalves

Descoberta automática de práticas de trabalho com técnicas de mineração de processos

Mestrado Integrado em Engenharia Eletrónica e Telecomunicações

Trabalho efetuado sob a orientação de:

Prof.ª Marielba Silva de Zacarias

Universidade do Algarve

Faculdade de Ciências e Tecnologias

2021

Descoberta automática de práticas de trabalho com técnicas de mineração de processos

Declaração de autoria de trabalho

Declaro ser o autor deste trabalho, que é original e inédito. Autores e trabalhos consultados estão devidamente citados no texto e constam da listagem de referências incluída.

_____

AGRADECIMENTOS

Agradeço a todas as pessoas que me acompanharam nesta etapa da minha vida.

A toda a minha família, especialmente aos meus pais que sempre estiveram disponíveis para me ajudar, desde o início da minha caminhada académica.

Obrigado, mãe e pai.

Á minha namorada, o amor da minha vida que me acompanha e esteve de perto desde o início desta etapa, por todo o companheirismo, motivação, ajuda e amor que nunca faltou em algum momento, que ajudou-me imenso em não desistir.

Obrigado, Patrícia.

Á professora doutora Marielba Silva de Zacarias, pela orientação ao longo desta caminha, por todas as sugestões, conhecimento partilhado e desafios propostos que me possibilitam concluir esta dissertação.

Obrigado, professora Marielba.

A todos os meus amigos, em especial ao Bruno Brasil, em que ambos sabemos o quanto foi difícil esta etapa e nos apoiamos um ao outro.

Obrigado, Brasil.

Aos meus sogros, que juntamente com os meus pais, motivaram e ajudaram imenso para a conclusão desta dissertação.

Obrigado, Margarida e Paulo.

A todos vós, agradeço novamente, Obrigado!

RESUMO

A passagem para um mundo cada vez mais digital tem vindo a concretizar-se com o avanço da tecnologia a ocorrer de forma exponencial. Com ela tem crescido a necessidade de se analisar a imensidão de dados que empresas e organizações produzem a fim de identificar padrões, melhorias operacionais e tendências.

A literatura tem-se dedicado ao estudo de processos de negócio e o *Process Mining* tem vindo a ganhar notoriedade como uma disciplina imprescindível ao crescimento de uma organização no que respeita à forma como analisa os seus dados, que passam a ser o *Storyteller* das mesmas. A mineração de processos dá respostas a questões que não ficam resolvidas pelas formas tradicionais de análise de dados, permitindo extrair conhecimento através de registos de eventos e assim descobrir novos modelos de processos, controlar e melhorar os processos já existentes.

O estudo de caso sobre o qual este trabalho é desenvolvido debruça-se sobre o contexto organizacional real, na identificação de práticas de trabalho a partir de registo de ações relacionadas ao desenvolvimento de aplicações de *web* para um banco comercial. Como objetivo de estudo, procurou-se realizar uma análise comparativa de vários algoritmos de *Process Mining* disponíveis na ferramenta *ProM*.

A análise realizada teve por base encontrar o algoritmo mais apropriado às especificações dos dados do caso de estudo, através de medidas de qualidade presentes na verificação de conformidade.

Numa primeira fase, os dados foram extraídos, mapeados e importados para a ferramenta *ProM* e seguidamente aplicaram-se algoritmos de descoberta de processo. O resultado final gerou modelos de processo em rede *Petri Net* que foram analisados através da verificação de conformidade. Por fim, foi possível verificar através dessas análises que dos algoritmos testados, o que obteve melhor resultado foi o *Heuristic Miner*

Palavras-chave: Process Mining, Heuristic Miner, ProM, Petri Net, Event log, Verificação de conformidade

ABSTRACT

A passage to an increasingly digital world has been taking place with the advancement of technology taking place exponentially. With it, the need to analyze the immensity of data that essential companies and associations to identify patterns, operational improvements and trends has grown.

Literature has been dedicated to the study of business processes and Process Mining has been gaining notoriety as an essential discipline for the growth of an organization with regard to the way it analyzes its data, which becomes the Storyteller of the previous ones. Process mining provides answers to questions that are not resolved by traditional forms of data analysis, allowing you to extract knowledge through event records and thus discover new process models, and control existing processes.

The case study on which this work is developed focuses on the real organizational context, in the identification of work practices from the registration of actions related to the development of web applications for a commercial bank. As a study objective, it is expected to perform a comparative analysis of several Process Mining algorithms available in the ProM tool.

The analysis performed was based on finding the most appropriate algorithm to the specifications of the case study data, through quality measures present in the compliance check.

In a first phase, data were extracted, mapped and imported into a ProM tool, then process discovery algorithms were applied. The end result generated process models in Petri Net network that went through the compliance check. Finally, it was possible to verify through these analyzes that of the tested algorithms, the one that obtained the best result was the Heuristic Miner.

Keywords: Process Mining, Heurisic Miner, ProM, Petri Net, Event log, Conformance Checker

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABELAS

LISTA DE ABREVIATURAS

BI – *Business Inteligence*

BPI – *Business Process Intelligence*

BPA – *Business Process Analysis*

BAM – *Business Activity Monitoring*

BPM – *Business Process Management*

WfM – *Workflow Management*

SI – *Sistemas de Informação*

BPMN – *Business Process Modeling Notation*

UML – *Unified Modeling Language*

OMG – *Object Management Group*

EPC – *Event-driven Process Chain*

YAWL – *Yet Another Workflow Language*

MXML – *Magic Extensible Markup Language*

XES – *Extensible Event Stream*

Capítulo 1

INTRODUÇÃO

1.1 CONTEXTO

Segundo (Bobrovski, Grigorova & Malysheva, (2017) *Data Mining* (DM) é uma área multidisciplinar, que surgiu e se desenvolveu com base em campos da ciência como a estatística aplicada, inteligência artificial, reconhecimento de padrões, algoritmos, teoria de banco de dados, entre outros. Refere-se à extração de informações implícitas, anteriormente desconhecidas e potencialmente úteis. É, na verdade, parte do processo da descoberta de conhecimento (Zaïane,1999).

O Business Intelligence (BI) pode ser visto como uma análise ascendente dos dados corporativos para melhorar o processo de tomada de decisão e ajudar os gerentes a entender os seus negócios e o posicionamento em um ambiente competitivo (Baesens, Schupp, Weerdt & Vanderloock, 2012). No início dos anos 90, as ferramentas de BI ganharam popularidade, sendo um processo de transformar dados em informação e depois em conhecimento, frequentemente com recurso às técnicas de DM.

Paralelamente, *Business Process Management* (BPM) tornou-se uma disciplina que pode ser usada para otimizar processos de negócio (Bach, Hernaus & Vuksic, 2012). Um processo de negócio é um conjunto de atividades realizadas de forma coordenada no fornecimento de produtos ou serviços com valor acrescentado aos clientes ou para satisfazer outros objetivos estratégicos (Strnadl, 2006). O BPM é reconhecido como uma abordagem que promove melhorias continuas dos processos de uma organização, influenciando positivamente o desempenho geral da mesma (Bach, Stjepic, Vugec & Vuksic, 2019). Abrange um conjunto de atividades que envolvem (1) identificação, modelação e implementação dos processos, (2) Configuração do sistema de suporte ao BPM (BPMS), (3) execução dos processos modelados e (4) monitorização, análise e melhoria dos processos (Van der Aalst, 2013).

A junção de BI e o BPM é uma abordagem frequente que permite tornar os processos mais flexíveis e oferece uma maior capacidade de análise (Kerremans & Kitson, 2012).

O *Process Mining* é uma disciplina mais recente que interliga técnicas *de DM* e as atividades do BPM, combinando os pontos fortes de ambos. O *Process Mining* inclui algoritmos para a descoberta automática de modelos de processos, a partir de *event logs* gerados pelas aplicações empresariais (Sahingoz & Saylam, 2013).

Os *event logs* podem ser usados para conduzir três tipos de *Process Mining*, sendo eles: (1) Descoberta, que é a produção do modelo; (2) Verificação de Conformidade, que apresenta medidas de qualidade, visando a deteção de inconsistências entre um modelo de processo e o seu correspondente *log* de execução; (3) Aprimoramento, melhora um modelo de processo existente usando informações sobre o processo real registado em algum event log (Adriansyah, Arcieri, Baier & Van der Aalst, 2011).

Existem várias ferramentas vocacionadas para o *Process Mining*, como *Disco* (Günther & Rozinat, 2012) e *ProM* (Günther et al., 2009). Esta última, é uma ferramenta gratuita e desenvolvida pela Universidade de Tecnologia de Eindhoven, a qual permite gerar vários modelos de processos através da presença de vários algoritmos e avaliar a qualidade do modelo gerado (Günther et al., 2009).

1.2 PROBLEMA E OBJETIVO

O PM é uma disciplina promissória na descoberta de práticas de trabalho. As práticas de trabalho foram definidas em (Zacarias, 2008) como "padrões recorrentes de ação e interação entre trabalhadores", onde os padrões são olhados na perspetiva das sequências de ações realizadas por um ou mais indivíduos. Desde esta perspetiva, as práticas de trabalho podem ser tratadas como processos. No entanto, os algoritmos desta disciplina funcionam com registos tipicamente bem estruturados e volumosos, na ordem dos milhares ou dezenas de milhares de registos.

Esta tese visa responder à seguinte questão: é possível descobrir sequências recorrentes de ações a partir de um registo de ações pequeno, utilizando técnicas de PM e com recurso à ferramenta ProM?

Para responder a esta questão, o principal objetivo desta tese é realizar uma análise comparativa de vários algoritmos de *Process Mining* disponíveis na ferramenta *ProM* (ferramenta para *Process Mining*), na identificação de práticas de trabalho a partir de um

registo de ações num contexto organizacional real. A avaliação dos algoritmos analisadas será realizada com base nas medidas de qualidade presentes na verificação de conformidade.

1.3 ESTRUTURA DA TESE

A tese está organizada em cinco capítulos. O capítulo 1 está dividido em três secções onde descreve o contexto, apresenta uma descrição sobre o problema juntamente com o objetivo e terminando com a descrição da estrutura da tese. O capítulo 2 apresenta o estado de arte que está dividido em onze secções onde dá a conhecer *Data Mining*, *Business Intelligence*, *Business Process Management*, *Process Mining*, tipos de *Process Mining*, perspetivas de descoberta, modelos de representação, algoritmos, medidas de qualidade, ferramentas para visualização de resultados de *event logs* e uma análise comparativa de vários algoritmos mencionados. O capítulo 3 apresenta toda a preparação para análise dos resultados. Os resultados e sua discussão são descritos no capítulo 4. Finalmente, o capítulo 5 apresenta as conclusões relevantes, bem como sugestões para trabalhos futuros.

Capítulo 2

ESTADO DA ARTE

2.1 DATA MINING

Nas últimas três décadas, as organizações geraram uma grande quantidade de dados que representaram na forma de arquivos e bancos de dados (Mostafa, 2018). Os sistemas de computador modernos estão cada vez mais acumulando dados a uma taxa quase inimaginável e de uma grande variedade de fontes: transações de cartão de crédito e até satélites de observação da Terra no espaço (Bramer, 2007). É aqui que aparece *Data Mining*, sendo um mecanismo de análise de muitos dados e resumindo-os para descobrir um padrão e conhecimento. O conhecimento é induzido a partir da informação que é extraída dos dados. Portanto, *Data Mining* tem sido usado em várias áreas, como estatística, reconhecimento de padrões e sistemas revolucionários. Tem métodos, modelos, técnicas e algoritmos que podem ser usados para extrair padrões de informação e conhecimento (Mostafa, 2018).

A preparação dos dados é uma etapa essencial na análise, sendo necessário uma coleção de procedimentos, podendo ser visualizado a partir da Figura 2.1 (Abdulwahid & Jassim, 2021):

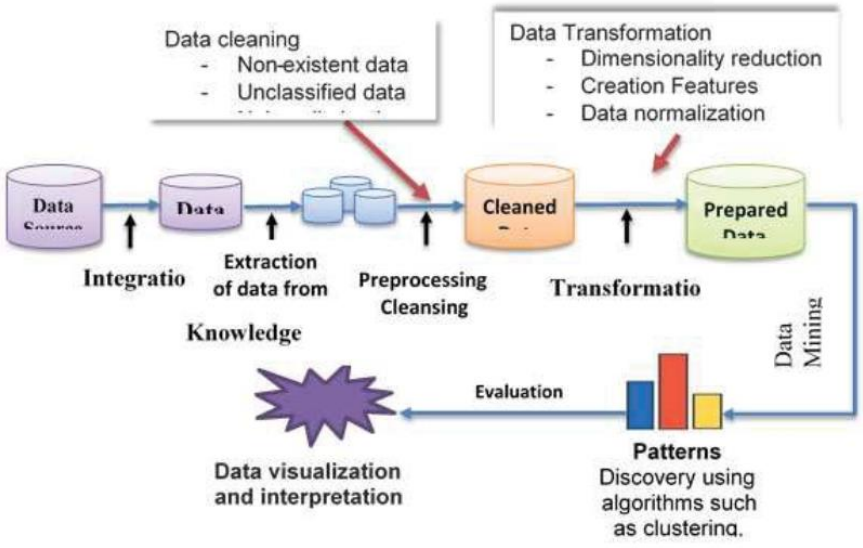


Figura 2.1: Preparação dos dados.

Na prática, os dois objetivos principais de *Data Mining* tendem a ser a previsão e a descrição dos dados. A previsão envolve o uso de algumas variáveis ou campos no conjunto de dados para prever valores desconhecidos ou outras futuras variáveis de interesse. Já a descrição, concentra-se em encontrar padrões que descrevem os dados que podem ser interpretados pelo Homem. Portanto, é possível colocar as atividades de *Data Mining* em uma de duas categorias (Han & Kamber, 2000):

1) **Mineração de dados preditiva**: Produz o modelo do sistema descrito por um determinado conjunto de dados. O modelo é expresso como um código executável, que pode ser usado para realizar classificação, previsão, estimativa ou outras tarefas semelhantes.

2) **Mineração de dados descritiva**: O objetivo é obter uma compreensão do sistema analisado, descobrindo padrões e relacionamentos em grandes conjuntos de dados.

Existem várias técnicas importantes de *Data Mining* que vêm sendo desenvolvidas e utilizadas em projetos deste. Existem várias técnicas inovadoras e intuitivas que ajustam os conceitos de *Data Mining* em uma tentativa de dar às empresas uma visão mais abrangente dos seus próprios dados com úteis tendências futuras. Muitas técnicas são utilizadas pelos especialistas em *Data Mining*, onde são divididas em preditiva e descritiva (Duraisamy & Shankar, 2018).

A partir de (Han & Kamber, 2000; Duraisamy & Shankar, 2018), do lado preditivo temos técnicas mais usadas como:

- **Classificação**: Esta análise é usada para recuperar informações importantes e relevantes sobre dados, ajudando a classificá-los em classes diferentes.
- **Regressão**: Identifica e analisa a relação entre as variáveis. Pode ser possível identificar a probabilidade de uma variável especifica, no meio de outras tantas.
- **Previsão**: Analisa eventos ou instâncias passadas em uma sequência correta para prever um evento futuro.

Já do lado descritivo temos outras técnicas como:

- ***Clustering***: É uma técnica para identificar dados semelhantes. Este processo ajuda a compreender as diferenças e semelhanças entre um conjunto de dados.
- **Regras de associação:** Esta técnica ajuda a encontrar a associação entre dois ou mais tipos de dados. Descobre um padrão oculto no conjunto de dados.
- **Sequencia de padrões**: Esta técnica ajuda a descobrir ou identificar padrões ou tendências semelhantes nos dados por um determinado período.

Ferramentas de software gratuitas (*RapidMiner*, *Weka*, entre outros) e publicamente disponíveis para *Data Mining* estão em desenvolvimento nos últimos 20 anos. O objetivo dessas ferramentas é facilitar o processo bastante complicado de análise de dados e oferecer a todos os pesquisadores interessados uma alternativa gratuita às plataformas comerciais de análise de dados (Jovic, Bogunovic & Brkic, 2015). Com grande quantidade de dados para lidar, podemos aplicar algoritmos de *Data Mining* interessantes (*Clustering*, *Decision Trees*, *Regression*, entre outros) e conseguir visualizações com maior rapidez (Duraisamy & Shankar, 2018; Ashraf & Aslam, 2014).

2.2 BUSINESS INTELLIGENCE

Business Intelligence (BI) designa-se como o processo de transformar dados em informação e posteriormente em conhecimento. As necessidades do cliente, processos de tomada de decisão deste, concorrência, condições no setor e tendências económicas, tecnológicas e culturais em geral, tudo isto abrange o conteúdo do conhecimento. BI foi criado no mundo da indústria no princípio dos anos 90, para responder aos gestores como analisar da melhor maneira e eficientemente os dados da empresa, entendendo melhor a situação dos seus negócios e aperfeiçoar o processo de decisão (Golfarelli & Rizzi, 2004).

O surgimento da inteligência de negócios lançou uma luz sobre as novas dimensões dos dados adquiridos em uma empresa. BI só pode ser adquirido usando *Data Mining* de maneiras diferentes. O uso de armazenamento de dados e sistemas de informação (SI) possibilitou o rápido crescimento dos conjuntos de dados organizacionais. Há uma variedade de técnicas avançadas de processamento de dados que podem ajudar os processos de BI a serem executados com eficiência, oferecidas pelo *Data Mining*. O processo abrangente de aplicação de BI para um problema de negócios é conhecido como processo de Descoberta de Conhecimento em Bancos de Dados (KDD) e é vital para implementações de *Data Mining* bem-sucedidas com BI em mente (Hazra, Kumar, Mishra & Tarannum, 2016).

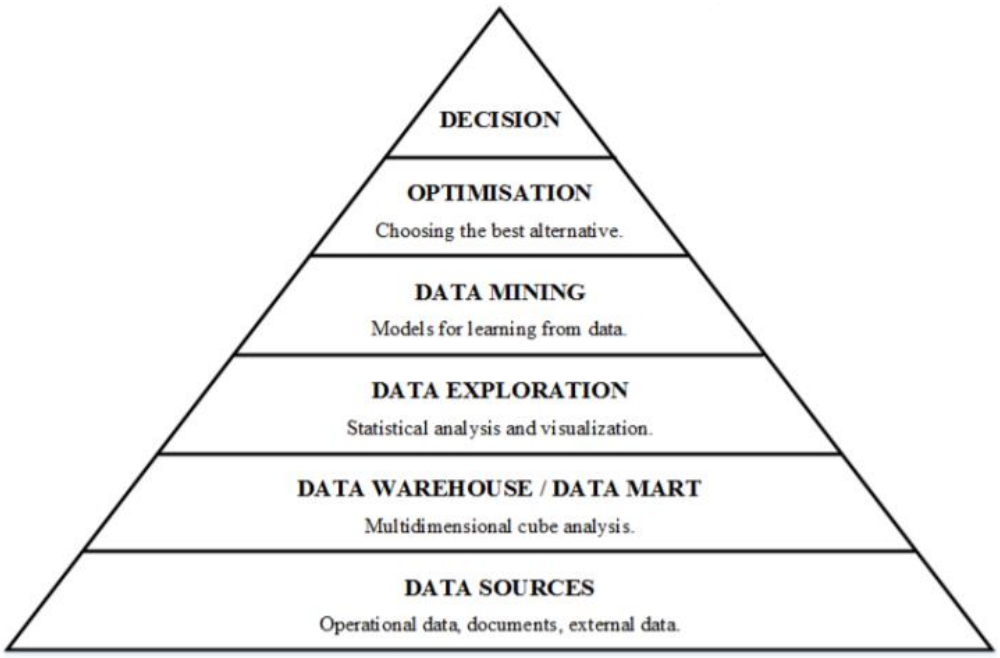


Figura 2.2: Arquitetura de BI (Kopcek, Kopceková & Tanuska, 2013).

Através da Figura 2.2 e segundo (Kopcek et al., 2013; Azeroual & Theel, 2018) é possível visualizar a arquitetura de BI dividida em:

- ***Data sources***: Reunir e integrar todos os dados disponíveis.
- ***Data warehouse***: Banco de dados para extração e transformação.
- ***Data exploration***: Todos os conceitos e ferramentas que se preocupam principalmente com a avaliação e análise dos dados são atribuídos a este processo.
- ***Data Mining***: Fornece ferramentas para a análise ativa de BI, cuja função é extrair informação e conhecimento dos dados, aumentando o conhecimento dos responsáveis de decisão.
- ***Optimisation***: Facilita a seleção da melhor solução a partir de um grande número de alternativas.
- ***Decision***: Seleção e aceitação de decisões específicas, que culminam no processo de tomada de decisão.

Com o aparecimento das técnicas de gestão relacionadas ao processo, *Business Process Intelligence* (BPI) foi criado como subdomínio de BI. Este novo processo designa-se como um conjunto de ferramentas integradas que oferece suporte aos utilizadores de negócios e de tecnologia de informação (TI) na gestão da qualidade de execução de processos, fornecendo vários recursos, como análise, previsão, monitorização, controlo e otimização (Baesens et al., 2012).

2.3 BUSINESS PROCESS MANAGEMENT

O *Business Process Management* (BPM) é muito importante do ponto de vista prático e oferece muitos desafios para criadores e cientistas de *software*. Antigamente o que dominava o cenário dos sistemas de informação eram as abordagens orientadas a dados. Mais recentemente ficou mais óbvio que os processos são igualmente importantes e precisam ser apoiados sempre que possível, resultando num grande aparecimento de sistemas de gestão de *Workflow* em meados dos anos 90 (Weske, 2007). O *Workflow* é um conceito relacionado com a automatização de processos de negócios e informações em uma organização. Consegue descrever tarefas do processo de negócios de modo a entender, avaliar e redesenhar esse mesmo processo. O *Workflow Management* é um *software* que permite automatizar os fluxos de trabalho de uma organização (Georgakopoulos & Hornick, 1995).

O BPM define-se como o suporte a processos de negócio usando métodos, técnicas e *software* para projetar, executar, controlar e analisar processos operacionais envolvendo humanos, organizações, aplicações, documentos e outras fontes de informação (Weske, 2007).

Por meio de modelos, será possível identificar visualmente os problemas, podendo até apontar melhorias antes desconhecidas para otimizar a situação. O mesmo se aplica aos processos de negócios. A modelação dos processos em curso de um negócio, ou mesmo entre os negócios, pode trazer a identificação instantânea de problemas e é uma ferramenta importante para a simulação de eficiências de determinados processos. Alguns dos benefícios da análise e modelação de processos de negócios segundo (Ko, 2009) são os seguintes:

- Maior visibilidade e conhecimento das atividades da empresa;
- Maior capacidade de identificar falhas;
- Maior identificação de áreas potenciais de otimização;
- Prazos de entrega reduzidos;
- Melhor definição de funções na empresa;
- Boa ferramenta para prevenção de enganos, auditoria e avaliação de conformidade com os regulamentos.

Notar que os sistemas de suporte ao BPM precisam estar cientes do processo, ou seja, sem informações sobre os processos operacionais, pouco suporte é possível. A Figura 2.3 mostra a relação entre *Workflow Management* e BPM usando o ciclo de vida BPM. Este ciclo de vida descreve as várias fases de suporte aos processos operacionais de negócios, onde segundo (van der Aalst & Weske, 2004) as fases são:

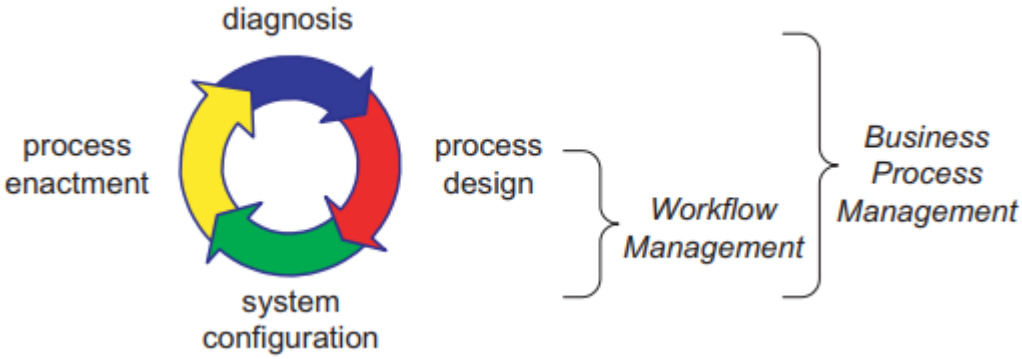


Figura 2.3: Ciclo de vida do BPM.

- ***process design***: os processos são projetados;
- ***system configuration***: os projetos são implementados pela configuração de um sistema de informação do processo (por exemplo, um sistema *Workflow Management*);
- ***process enactment***: a fase de aprovação começa onde os processos operacionais de negócios são executados usando o sistema configurado;
- ***diagnosis***: os processos são analisados para identificar problemas e encontrar possíveis melhorias.

Através deste ciclo é possível mostrar como os processos operacionais de negócios são concebidos, configurados, executados e analisados. No campo de BPM, o diagnóstico de processo a posteriori é amplamente coberto pelo termo *Business Process Analysis* (BPA) (Baesens et al., 2012). BPA caracteriza-se em investigar propriedades de processos de negócios que não são óbvias nem comuns, a fim de esclarecer as características dos processos de negócios, identificar possíveis falhas e comparar alternativas de processos potenciais (Majeed, Tiwari & Vergidis, 2008). Quando o foco muda para a gestão de desempenho de processos em tempo real, o campo de *Business Activity Monitoring* (BAM) aparece (Baesens et al., 2012). BAM fornece informações em tempo real sobre o *status* e os resultados de várias operações, processos e transações comerciais. Tem como

principal objetivo reduzir ou eliminar atrasos e falhas, como também analisar o desempenho de negócio baseado em dados fornecidos em tempo real (Choi, Han, Han & Lee, 2010).

2.4 PROCESS MINING

Na área de *Business Intelligence*, mais recentemente, surge uma nova forma de lidar com o processamento de informação requerido no BPM, denominada *Process Mining*. Considerado como um ramo independente da ciência de dados, aplica técnicas de mineração de dados, a conjuntos de dados gerados por execuções de processos (os chamados *event logs* ou registos de eventos) (Batyuk & Voityshyn, 2018). Usualmente, as empresas fazem uso de documentação e levam a cabo investigações manuais para obter uma visão geral dos seus processos de negócios. Acontece que, abordagens tradicionais têm a capacidade de fornecer informações de que algo não funciona adequadamente, porém, a informação não é precisa o suficiente, porque não mostra todos os fatos que revelam o que exatamente não funciona bem e em que circunstâncias. Outra desvantagem é que a análise manual consome tempo e recursos, o que no mundo ágil significa que os resultados da análise podem perder rapidamente o seu valor devido a mudanças constantes do ambiente de negócios. É precisamente nesse ponto que atua o *Process Mining*, o objetivo da mineração de processos é utilizar *event logs* para extrair informações relacionadas aos processos em análise. Em geral, através da mineração de processos são aplicadas técnicas de mineração de dados sobre os *event logs* para atingir objetivos específicos (Saravanan & Sree, 2010).

Desta forma, o conjunto de ferramentas de mineração de processos fornece *insights* valiosos sobre a situação real. No caso de aplicação de mineração de processos aos principais processos de negócios de uma organização, os *insights* obtidos podem ser usados não só para melhorar os processos controlados, mas são entradas valiosas para a transformação digital de toda a empresa. (Batyuk & Voityshyn, 2018)

A sua crescente integração, fornece meios para aumentar a eficácia e eficiências dos processos e abre novas possibilidades de acesso e análise de dados, se estes forem bem avaliados e processados de forma adequada, a fim de serem tomadas decisões para o crescimento da empresa ou organização. A base lógica em que é assente o *Process Mining* é a de descobrir, controlar e melhorar os processos reais, extraindo conhecimento através

dos dados e gerar *event logs*. As técnicas de *Process Mining* revelam claros benefícios já cientificamente comprovados, uma vez que permitem que a informação seja compilada objetivamente, são úteis porque reúnem informações sobre o que está a acontecer em formato real de acordo com um registo de eventos de uma organização. O algoritmo de *Process Mining* ajuda no processamento dos dados necessários, podendo ser extraídos diferentes tipos de modelos que descrevem os processos implícitos. Um modelo de processo de negócios expõe uma ordem específica de atividades de trabalho com um início, um fim e entradas e saídas definidas. Diversas podem ser as abordagens com base em *Process Mining* porque diferentes algoritmos de *Process Mining* têm as suas próprias especificidades, entre eles são conhecidos na literatura o *Alpha Miner, Heuristic Miner, Genetic Miner, Fuzzy Miner*, entre outros. (Gupta, 2015; Grigorova, Malysheva & Mironov, 2018).

A criação dos modelos para uma maior facilidade de análise de dados é possível com recurso a um dos mais antigos e mais utilizados *frameworks* em investigação, o *ProM*, uma plataforma que tem o grande benefício de dispor de vários *plugins* baseado em Java, com código aberto (process_mining_important_paper) [9].

Segundo (Batyuk & Voityshyn, 2018), o *Process Mining* atua como uma ligação entre campos amplamente conhecidos como BI e BPM.

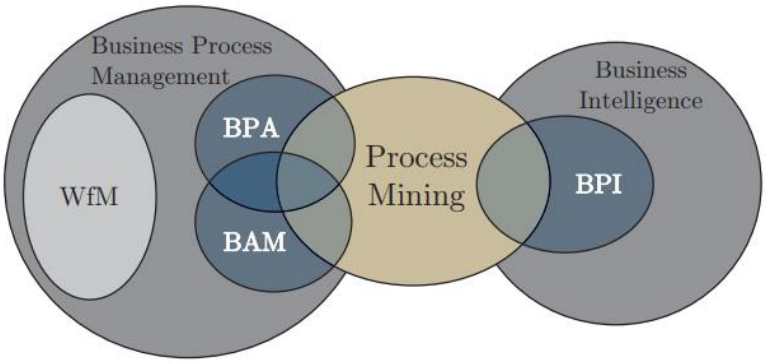


Figura 2.4: *Process Mining* num contexto mais alargado (Baesens et al., 2012).

Atualmente uma organização adquire muita informação durante a sua rotina diária. Nas últimas décadas, os Sistemas de Informação (SI) passaram de sistemas simples com funcionalidade reduzida, para arquiteturas complexas e integradas sendo mais complicado entender e acompanhar como esses sistemas afetam a execução de processos diários nas organizações. Sendo assim o *Process Mining* oferece uma solução baseada na extração, análise, diagnóstico e visualização dos dados registados por um SI durante a execução do processo (Baesens et al., 2012). Esses dados estão contidos em *event logs* que correspondem à execução histórica dos processos de negócios (Garcia & Mayorga, 2015). Um *event log* contém eventos relacionados a um único processo de negócios capturado por um sistema de informações. Ele cobre o que foi feito, quando foi feito, por quem foi feito (também pode ter dados descritivos adicionais), em relação a qual instância do processo, também conhecida como *Case ID* (Jans, Jouck & Soffer, 2019).

Na Figura 2.5, um exemplo de *event log* é abaixo apresentado onde se pode visualizar as instâncias ou casos (*Case ID*) do processo (por exemplo, "1"), as atividades (*Activity)* do processo (por exemplo, "*Create Purchase Order*"), as pessoas (*Resource*) que executam cada atividade (por exemplo, "*Joyce*") e a data e hora (*Timestamp*) de cada atividade (por exemplo, "2018/08/16 10:16").

Case ID	Event ID	Timestamp	Activity	Resource
1	1	2018/08/16 10:16	Create purchase order	Joyce
2	2	2018/08/16 13:07	Create purchase order	Steve
2	3	2018/08/17 12:11	Reject purchase order	Max
1	4	2018/08/17 14:13	1st approval PO	Jim
1	5	2018/08/17 17:25	2nd approval PO	Mike
1	6	2018/08/28 08:47	Enter goods receipt	Dustin
1	7	2018/08/29 09:34	Book invoice	Nancy
1	8	2018/09/16 11:46	Pay invoice	Jonathan

Figura 2.5: Exemplo de um *event log*.

No centro de *Process Mining* e *Data Mining* estão os dados. Ambos têm muito em comum, pois usam os mesmos algoritmos e técnicas matemáticas. A principal diferença é que o *Data Mining* atua com os dados em geral, enquanto o *Process Mining* trabalha com os dados sobre eventos, que incluem informações sobre os processos (Grigorova, Malysheva & Bobrovski, 2017).

Na Tabela 2.1, segundo (Houthoofd, 2015) estão detalhadamente as diferenças entre *Data Mining* e *Process Mining*.

Tabela 2.1 - Diferença detalhada entre *Data Mining* e *Process Mining*.

Data Mining	**Process Mining**
Analisar dados e detetar ou prever padrões	Concentra em descobrir, controlar e melhorar os processos de negócios reais, fornecendo uma visão verdadeira e completa de como os processos de negócios operam
Analisa informações estáticas	Pode fornecer uma visualização em tempo real dos processos de negócios
Procura padrões ocultos nas coleções de dados, mas não responde a perguntas específicas	Permitem procurar especificamente respostas para perguntas claras e predefinidas
Revela certos padrões, mas não responde à pergunta de como esses padrões foram estabelecidos, sendo limitada apenas aos resultados	Pode fornecer informações sobre como os resultados foram alcançados. A técnica não procura padrões nos dados, mas processos causais
Foca nos principais padrões de um conjunto de dados, onde os que se enquadram fora desses padrões comuns, geralmente não são incluídos na análise	As exceções às vezes podem ser tão importantes por ser um indicador precoce de ineficiências ou oportunidades de melhoria

2.5 TIPOS DE PROCESS MINING

Process Mining resolve o problema que a maior parte das organizações possuem como informações insuficientes sobre o que realmente está acontecendo. Exclusivamente uma avaliação precisa da realidade pode ajudar na verificação de modelos de processos e ser usada em esforços de redesenho de sistemas ou processos. Tem como ideia encontrar, monitorizar e melhorar processos reais que não são assumidos, obtendo conhecimento dos *event logs* (Saravanan & Sree, 2010). Com esse conhecimento, três tipos de *Process Mining* podem ser orientados para diferentes propósitos, sendo eles: Descoberta do Processo, Verificação de Conformidade e, por fim, Aprimoramento.

- **Descoberta do Processo:** Este tipo através de um registo de evento, produz um modelo sem usar nenhuma informação a priori (van der Aalst, 2011). Através de um *event log*, a execução dos processos existentes é registada como entrada e produz um modelo de processo de negócios como saída. Com base nas informações disponíveis nos *events logs*, três tipos de perspetivas de descoberta podem ser explorados: *Control-flow Perspective*, *Organizational Perspective* e *Case Perspective*. As três perspetivas são complementares e relevantes para a análise de processos (Cho & R'bigui, 2016).
- **Verificação de Conformidade:** Neste tipo os *events logs* e o modelo de processo existente são comparados. O objetivo é detetar desvios e identificar limitações (Cho & R'bigui, 2016). Por exemplo, um modelo de processo que indique que reservar uma compra é necessário verificar nome e contacto. Neste caso será feita um estudo aos *events logs* para verificar se esta norma é cumprida em ambos (*event log* e modelo).
- **Aprimoramento:** Comporta-se como a conformidade, mas com a diferença que o seu objetivo aqui é enriquecer o modelo com os dados do *events logs*. Por exemplo um modelo de processo com dados de desempenho é usado onde as limitações são mais visíveis (Saravanan & Sree, 2010). Assim é mais fácil detetar onde existe falhas para que se possa, de alguma maneira, melhorar.

2.6 PERSPETIVAS DA DESCOBERTA DE PROCESS MINING

Process Mining é uma ferramenta que extrai informação útil dos *event logs* e que usa como entrada os mesmos, criando um modelo para o processo de negócios que produza os *logs* (Devi & Suryakala, 2014). Para a Descoberta de Processos de Negócio, existem três diferentes perspetivas:

- **Control-flow:** Centraliza-se na ordem das atividades (o chamado controlo de fluxo), tendo como objetivo caracterizar devidamente todos os possíveis caminhos que se podem manifestar em termos de uma *Petri net* ou qualquer outro tipo de modelo (por exemplo BPMN e UML) (van der Aalst, 2011).
- **Organizational:** Dedica-se às informações relativas aos recursos não visíveis no *log* e à forma como estas se relacionam podendo ser por exemplo, pessoas, sistemas, funções e departamentos. O seu objetivo é classificar a informação em categorias individuais ou mostrar eventuais conexões (rede social) (van der Aalst, 2011).
- **Case:** Concentra-se nas características dos casos, sendo que um caso pode ser determinado pelo seu caminho no processo ou autores que nele trabalham e/ou pelos valores dos elementos de dados que lhe correspondem (Devi & Suryakala, 2014). Se um caso representar uma compra online pode ser interessante conhecer o comprador ou o número de produtos comprados.

2.7 LINGUAGENS DE MODELAÇÃO DE PROCESSOS

Desde a revolução industrial, a produtividade tem aumentado devido a inovações técnicas, melhorias na organização do trabalho e uso da tecnologia de informação (van der Aalst, 2011). Isso resultou em mudanças drásticas na organização do trabalho e permitiu novas formas de fazer negócios. Hoje, as inovações em computação e comunicação ainda são os principais fatores por trás das mudanças nos processos de negócios. Portanto, os processos de negócios se tornaram mais complexos, dependem fortemente de sistemas de informação e podem abranger várias organizações. Portanto, a modelação de processos se tornou de maior importância. Os modelos de processo auxiliam na gestão da complexidade, fornecendo informações e procedimentos de documentação. Os sistemas de informação precisam ser configurados e guiados por instruções precisas. Como resultado, os modelos de processo são amplamente usados nas organizações de hoje Deste modo, dois dos principais exemplos de modelação de processos de negócio são: *Petri Net* e *Business Process Modeling Notation* (BPMN).

2.7.1 PETRI NET

Petri Net é uma ferramenta gráfica para uma descrição bem definida do fluxo de atividades em sistemas complexos (Balogh & Kuchárik, 2019). São gráficos diretamente bipartidos, existindo dois tipos de nós, sendo eles as posições, representados como círculos e as transições, representados como quadrados. Os arcos de ligação são representados por setas, usados para conectar dois nós de tipos diferentes, podendo ir de uma posição para uma transição ou de uma transição para uma posição. Na modulação, as posições representam condições ou estados, transições representam eventos ou atividades e um *token* (representado por um ponto no centro da posição) que ocupa uma posição, indica que a condição/estado relacionado a essa posição é válido. Se pelo menos apresentar um *token* em cada entrada de uma transição, isso significa que as condições prévias da transição são mantidas e a transição está ativa. Se uma transição ativa for iniciada, ela remove um *token* de cada local de entrada e vai gerar um *token* em cada local

de saída. Isso significa que o avanço da transição torna a pós-condição verdadeira (Cheng, Juan & Yang, 2014). Na Figura 2.6 é possível visualizar um exemplo de uma *Petri Net*.

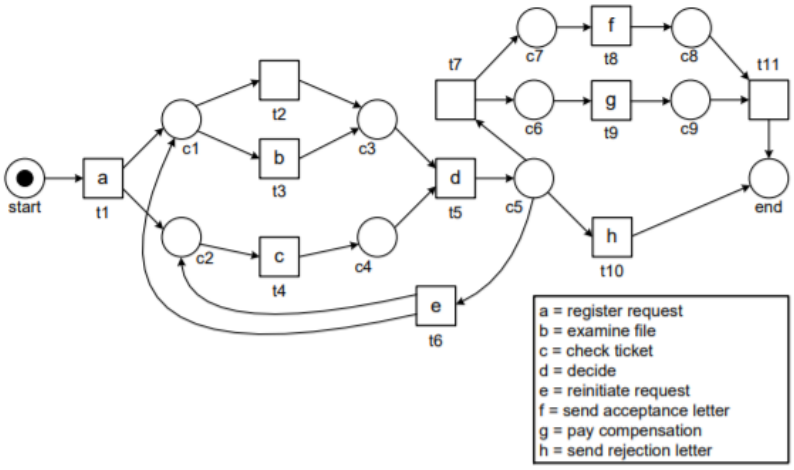


Figura 2.6: Exemplo de uma Petri Net (van der Aalst, 2013).

2.7.2 Business Process Modeling Notation

No caso do BPMN, este tornou-se uma das linguagens mais amplamente usadas para modelar processos de negócios, sendo suportado por muitos fornecedores e abraçado por *Object Management Group* (OMG) (van der Aalst, 2011). A Figura 2.7 mostra um exemplo de BPMN.

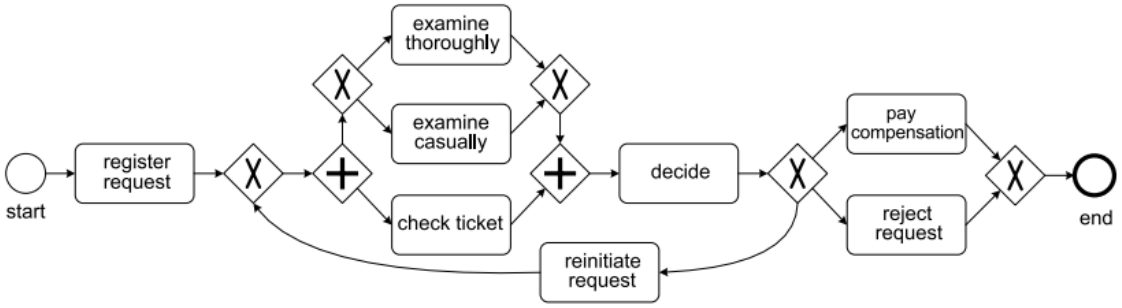


Figura 2.7: Modelo de processo usando BPMN.

Os três conceitos mais básicos do BPMN são os eventos, atividades e arcos de ligação muito semelhante a uma *Petri Net*. Eventos são representados por círculos, atividades por retângulos e os arcos (chamados fluxos de sequência no BPMN) são representados por setas. Além dos mais básicos, o BPMN contem mais de 100 símbolos,

tornando-se uma linguagem bastante complexa (Dumas, Mendling, Rosa & Reijers, 2018). O BPMN aponta um suporte a uma vasta gama de níveis de abstração, incluindo níveis de negócios e níveis de tecnologia de software. Tem como principal objetivo fornecer uma notação de simples perceção por todos os utilizadores de negócios, desde os analistas de rascunhos iniciais dos processos até aos desenvolvedores técnicos responsáveis pela implementação da tecnologia que executará esses processos e, por fim, às pessoas de negócios que irão gerir esses processos (Weske, 2007).

2.8 ALGORITMOS

A disciplina de *Process Mining* tem muitas abordagens diferentes, onde estas são algoritmos com diferentes particularidades. Cada algoritmo pode ser mais adequado para um determinado *event log*. Conforme as principais perspetivas já mencionadas numa secção anterior, os algoritmos podem ser agrupados em cada uma delas. Na perspetiva de *Control-flow*, destacam-se os algoritmos *Alpha Miner*, *Heuristic Miner*, *Multi-phase Miner*, *Fuzzy Miner* e *Genetic Miner* enquanto na perspetiva organizacional destacam-se os algoritmos *Organizational Model Miner* e *Social Network Miner*. (van der Aalst, 2011)

2.8.1 ALPHA MINER

Este algoritmo foi o primeiro a ser utilizado para analise de *event logs*, apresentado por van der Aalst, Weijters e Măruşter. Geralmente como a maior parte das coisas que são criadas primeiro, o *Alpha Miner* apresentou várias debilidades no momento que foi posto em prática. O *Alpha Miner* pressupõe que o *event log* não contenha nenhum ruído (Backer, Baesens, Vanthienen & Weerdt, 2012). Além disso, permite que os utilizadores apenas se concentrem no fluxo do processo principal, em vez de em cada detalhe comportamental que aparece no log do processo, fazendo parecer que todo o comportamento desse mesmo processo está presente (Saravanan & Sree, 2010). É um algoritmo simples do ponto de vista científico pela maneira como foi formalizado (apenas 8 linhas), mas para *logs* da vida real, quase nunca é a escolha certa pelo resultado não ser o mais correto (Rozinat, 2010).

O resultado do *Alpha Miner* é conseguido através de algumas relações dentro do registo de eventos (van der Aalst, 2011):

- Uma das relações é descrever quais tarefas aparecem em sequência (por exemplo A > B).
- A partir da sequência de tarefas é referido como a relação causal deriva do registo de eventos W. Com W = [ABCD, ACBD, AED] é descoberta a sequência de tarefas W = [A>B, A>C, A>E, B>C, B>D, C>D e E>D], sendo a relação causal W = [A-B, A-C, A-E, B-D, C-D e E-D], onde B não segue C porque C>B (duas tarefas só podem seguir um caminho direto, não existe *loop*).
- Para o caso de cima, é sugerido comportamento concorrente, ou seja, potencial paralelismo. Se duas tarefas podem seguir uma á outra diretamente em qualquer ordem (B>C e C>B), então é provável que estejam em paralelo.
- Por fim a última relação que existe no *Alpha Miner* é a quantidade de pares de transições que nunca se seguem diretamente. Não há relações causais direta e o paralelismo é improvável.

2.8.2 HEURISTIC MINER

Com o conhecimento da existência do *Alpha Miner*, surgiu o algoritmo *Heuristic Miner* criado pelo Dr. Ton Weijters, com o intuito de melhorar o algoritmo anterior, como por exemplo, o caso de ruido. O *Heuristic Miner* pode descobrir loops curtos e dependências não locais, mas não possui a capacidade de detetar atividades duplicadas. Este é propenso a ser resiliente ao ruído e, portanto, é esperado ser robusto em um contexto da vida real (Backer et al., 2012). Permite que os utilizadores se concentrem no fluxo do processo principal, em vez de em cada detalhe do comportamento que aparece no log do processo, muito semelhante neste aspeto ao *Alpha Miner* (Saravanan & Sree, 2010). Basicamente este algoritmo foca-se mais nos acontecimentos com maior frequência e não especificamente na sua ordem (Medeiros, van der Aalst & Weijters, 2006). Este algoritmo é ótimo quando temos dados da vida real com muitos eventos diferentes tendo como vantagem uma rede heurística poder ser convertida para outros tipos de modelos de processos, como uma *Petri Net* para análises adicionais no *ProM* (Rozinat, 2010).

Toma em consideração as frequências de eventos e sequencias ao construir um modelo de processo. A ideia base é que caminhos poucos frequentes não devem ser

incorporados ao modelo. Tendo em conta A>B (A segue B) e/ou B>A (B segue A) é calculado o valor de dependência entre A e B através de:

$$|A => B| = \begin{cases} \dfrac{|A > B| - |B > A|}{|A > B| + |B > A| + 1}, & se\ A \neq B \\ \dfrac{|A > A|}{|A > A| + 1}, & se\ A = A \end{cases}$$

$|A => B|$ produz um valor entre -1 e 1.

Quanto mais perto do valor +1, maior é a dependência entre A e B. Um valor deste tipo só pode ser alcançado se A for frequentemente seguido diretamente por B, mas B é dificilmente seguido por A. Um valor perto de -1 tem ocorrência contrária a um acontecimento de +1. Se A for seguido por ele próprio, isso sugere um *loop* e uma forte dependência reflexiva (van der Aalst, 2011).

2.8.3 MULTI-PHASE MINER

Este algoritmo foi desenvolvido pelo Dr. Boudewijn van Dongen, um dos criadores do *software ProM*. Ele usa EPCs (*Event-driven Process Chains*) como uma representação padrão. Entende-se por EPC, uma linguagem de modulação de processos de negócios para a representação de dependências lógicas e temporais das atividades em um processo de negócios (Mendling, 2008). No entanto, podem ser convertidos em outros formatos, como vários tipos de *Petri net*, modelos YAWL (*Yet Another Workflow Language*), etc. Um modelo YAWL é baseado em *Petri net*, mas estendido com recursos adicionais para facilitar a modelação de fluxos de trabalho complexos (Hofstede & van der Aalst, 2005). Conforme encontrado em EPCs foi permitido expressar comportamento complexo em modelos relativamente bem estruturados. Uma das vantagens do *Multi-phase Miner* é que ele constrói um modelo que sempre "se encaixa" num *log* de eventos completo (Rozinat, 2010). Como desvantagens, ele tem uma tendência a generalizar demais, ou seja, às vezes o modelo permite maior procedimento (van der Aalst, 2007). No entanto, sendo um algoritmo para *log data* mais simples raramente é útil para processos mais complexos porque o modelo torna-se ilegível.

2.8.4 FUZZY MINER

Estamos perante um dos mais novos algoritmos de *Process Mining* desenvolvido por Christian W. Günther. É o primeiro algoritmo para abordar diretamente os problemas de um grande número de atividades. A principal contribuição do *Fuzzy Miner* é que

também pode ser aplicado a processos menos estruturados ou não estruturados dos quais os *event logs* não podem ser facilmente resumidos em modelos de processo estruturados e concisos. Embora essa abordagem seja uma técnica extraordinária de exploração de dados, ela sofre de uma desvantagem no sentido de que um modelo *Fuzzy* não pode ser traduzido para uma *Petri Net* formal que limita severamente uma avaliação comparativa a outras técnicas de descoberta de processos (Backer et al., 2012). Usa métricas de significância e semelhança para simplificar interactivamente o modelo de processo no nível desejado de abstração (Porouhan & Premchaiswadi, 2018). Tornou-se uma das ferramentas mais úteis em aplicações de estudo, sendo capaz de limpar uma grande quantidade de comportamento confuso (Gunther & van der Aalst, 2007). Consegue criar um padrão para processos concluídos ou em execução com base na sequência de eventos, considerando a frequência dos eventos e a sequência dos executados (Jangvaha, Palangsantikul, Porouhan & Premchaiswadi, 2017).

2.8.5 GENETIC MINER

Este algoritmo foi criado somente para estudo de individuos, mais concretamente a evolução de uma população, como por exemplo o desenvolvimento de uma familia, entre pais, filhos, etc. Algoritmos genéticos são métodos de busca adaptativa que tentam imitar o processo de evolução, começando com uma população inicial de indivíduos sendo que a cada indivíduo é atribuído uma medida de aptidão (*fitness*) para indicar a sua qualidade (Medeiros, van der Aalst & Weijters, 2006). As populações evoluem selecionando os indivíduos mais aptos e gerando novos indivíduos usando operadores genéticos como *crossover* (combinando partes de dois ou mais indivíduos) e mutação (modificação aleatória de um indivíduo). Podem ser usados para descobrir modelos de *Petri net* a partir de *event logs* (Medeiros, van der Aalst & Weijters, 2005). Este algoritmo cumpre as seguintes etapas principais:

(I) Ler o *event log*;

(II) calcular relações de dependência entre atividades;

(III) construir a população inicial;

(IV) calcular a aptidão dos indivíduos;

(V) parar e devolver os indivíduos mais aptos (ou continuar para etapa seguinte);

(VI) criar a próxima população utilizando os operadores genéticos.

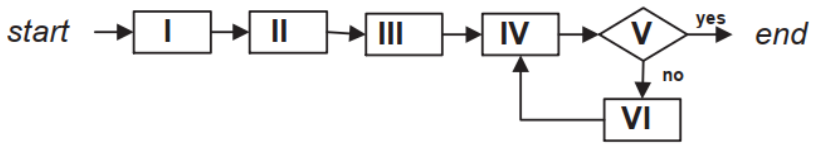


Figura 2.8: Etapas do algoritmo *Genetic Miner* (Medeiros, van der Aalst & Weijters, 2005).

2.8.6 Organizational Model Miner e Social Network Miner

Ambos têm algo em comum: os utilizadores. O *Social Network Miner* consegue identificar o papel e interação do utilizador com outros dentro da organização. Este pode lidar com ruídos e exceções, e permite que os utilizadores se concentrem no fluxo do processo principal em vez de em cada detalhe do comportamento que aparece no log do processo (Saravanan & Sree, 2011). Quando analisamos os *logs* de processo, é difícil encontrar uma hierarquia explícita de unidades organizacionais. No entanto, é possível encontrar grupos nos quais as pessoas têm permissão para executar tarefas semelhantes. Apenas um grupo específico tem permissão para realizar tarefas semelhantes. Assim, a partir de um "perfil" descrevendo a frequência com que os indivíduos realizam tarefas específicas, podemos derivar grupos. Um grupo pode ser uma unidade organizacional ou um agrupamento de pessoas que executam as mesmas funções na vida real (Song & van der Aalst, 2008). Basicamente o *Organizational Model Miner* identifica e adquire a informação do criador de cada evento e liga atividades a cada utilizador, juntando os semelhantes. Depaire

2.9 Medidas de qualidade dos modelos de processo.

Avaliar a qualidade de um modelo descoberto é essencial para descobrir se ele constitui uma representação adequada do processo. No âmbito do PM têm sido definidas várias medidas de qualidade. A qualidade dos modelos de processo descobertos foi dividida em quatro dimensões: *fitness*, *precision*, *simplicity* e *generalization* (Depaire, Donders, Janssenswillen & Jouck, 2017).

- ***Fitness***: Quantifica em que medida o modelo descoberto pode reproduzir precisamente os casos registados no *log*. *Heuristic Miner* e *Genetic Miner* focam-se no *fitness* como guia principal na descoberta de um modelo de processo, mas não garantem ótimos resultados (Buijs, van der Aalst & van Dongen, 2012). Um modelo tem um *fitness* perfeito se todos os caminhos no *log* puderem ser reproduzidos pelo modelo do começo ao fim (Rudnitckaia, 2014).
- ***Precision***: Quantifica a fração do comportamento permitido pelo modelo que não é visto no *event log* (Buijs et al., 2012), logo deve evitar *"underfitting"* para que tenha uma ótima Precision. *"Underfitting"* significa que o modelo permite comportamentos muito diferentes do que foi visto no *log* (Rudnitckaia, 2014).
- ***Simplicity***: O modelo descoberto deve ser o mais simples possível (Rudnitckaia, 2014). A complexidade de um modelo de processo é capturada pela dimensão de *Simplicity*. Os algoritmos de descoberta de processos geralmente resultam em modelos de processo parecidos com "esparguete", que são modelos de processo muito difíceis de ler. Os que se concentram fortemente em *Simplicity* é a classe dos *Alpha Miner*. Essas técnicas de descoberta geralmente resultam em modelos simples, mas com baixo *fitness* e / ou *precision* (Buijs et al., 2012).
- ***Generalization***: Avalia até que ponto o modelo resultante será capaz de reproduzir o comportamento futuro do processo. Nesse sentido, a generalização também pode ser vista como uma medida da confiança em *Precision*. Se existir muitos caminhos possíveis, é improvável que o próximo caso seja adequado (Buijs et al., 2012). Para uma boa generalização deve-se evitar *"Overfitting"*. Este termo demonstra um modelo que explica o *log* da amostra específica, mas um próximo *log* da amostra do mesmo processo pode vir a produzir um processo completamente diferente do modelo. *Generalization* especifica que os modelos devem generalizar e não apenas restringir o comportamento à amostra contida nos *event logs*. Em outras palavras, um modelo com alta qualidade também deve ser capaz de reproduzir um comportamento nunca antes visto do processo (Depaire et al., 2017). Exemplos de algoritmos com resultados bons em generalização são o *Fuzzy Miner* e o *Heuristic Miner* (Rudnitckaia, 2014).

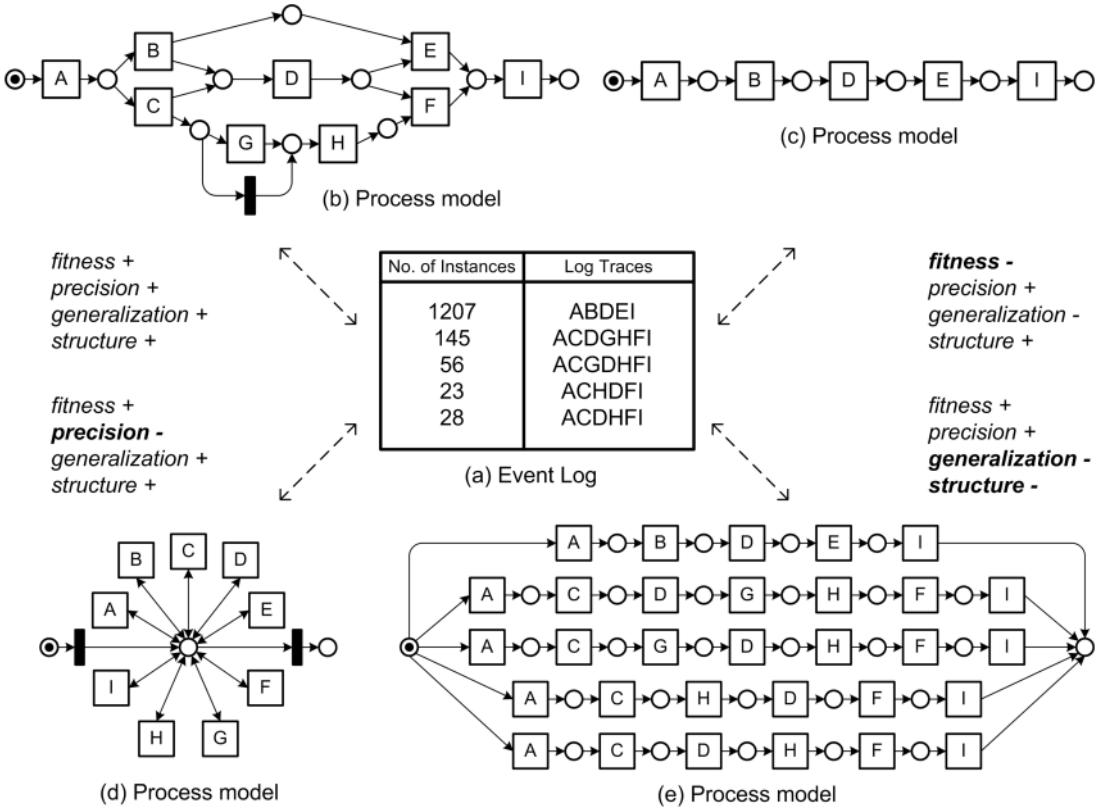


Figura 2.9: A avaliação de um modelo de processo pode ocorrer em diferentes dimensões (Gunther et al., 2007)

2.10 FERRAMENTAS

Os principais produtos de *Business Inteligence* fornecidos, tais como *Oracle* e *Microsoft*, não oferecem suporte ao *Process Mining* por serem centrados em dados, além de limitados quando se trata de formas mais avançadas de análise e desconhecem os processos aos quais os dados se referem. As plataformas de *software* tradicionais não são úteis para *Process Mining*, por esse motivo houve necessidade de desenvolver ferramentas autónomas focadas nesta disciplina. Simples ferramentas com o passar dos anos foram desenvolvidas. No entanto estas foram baseadas em modelos de processos simples e pequenos, não fornecendo suporte para projetos de *Process Mining* da vida real (escalabilidade, interface intuitiva do utilizador, etc.). Chegou-se à conclusão que não fazia sentido criar uma ferramenta de *Process Mining* dedicada para todas as técnicas de processo de descoberta recém-concebidas. Essa observação desencadeou o desenvolvimento da estrutura *ProM*, um ambiente "*plug-able*" para *Process Mining*

usando MXML como formato de entrada (van der Aalst, 2011). Mais tarde outras ferramentas também foram desenvolvidas, sendo *Disco* uma delas.

2.10.1 PROM

O *ProM* é uma ferramenta desenvolvida pela Universidade de Tecnologia de Eindhoven com o intuito de apoiar e analisar modelos de processo. É uma ferramenta de código aberto especialmente adaptada para suportar o desenvolvimento de *plug-ins* de *Process Mining*, contendo uma grande variedade destes (Bose & Verbeek, 2010). O crescimento do número de *plug-ins* (mais de 300 *plug-ins*) no período de 2004 (primeira versão *ProM 1.1*) a 2009 (*ProM 5.2*) ilustra que o *ProM* alcançou o seu objetivo inicial de fornecer uma plataforma para o desenvolvimento de novas técnicas de *Process Mining*. Esta ferramenta é maioritariamente usada no contexto de projetos de pesquisa conjuntos, projetos de mestrado e projetos de consultoria (van der Aalst, 2011). O *ProM* 6 foi desenvolvido em 2010, baseado no XES, e não no MXML. XES é o novo padrão de *Process Mining* adotado pela *IEEE Task Force* e consagra-se até agora como a última versão desta ferramenta. A grande variedade de *plug-ins* que esta dispõe assume-se simultaneamente como uma vantagem e desvantagem para análise de modelos de processo, isto porque por um lado a variedade de *plug-ins* auxilia o utilizador pela sua capacidade de analisar e gerar modelos muito variados e compostos, que também se pode apresentar como uma desvantagem tendo em conta que a variedade pode tornar-se excessiva/confusa aquando a análise dificultando-a pelo grande numero de plug-ins correntes ao mesmo tempo (van der Aalst, 2011). Sendo uma ferramenta de código aberto, pode ser feito o seu download gratuito através de www.processmining.org.

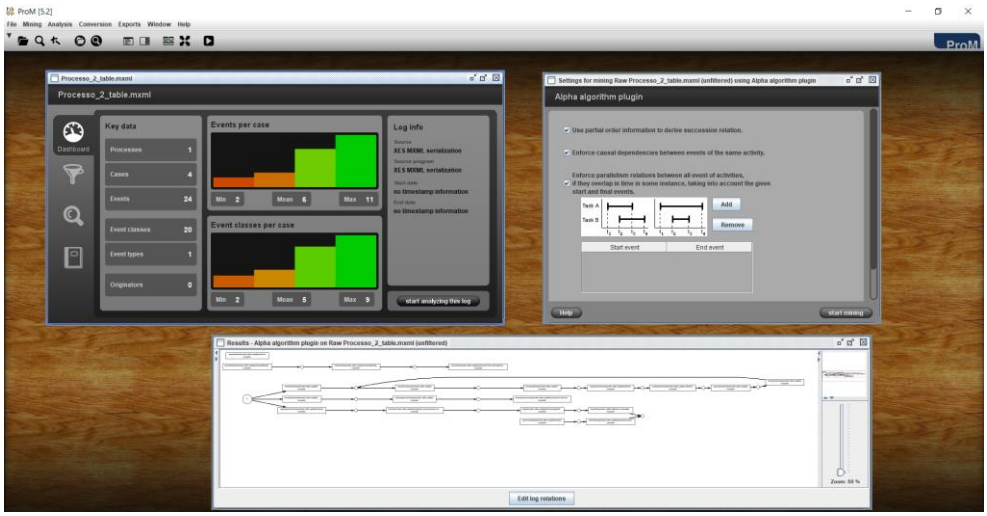


Figura 2.10: Ambiente de trabalho da ferramenta *ProM 5.2*.

2.10.2 DISCO

O *Disco* é uma ferramenta desenvolvida pela empresa *Fluxicon*. Ao contrário do *ProM*, esta é comercial para *Process Mining*, mas pode ser utilizado como versão completa tendo uma licença académica (Akcetin, Celik & Yaldir, 2016). É uma ferramenta independente para análise de *Process Mining*, com foco no alto desempenho (ou seja, no gerenciamento de conjuntos de dados grandes e complexos) e na facilidade de uso, sendo baseado no algoritmo *Fuzzy Miner* (van der Aalst, 2011). Um dos pontos fortes desta ferramenta é que a sua versão completa pode lidar com grandes *event logs* e *big data*. Além disso, a ferramenta tem várias opções de filtragem (ou seja, como *Timeframe*, *Variation*, *Performance*, *endpoints*, *Attributes* e *Follower filtering*) que o tornam adequado para limpeza e manipulação dos dados de diferentes aspetos e dimensões em relação à finalidade/objetivo do utilizador (Koosawad et al., 2018). O principal objetivo do *Disco* é criar imagens gráficas do processamento dos dados que podem ser usados para melhorar e aumentar a eficiência do sistema para o qual é destinado, além que o seu ambiente é de fácil utilização, o que o torna popular para os utilizadores (Jangvaha et al., 2017).

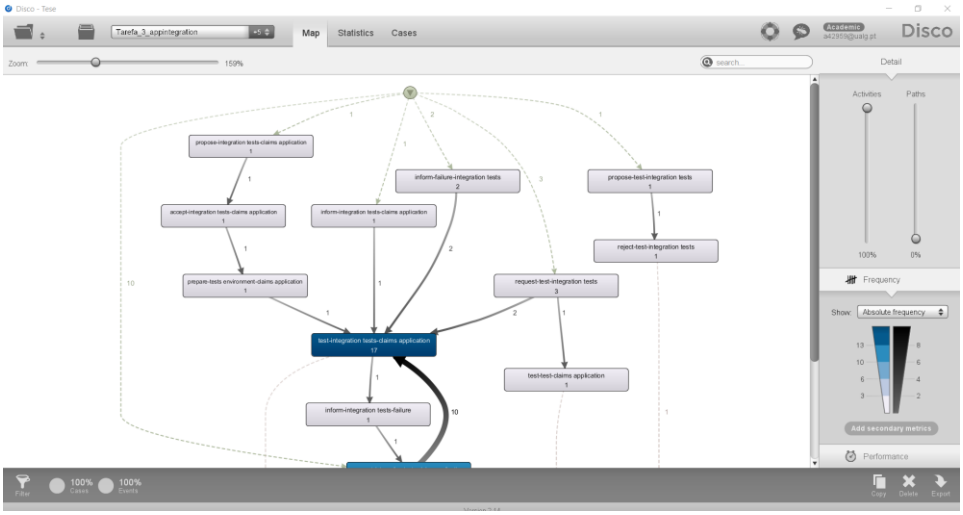


Figura 2.11: Ambiente de trabalho da ferramenta *Disco*.

2.10.3 CARACTERÍSTICAS DO PROM E DISCO

Segundo (Rudnitckaia, 2014), as principais características das ferramentas *ProM* e *Disco* são descritas em baixo.

Para a ferramenta *ProM*, destaca-se:

- Visa cobrir tudo o que respeita ao *Process Mining*;
- Suporta notações como *Petri Net*, BPMN, C-Nets, Fuzzy models, etc;
- Também suporta verificação de conformidade e suporte operacional;
- Muitos *plug-ins* são protótipos experimentais e não são fáceis de usar.

Referindo à ferramenta *Disco*, destaca-se:

- Foco na descoberta e análise de desempenho (incluindo animação);
- Poderosos recursos de filtragem para comparação *Process Mining* e verificação *ad-hoc* de padrões;
- Não suporta verificação de conformidade e suporte operacional;
- Fácil de usar e excelente desempenho.

2.10.4 OUTRAS FERRAMENTAS

Além das duas ferramentas explicadas acima, existem várias outras com o mesmo objetivo. De acordo com (van der Aalst, 2011) a Tabela 2.2 abaixo representada, descreve outras ferramentas com os seus devidos nomes, tipo (comercial, académico e *open-source*) e Organização de origem.

Tabela 2.2 - Ferramentas de Process Mining.

Ferramenta	Tipo	Organização
ARIS Process Performance Manager	Comercial	Software AG
Enterprise Visualization Suite	Comercial	Businesscape
Genet/Petrify	Académico	Universitat Politècnica de Catalunya
Interstage BPME	Comercial	Fujitsu
OKT Process Mining suite	Open-source	Exeura s.r.l.
Process Discovery Focus	Comercial	Iontas (Verint Systems)
ProcessAnalyzer	Comercial	QPR
Rbminer/Dbminer	Académico	Universitat Politècnica de Catalunya

Reflect|one	Comercial	Pallas Athena
Reflect	Comercial	Futura Process Intelligence
ServiceMosaic	Académico	University of New South Wales

2.11 ANÁLISE COMPARATIVA DOS ALGORITMOS

Nesta secção é resumido na tabela 2.3 o estudo dos algoritmos *Alpha Miner*, *Heuristic Miner*, *Multi-Phase Miner*, *Fuzzy Miner* e *Genetic Miner* mencionados e explicados já anteriormente, focando de uma maneira mais simples uma forma de analisar e escolher quais são os melhores algoritmos para cada tipo de dados. Essa análise consiste em mencionar a sua utilidade para cada tipo de dados, o tipo de notação a ser usado para o seu modelo e também algumas das suas vantagens e desvantagens.

Tabela 2.3: Comparação dos algoritmos em termos da sua utilidade, notação, vantagens e desvantagens.

Algoritmo	Utilidade	Linguagem/ Notação	Vantagens	Desvantagens
Alpha Miner	Logs simples que não contenham ruído; Estudos teóricos	Petri Net	Simples do ponto de vista científico (formalizado apenas em 8 linhas).	Sensível ao ruído; Não tolera event logs reais; Não se foca em cada detalhe do comportamento que aparece no log do processo.
Heuristic Miner	Dados da vida real com muitos eventos diferentes; Necessário um modelo de Petri Net para análises adicionais no ProM.	Heuristic Net (pode ser convertido em Petri Net).	Descobre loops curtos e dependências não locais; Supera bem a presença de ruído; Ótimo para eventos da vida real.	Não possui a capacidade de detetar atividades duplicadas; Não se foca em cada detalhe do comportamento que aparece no log do processo.
Multi-Phase Miner	Quando possui dados de log simples e estruturados.	EPC (pode ser convertido em Petri Net e YAWL).	Constrói um modelo que sempre "se encaixa" num *event logs* completo.	Generaliza demais, ou seja, às vezes o modelo permite maior procedimento; Processos mais complexos o modelo torna-se ilegível.
Fuzzy Miner	Aplicado a processos poucos estruturados; aborda diretamente os problemas de um grande número de atividades.	Fuzzy model.	Capaz de limpar uma grande quantidade de comportamento confuso; simplifica o modelo de processo no nível desejado de abstração.	O Fuzzy model não pode ser traduzido para uma Petri Net.
Genetic Miner	Busca adaptativa que tenta imitar o processo de evolução de uma população.	Heuristic Net (pode ser convertido em Petri Net).	Unico que estuda a evolução natural dos individuos dentro de uma população (neste caso modelos de processo).	Carrega e requer muitos recursos do computador.

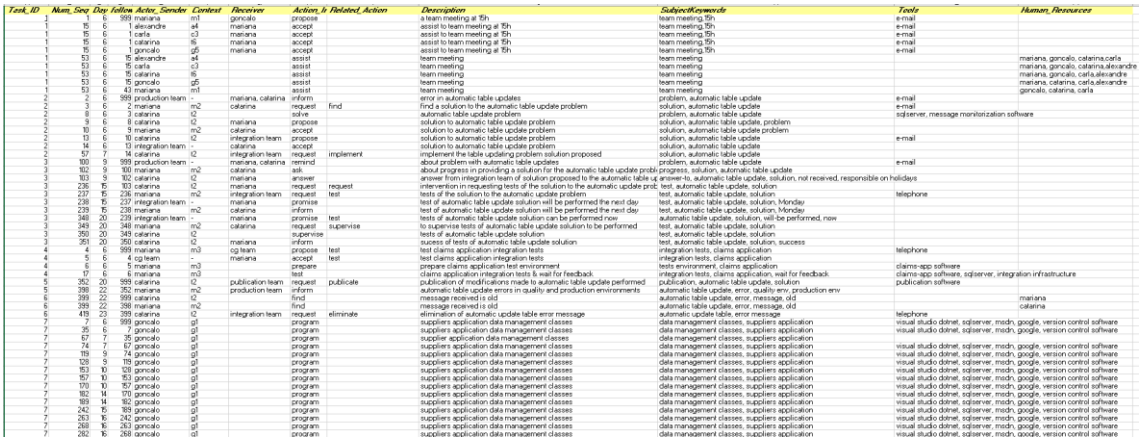
CAPÍTULO 3

PREPARAÇÃO PARA ANÁLISE

3.1 DESCRIÇÃO DOS DADOS

Os dados foram o resultado de um estudo prévio (Zacarias, 2008). O estudo de caso envolveu uma equipa de desenvolvimento de software de 4 programadores (Gonçalo, Carla, Catarina, Alexandre) e a líder do projeto (Mariana), que executa tarefas de programação e gerenciamento de projetos. A equipa desenvolve aplicações de web para um banco comercial. Os membros da equipa executam tarefas de análise, design, programação, teste e manutenção de sistemas. Durante o período de observação, a equipa trabalhou nas seguintes aplicações; (1) Fornecedores, (2) Reclamações, (3) Correspondência por Correio dos Clientes (tem como nome aplicação Mail), (4) Despejos e (5) Campanhas de Marketing.

Após a recolha, os verbos recorrentes foram identificados, discutidos e normalizados (Zacarias, 2008). Essa normalização envolveu primeiro a identificação de sinónimos e homónimos. Em seguida, uma lista de tipos de ação representando todas as ações recolhidas foi discutida e definida. Um verbo único e distinto foi associado a cada tipo de ação. Em total, foram registadas 653 ações num ficheiro Excel. O ficheiro continha o identificador da tarefa (*TASK_ID)*, número de sequência (*NUM_SEQ*), dia (Day), seguimento (*follows*), ator remetente (*ACTOR_Sender*), contexto (*Context*), recetor (*Receiver*), atividade (*Action_Interaction*), relação (*Related_Action*), descrição (*Description*), palavra-chave do assunto (*SubjectKeyword*), ferramentas (*Tools*) e recursos humanos (Human_Resources). Através da Figura 3.1 pode visualizar-se o conteúdo descrito anteriormente.



Task_ID	Num_Seq	Day	follow	Actor_Sender	Context	Receiver	Action_Ir	Related_Action	Description	SubjectKeywords	Tools	Human_Resources
1	1	6	999	mariana	m1	goncalo	propose		a team meeting at 15h	team meeting,15h	e-mail	
1	15	6	1	alexandre	a4	mariana	accept		assist to team meeting at 15h	team meeting,15h	e-mail	
1	15	6	1	carla	c3	mariana	accept		assist to team meeting at 15h	team meeting,15h	e-mail	
1	15	6	1	catarina	t6	mariana	accept		assist to team meeting at 15h	team meeting,15h	e-mail	
1	15	6	1	goncalo	g5	mariana	accept		assist to team meeting at 15h	team meeting,15h	e-mail	
1	53	6	15	alexandre	a4		assist		team meeting	team meeting		mariana, goncalo, catarina,carla
1	53	6	15	carla	c3		assist		team meeting	team meeting		mariana, goncalo, catarina,alexandre
1	53	6	15	catarina	t6		assist		team meeting	team meeting		mariana, goncalo, carla,alexandre
1	53	6	15	goncalo	g5		assist		team meeting	team meeting		mariana, catarina, carla,alexandre
1	53	6	43	mariana	m1		assist		team meeting	team meeting		goncalo, catarina, carla
2	2	6	999	production team	-	mariana, catarina	inform		error in automatic table updates	problem, automatic table update	e-mail	
2	3	6	2	mariana	m2	catarina	request	find	find a solution to the automatic table update problem	solution, automatic table update	e-mail	
2	8	6	3	catarina	t2		solve		automatic table update problem	problem, automatic table update	sqlserver, message monitorization software	
2	9	6	8	catarina	t2	mariana	propose		solution to automatic table update problem	solution, automatic table update, problem		
2	10	6	9	mariana	m2	catarina	accept		solution to automatic table update problem	solution, automatic table update problem		
2	13	6	10	catarina	t2	integration team	propose		solution to automatic table update problem	solution, automatic table update	e-mail	
2	14	6	13	integration team	-	catarina	accept		solution to automatic table update problem	solution, automatic table update		
2	57	7	14	catarina	t2	integration team	request	implement	implement the table updating problem solution proposed	solution, automatic table update		
3	100	9	999	production team	-	mariana, catarina	remind		about problem with automatic table updates	problem, automatic table update	e-mail	
3	102	9	100	mariana	m2	catarina	ask		about progress in providing a solution for the automatic table update probl	progress, solution, automatic table update		
3	103	9	102	catarina	t2	mariana	answer		answer from integration team of solution proposed to the automatic table up	answer-to, automatic table update, solution, not received, responsible on holidays		
3	236	15	103	catarina	t2	mariana	request	request	intervention in requesting tests of the solution to the automatic update prob	test, automatic table update, solution		
3	237	15	236	mariana	m2	integration team	request	test	tests of the solution to the automatic update problem	test, automatic table update, solution	telephone	
3	238	15	237	integration team	-	mariana	promise		test of automatic table update solution will be performed the next day	test, automatic table update, solution, Monday		
3	239	15	238	mariana	m2	catarina	inform		test of automatic table update solution will be performed the next day	test, automatic table update, solution, Monday		
3	348	20	239	integration team	-	mariana	promise	test	tests of automatic table update solution can be performed now	automatic table update, solution, will be performed, now		
3	349	20	348	mariana	m2	catarina	request	supervise	to supervise tests of automatic table update solution to be performed	test, automatic table update, solution		
3	350	20	349	catarina	t2		supervise		tests of automatic table update solution	test, automatic table update, solution		
3	351	20	350	catarina	t2	mariana	inform		sucess of tests of automatic table update solution	test, automatic table update, solution, success		
4	4	6	999	mariana	m3	cg team	propose	test	test claims application integration tests	integration tests, claims application	telephone	
4	5	6	4	cg team	-	mariana	accept	test	test claims application integration tests	integration tests, claims application		
4	6	6	5	mariana	m3		prepare		prepare claims application test environment	tests environment, claims application	claims-app software	
4	17	6	6	mariana	m3		test		claims application integration tests & wait for feedback	integration tests, claims application, wait for feedback	claims-app software, sqlserver, integration infrastructure	
5	352	20	999	catarina	t2	publication team	request	publicate	publication of modifications made to automatic table update performed	publication, automatic table update, solution	publication software	
5	398	22	352	mariana	m2	production team	inform		automatic table update errors in quality and production environments	automatic table update, error, quality env, production env		
6	399	22	999	catarina	t2		find		message received is old	automatic table update, error, message, old		mariana
6	399	22	398	mariana	m2		find		message received is old	automatic table update, error, message, old		catarina
6	419	23	399	catarina	t2	integration team	request	eliminate	elimination of automatic update table error message	automatic update table, error message	telephone	
7	7	6	999	goncalo	g1		program		suppliers application data management classes	data management classes, suppliers application	visual studio dotnet, sqlserver, msdn, google, version control software	
7	35	6	7	goncalo	g1		program		suppliers application data management classes	data management classes, suppliers application	visual studio dotnet, sqlserver, msdn, google, version control software	
7	67	7	35	goncalo	g1		program		supplier application data management classes	data management classes, suppliers application		
7	74	7	67	goncalo	g1		program		suppliers application data management classes	data management classes, suppliers application	visual studio dotnet, sqlserver, msdn, google, version control software	
7	119	9	74	goncalo	g1		program		suppliers application data management classes	data management classes, suppliers application	visual studio dotnet, sqlserver, msdn, google, version control software	
7	128	9	119	goncalo	g1		program		suppliers application data management classes	data management classes, suppliers application	visual studio dotnet, sqlserver, msdn, google, version control software	
7	153	10	128	goncalo	g1		program		suppliers application data management classes	data management classes, suppliers application	visual studio dotnet, sqlserver, msdn, google, version control software	
7	157	10	153	goncalo	g1		program		suppliers application data management classes	data management classes, suppliers application	visual studio dotnet, sqlserver, msdn, google, version control software	
7	170	10	157	goncalo	g1		program		suppliers application data management classes	data management classes, suppliers application	visual studio dotnet, sqlserver, msdn, google, version control software	
7	182	14	170	goncalo	g1		program		suppliers application data management classes	data management classes, suppliers application	visual studio dotnet, sqlserver, msdn, google, version control software	
7	189	14	182	goncalo	g1		program		suppliers application data management classes	data management classes, suppliers application	visual studio dotnet, sqlserver, msdn, google, version control software	
7	242	15	189	goncalo	g1		program		suppliers application data management classes	data management classes, suppliers application	visual studio dotnet, sqlserver, msdn, google, version control software	
7	263	16	242	goncalo	g1		program		suppliers application data management classes	data management classes, suppliers application	visual studio dotnet, sqlserver, msdn, google, version control software	
7	268	16	263	goncalo	g1		program		suppliers application data management classes	data management classes, suppliers application	visual studio dotnet, sqlserver, msdn, google, version control software	
7	282	16	268	goncalo	g1		program		suppliers application data management classes	data management classes, suppliers application	visual studio dotnet, sqlserver, msdn, google, version control software	

Figura 3.1 - Ficheiro original dos dados recolhidos.

3.2 ORGANIZAÇÃO DOS DADOS

O ficheiro encontrava-se desorganizado e foi então como segunda fase o objetivo de organizá-lo. Notava-se que o número de sequência (*NUM_SEQ*) e o seguimento (*follows*) não seguiam uma ordem e que todas as atividades com o número de *follows* "999" seriam a primeira atividade de um caso, que era inexistente no ficheiro. Então foi necessário criar mais uma coluna intitulada de *CASE_ID*, onde iria ser mencionado o número de cada caso consoante o número de sequência e seguimento. Por exemplo, para o primeiro caso foi-se ao encontro da primeira linha do ficheiro onde se tinha um *follows* "999" com um número de sequência "1". A partir daí por todo o ficheiro o objetivo foi encontrar uma conjugação entre o número de sequência e o *follows*. A segunda linha do primeiro caso teve como número de sequência "15" e *follows* "1", onde o resto foi assim sucessivamente. A partir da Figura 3.2 pode visualizar-se todo o primeiro caso.

TASK_ID	CASE_ID	Day	Num_Seq	follows	Actor_Sender	Context	Receiver	Action_Interaction
1	1	6	1	999	mariana	m1	goncalo	propose
1	1	6	15	1	alexandre	a4	mariana	accept
1	1	6	15	1	carla	c3	mariana	accept
1	1	6	15	1	catarina	t6	mariana	accept
1	1	6	15	1	goncalo	g5	mariana	accept
1	1	6	53	15	alexandre	a4		assist
1	1	6	53	15	carla	c3		assist
1	1	6	53	15	catarina	t6		assist
1	1	6	53	15	goncalo	g5		assist
1	1	6	53	43	mariana	m1		assist

Figura 3.2 - Todo o caso (CASE_ID) "1" apresentado.

É de notar que existem número de sequências e follows repetidos. Isto deve-se ao facto de existir mais que um *Actor_Sender* presente no mesmo caso para uma mesma especifica atividade (*Action_Interaction*). Um caso termina no momento que já não existe

mais nenhum *follows* de uma igual sequência anterior. Supondo isso o processo para o encontro de um segundo caso é feito da mesma maneira, procurando novamente um *follows* "999". Todo este processo foi feito igualmente para o resto do ficheiro concluindo com um total de 141 casos.

Com os casos todos descobertos teve-se em atenção, de seguida, a coluna do contexto (*Context*), sendo uma terceira fase necessária. Contexto é qualquer informação que possa ser utilizada para caracterizar a situação de uma entidade. Uma entidade é uma pessoa, local ou objeto que é considerado relevante para a interação entre um utilizador e uma aplicação, incluindo o próprio utilizador e as aplicações. Se uma informação pode ser usada para caracterizar a situação de um participante de uma interação, essa informação é contexto (Abowd & Dey, 2015). Um sistema sensível ao contexto é um sistema capaz de fornecer serviços ou informações adequadas em relação à tarefa de um utilizador. Informações adicionais de contexto podem ser extraídas dos *event logs*, como por exemplo agrupamento de eventos semelhantes (Becker & Intoyad, 2017). Neste caso o contexto teve como papel agrupar os eventos semelhantes entre pessoas (*Actor_Sender* e *Receiver*) e atividades (*Action_Interaction*). Houve a necessidade de auxílio da descrição (*Description*) e de algumas palavra-chave (*SubjectKeywords*) pelo facto de cada atividade conter apenas um verbo, dificultando o agrupamento dos casos através de apenas o contexto.

A terceira fase começa aqui com o agrupamento dos casos consoante o contexto. Para uma melhor noção de quantos processos foram criados com os casos agrupados, consoante o contexto, o ficheiro foi dividido em vários outros ficheiros mais pequenos. Com isto resultou em 17 ficheiros, onde cada ficheiro a partir da Figura 3.3, tem um nome específico para facilitar o conhecimento do conteúdo presente nele. De certa forma, na secção dos resultados, os processos divididos em vários ficheiros vai tornar a análise mais facilitada.

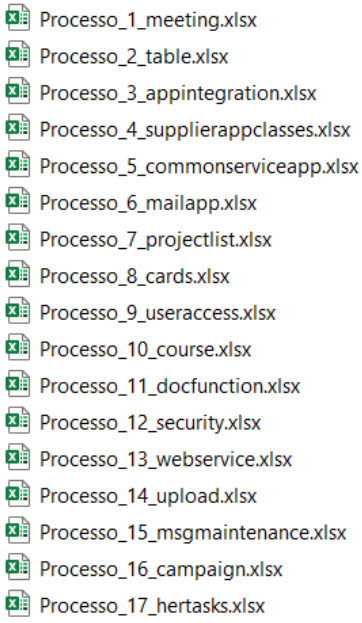


Figura 3.3 - Ficheiros de processo após agrupar os casos consoante o contexto.

Terminada a terceira fase, conclui-se assim a organização dos dados por completo. Através da Tabela 3.1 é possível visualizar o Número de Casos, Número de Eventos e Número de Atividade (contêm apenas um verbo) mais o número de chaves (descritas por "+x", sendo "x" referente ao número de chaves usadas) necessárias para distinguir atividades que contenham o mesmo verbo.

Tabela 3.1 - Total de processos com referido número de casos, eventos e atividades mais chave relevante.

Processos	Número de Casos	Número de Eventos	Número de Atividade + chave relevante
Processo_1_meeting	19	74	+4 (5)
Processo_2_table	4	24	+4 (5)
Processo_3_appintegration	19	42	+4 (5)
Processo_4_supplierappclass	34	188	+3 (4)
Processo_5_commonserviceapp	1	23	+1 (2)
Processo_6_mailapp	5	32	+1 (2)
Processo_7_projectlist	13	42	+3 (4)
Processo_8_cards	4	23	+3 (4)
Processo_9_useraccess	4	13	+4 (5)
Processo_10_course	2	7	+2 (3)
Processo_11_docfunction	5	30	+4 (5)
Processo_12_security	1	13	+3 (4)
Processo_13_webservice	10	45	+3 (4)
Processo_14_upload	3	15	+4 (5)
Processo_15_msgmaintenance	1	19	+2 (3)
Processo_16_campaign	3	26	+3 (4)
Processo_17_hertasks	1	3	+4 (5)

Analisando a Tabela 3.1, juntando a cada verbo de uma atividade, em média foi necessário usar três chaves para cada processo. Notou-se que em alguns processos usar apenas mais uma ou duas chaves a um verbo, não seria o suficiente para distinguir cada atividade pelo facto de muitas das vezes as chaves juntas também serem iguais e criar uma descrição da atividade praticamente idêntica. Referir que as chaves, em alguns processos, são divididas até oito colunas diferentes (*SubjectKeywords 1, SubjectKeywords 2, SubjectKeywords 3, …, SubjectKeywords 8)* onde cada uma tem apenas uma palavra. No momento que "x" chaves conseguem distinguir todas as atividades com o mesmo verbo, não é necessário usar todas (por exemplo as oito chaves que certos processos chegam a ter, mas que não chegam a ser necessário para o estudo).

Além da análise sobre as atividades e chaves usadas, nota-se que existe uma diferença de tamanhos entre processos no que se refere ao número de casos e número de eventos. Sabendo que existe 17 processos, foi decidido agrupar metade em grupos de três consoante o seu tamanho. Na Tabela 3.2 é possível visualizar os processos escolhidos apenas através do seu número de casos e número de eventos, sem ter em conta o conteúdo a que se refere cada processo.

Tabela 3.2 - Processos resumidos pelo seu tamanho em termos de casos e eventos.

Tamanho	Casos	Eventos	Casos/Eventos
Pequeno	Processo_2_table Processo_8_cards Processo_9_useraccess Processo_16_campaign	Processo_2_table Processo_8_cards Processo_16_campaign	Processo_2_table Processo_8_cards Processo_16_campaign
Médio	Processo_6_mailapp Processo_7_projectlist Processo_11_docfunction Processo_13_webservice	Processo_3_appintegration Processo_6_mailapp Processo_7_projectlist Processo_11_docfunction	Processo_7_projectlist Processo_11_docfunction Processo_13_webservice
Grande	Processo_1_meeting Processo_3_appintegration Processo_4_supplierappclass	Processo_1_meeting Processo_4_supplierappclass Processo_13_webservice	Processo_1_meeting Processo_3_appintegration Processo_4_supplierappclass

Deste modo é possível visualizar três processos agrupados na coluna "Casos/Eventos", uma decisão para no fundo ver o comportamento de cada algoritmo com um certo número de casos e eventos e também ter um número mais pequeno de processos aquando da análise dos resultados. A decisão dos três processos escolhidos em cada grupo foi feita a olho, não colocando um limite inferior ou superior entre o número de casos e eventos para cada tipo de tamanho. É de referir que foi decidido que processos de dois ou menos casos não foram contabilizados por serem simples e pequenos demais.

Concluindo, todo este procedimento de divisão e agrupamento serve para facilitar toda a análise dos processos que é necessária para tirar resultados e conclusões.

3.3 SELEÇÃO DOS ALGORITMOS

A partir da secção 2.7, onde são descritos alguns dos principais algoritmos, são utilizados para análise os algoritmos *Alpha*, *Heuristic Miner*, *Genetic Miner* e *Fuzzy Miner*. Todos estes algoritmos, exceto o *Fuzzy Miner*, os seus modelos de processo gerados podem ser convertidos para outras notações. Estes apresentam caraterísticas que podem ser apropriados à estrutura dos *event logs* e são muito utilizados na área de *Process Mining*. O algoritmo *Fuzzy Miner* é considerado um dos algoritmos mais dominadores dado que apresenta melhoramentos perante os algoritmos anteriores. A desvantagem deste, como já mencionada na Tabela 4, o seu output final é um modelo *Fuzzy* e este não pode ser convertido para outros tipos de notações. A conversão para uma notação *Petri net* é necessária na etapa dos resultados porque só desta forma é possível avaliar os processos a partir das medidas de qualidade já mencionadas, como *Fitness*, *Precision*, *Simplicity* e *Generalization*. A única medida de qualidade que de momento o modelo *Fuzzy* consegue calcular é o *Fitness*, sendo só essa medida possível de comparar, entre este e os restantes algoritmos.

É de notar que existe mais algoritmos nesta área de *Process Mining* que não são utilizados por duas razões: (1) não foram detetados no *ProM* e/ou (2) a conversão para a notação *Petri net* não foi obtida para poder usar a análise de *Conformance Checker*, que é a análise disponível no *ProM* 5.2 de modo a calcular as medidas de qualidade.

3.4 CONFIGURAÇÃO DOS DADOS NO FRAMEWORK

Concluída toda a organização dos dados da subsecção acima teve como próximo passo tratar do *Process Mining* dentro de uma ferramenta. Das ferramentas estudadas na secção 2.4, a escolhida foi o *ProM*. Além de ser uma ferramenta *open-source*, ela dispõe de várias técnicas de mineração e análise importantes e necessárias para o estudo de *Process Mining*. É usado duas versões do *ProM*, versão 5.2 e versão 6.9 (até ao momento da escrita desta tese é a última versão da ferramenta). Em 2010, segundo (Gunther, 2010) este transmite algumas das razões pelo facto da versão 5.2 ser uma melhor escolha para o

estudo de *Process Mining*. Uma das razões é o facto do *ProM* 5.2 ser o último de uma série de iterações ao longo de códigos com mais de seis anos. Ele teve inúmeras pessoas examinando-o, aplicando-o em projetos industriais e cursos universitários e corrigindo muitos erros ao longo do caminho, enquanto o *ProM* 6 e suas subversões serem ainda toda uma nova reestruturação da ferramenta, sendo suscetível a erros. Como a estrutura do *ProM 6* foi completamente redesenhada, isso significa que todos os *plugins* também precisam ser reescritos para trabalhar com a nova estrutura. Apesar de já terem passado nove anos, testando uma versão e outra, a 5.2 gerou uma melhor compreensão e simplicidade para o que era necessário.

Ao ser decidido usar o *ProM 5.2*, não foi posto de lado numa primeira fase o uso do *ProM 6.9*. Refere-se o facto da versão 5.2 só suportar ficheiros MXML como único formato, enquanto a versão 6.9 consegue suportar também formato XES. Para ter disponível os ficheiros em formato MXML e ser possível abrir na versão 5.2, algumas etapas de configuração foram feitas. Primeiro, todos os ficheiros dos processos que estavam guardados em formato XLSX foram guardados em formato CSV simplesmente através do software *Excell*, isto porque só é possível ter ficheiros em formato XES através de uma conversão existente no *ProM 6.9*. Pode ser visualizado através da Figura 3.4, depois do ficheiro importado, esta primeira etapa.

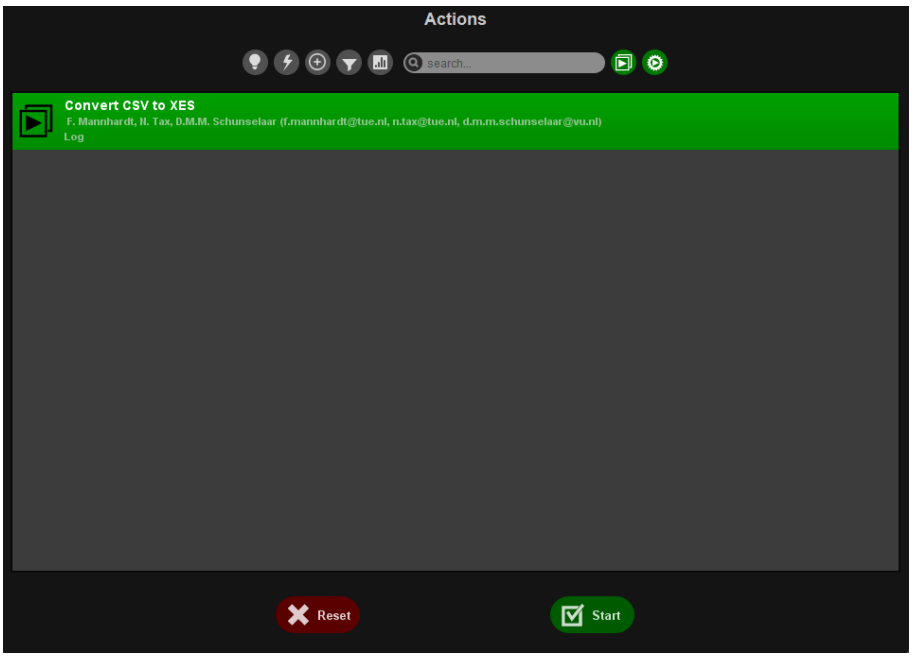


Figura 3.4 - Plugin de conversão do formato CSV para XES.

Segundo, momentos antes de concluir a conversão é necessário configurar o ficheiro selecionando apenas a coluna do *Case*, *Activity* (neste caso as *keywords* também são necessárias) e *TimeStamp* (não está presente em nenhum processo, logo não é inserida nenhuma coluna). Esta etapa é vista a partir da Figura 3.5, sendo que a partir daqui o ficheiro fica convertido em formato XES e assim pode ser guardado em formato MXML.

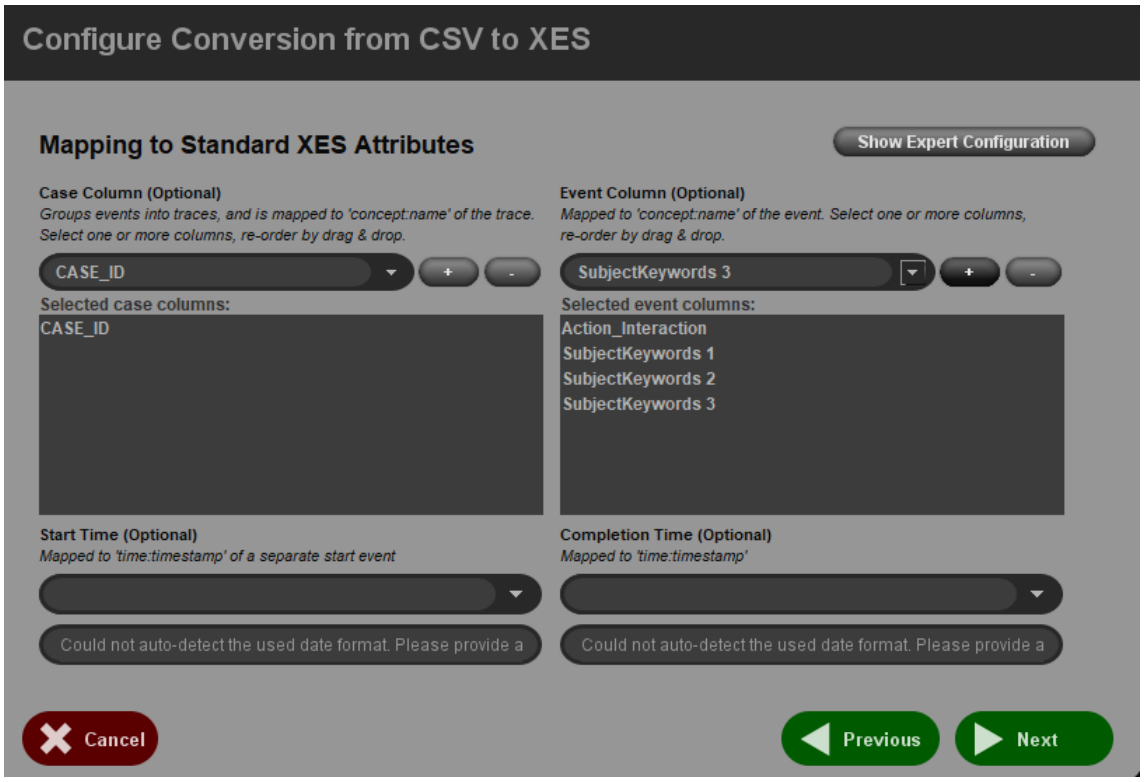


Figura 3.5 - Configuração do ficheiro de processos em formato XES.

Por fim, é possível abrir ficheiros prontos e configurados em formato MXML na versão 5.2 do *ProM*, apenas abrindo o ficheiro na pasta onde este foi guardado. Na figura 3.6 é possível visualizar um dos ficheiros de processo aberto e pronto a ser analisado no *ProM* 5.2.

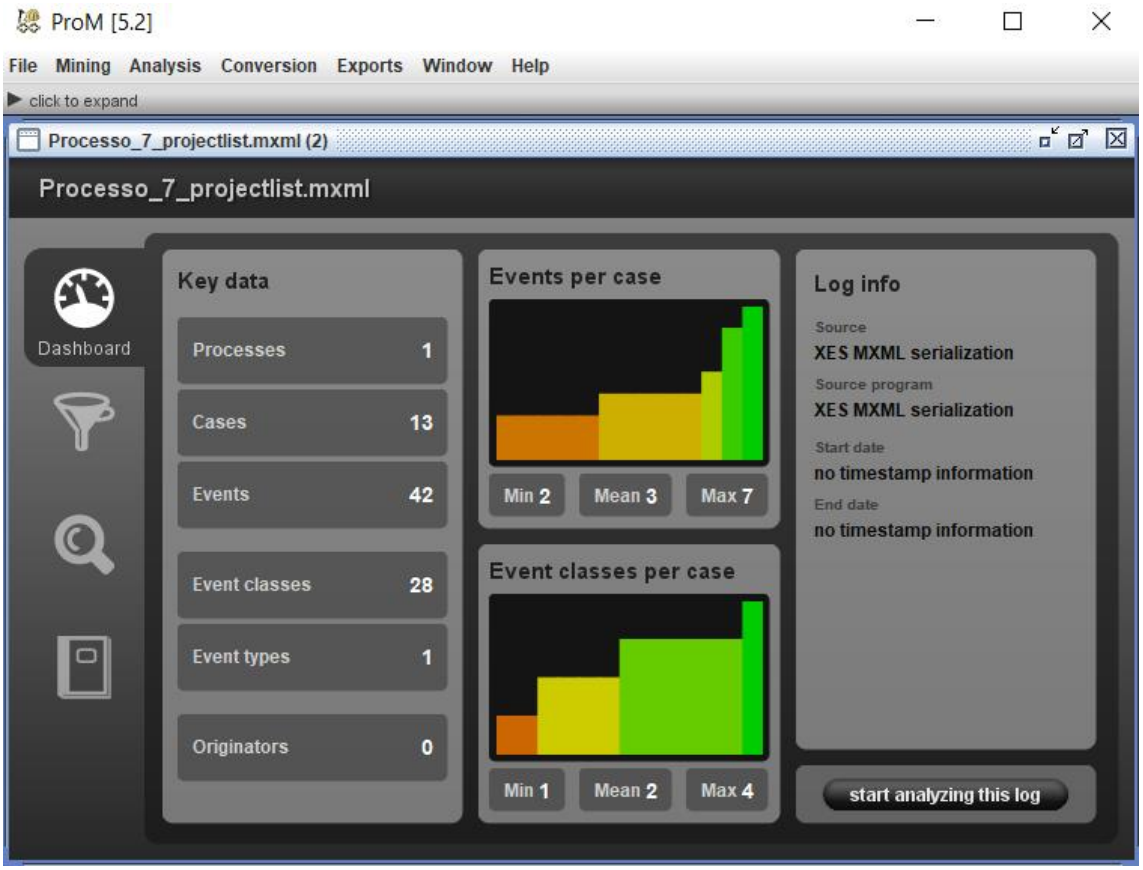


Figura 3.6 - Ficheiro de processo em formato MXML no ProM 5.2.

Todos os restantes ficheiros usados passam pelo mesmo processo para poderem ser analisados da mesma maneira.

3.5 CONFIGURAÇÃO DOS ALGORITMOS NO FRAMEWORK

Nesta secção pode-se visualizar todas as configurações que o *ProM* disponibiliza por natureza de cada algoritmo que se deseja utilizar, para analisar um *log*. Cada algoritmo tem a sua própria configuração e como o objetivo é apenas a descoberta de modelos de cada processo criado, os *logs* são analisados com a configuração padrão. Como já mencionado, os algoritmos a utilizar são: *Alpha Miner*, *Heuristic Miner* e *Genetic Miner*.

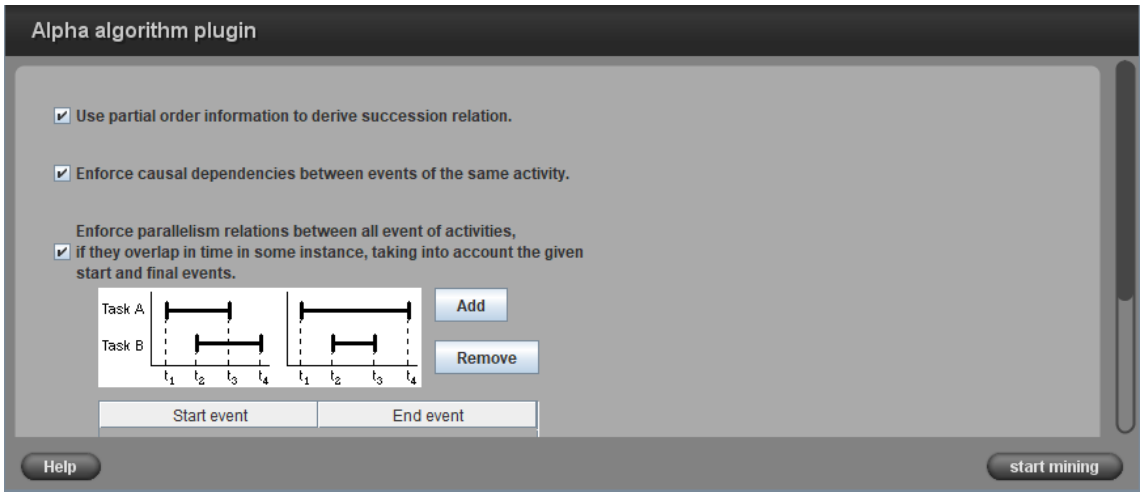


Figura 3.7 - Configuração presente no Alpha Miner.

O *Alpha Miner* apresenta uma configuração, que pode ser visualizado na Figura 3.7 onde tem selecionado três particularidades antes de partir para a análise, sendo elas:

(i) usar informações de ordem parcial para derivar a relação de sucessão;

(ii) impor dependências causais entre eventos da mesma atividade;

(ii) impor relações de paralelismo entre todos os eventos de atividades, se elas se sobrepuserem no tempo em algum instante, levando em consideração os eventos inicial e final.

No caso do *Heuristic Miner* este algoritmo concentra-se no cálculo da frequência de dependência e traços de eventos na construção de um modelo de processo. Para criar um modelo de processo, os *event logs* precisam ser analisados com base nos valores de dependência das atividades. Segundo (Kurniati, Kusuma & Wisudiawan, 2016; Gupta, 2007) e visualizando as configurações presentes neste *framework* para o *Heuristic Miner*, na Figura, ao construir um modelo de dependência, podemos definir alguns limites como:

(i) ***Dependency threshold***: valor mínimo de dependência entre eventos;

(ii) ***Positive observations***: valor mínimo da frequência de dependência entre os eventos;

(iii) ***Relative-to-best threshold***: valor mínimo da diferença entre o valor de dependência do evento e o valor máximo de dependência;

(iv) ***Length-one-loops threshold***: valor mínimo da mesma dependência de evento;

(v) ***Length-two-loops threshold***: valor mínimo da dependência do evento do par em *loop*;

(vi) ***Dependency divisor***: O divisor de dependência tem um valor padrão de 1 porque 1 é um número pequeno que pode afetar pequenos *logs* de forma significativa e ao mesmo tempo tem um efeito menos significativo nos *logs* grandes;

(vii) ***Long distance threshold***: O valor do parâmetro limite de longa distância especifica quais dependências de longa distância aceita ou rejeita. Se o valor da medida de dependência de longa distância for menor que o valor do limite de longa distância, a dependência será rejeitada;

(viii) ***AND threshold***: Este parâmetro indica que duas atividades em um *log* estão em paralelo se o valor da medida AND calculado for maior do que o valor especificado para o limite AND.

Em seguida o *Genetic Miner* usa métodos adaptativos que tentam imitar o processo de evolução, começando com uma população inicial de indivíduos, onde cada indivíduo apresenta uma medida de *fitness* para indicar a sua qualidade. Basicamente um indivíduo é um possível modelo de processo e o *fitness* é uma função que avalia quão bem o indivíduo é capaz de reproduzir o comportamento no *log*. As populações evoluem selecionando os indivíduos mais aptos e gerando novos indivíduos usando operadores genéticos, como *crossover* (combinando partes de dois ou mais indivíduos) e mutação (modificação aleatória de um indivíduo). Com a opção de ajuda presente no *ProM* e através de (Medeiros, 2006) é possível obter resumidamente informações específicas sobre cada parâmetro deste algoritmo. Sendo assim, visualizando a Figura, os parâmetros a definir são:

(i) ***Population size***: define o número de pessoas que serão usadas durante a pesquisa;

(ii) ***Initial population type***: define como a população inicial deve ser construída;

(iii) ***Maximum number population***: define o número máximo "n" vezes que o algoritmo genético pode repetir. Este parâmetro está relacionado aos critérios de interrupção. Resumidamente, o *plug-in* do *Genetic Algorithm* é interrompido quando (a) encontra um indivíduo cujo *fitness* é máximo, (b) repete "n" vezes ou (c) o indivíduo mais apto não foi alterado por n/2 iterações seguidas;

(iv) ***Seed***: define a semente usada para gerar o número aleatório para este *plug-in* (ponto de partida da geração);

(v) ***Power value***: define o valor de energia usado pelas heurísticas para construir a população inicial. Como já explicado, as heurísticas funcionam por dependência entre duas tarefas: quanto mais uma tarefa **t1** é seguida diretamente por uma tarefa **t2**, maior a probabilidade de os indivíduos serem construídos com uma dependência de **t1** a **t2**. O valor da energia é usado para controlar a "influência" das heurísticas na probabilidade de definir uma dependência entre duas tarefas. Valores mais altos para o valor de energia levam à dedução de menos dependências entre duas tarefas nos *event logs* e vice-versa;

(vi) ***Elitism rate***: define a percentagem dos indivíduos mais aptos da geração atual que serão copiados para a próxima geração. Por exemplo, uma taxa de elitismo de 0,02 significa que 2% dos melhores indivíduos da população são copiados para a próxima população;

(vii) ***Fitness type***: define o tipo de *fitness* que o *plug-in* do *Genetic Algorithm* usa para avaliar a qualidade de um indivíduo. Os tipos existentes são o *ProperCompletion*, *StopSemantics*, *ContinuousSemantics*, *ImprovedContinuousSemantics* e *ExtraBehaviorPunishment*. Cada tipo tem uma fórmula matemática para ser calculado, sendo que o *ExtraBehaviorPunishment* é extremamente utilizado como padrão para análise de um modelo, pelo facto de este incorporar os conceitos dos outros tipos de *fitness*.

Com os primeiros parâmetros definidos, de seguida a opção *use genetic operators* é utilizada como padrão. Define se os operadores genéticos de *crossover* e mutação serão usados para construir as populações que seguem a população inicial. Se esta opção estiver desmarcada, as próximas populações serão construídas exatamente como a população inicial. Se esta opção estiver marcada, os indivíduos da próxima população que não pertencem à elite serão criados aplicando *crossover* e mutação a indivíduos da população atual. O funcionamento é o seguinte:

(a) os dois primeiros pais são selecionados e em seguida passam por *crossover* (com probabilidade padrão de 0.8), produzindo dois filhos;

(b) toda a descendência pode sofrer mutação (com probabilidade padrão de 0.2);

(c) finalmente os filhos são inseridos na nova população.

Através desta opção marcada, é ainda disponibilizado mais cinco parâmetros:

(1) ***Selection method type***: define como os pais dos operadores genéticos serão escolhidos. Ambos os métodos são baseados em um *Tournament*. O tipo de método *Tournament* funciona selecionando aleatoriamente dois indivíduos na população e retorna o indivíduo mais apto em 75% das vezes, e o indivíduo menos apto em 25% das vezes. O tipo de método *Tournament5* (usado como padrão) seleciona aleatoriamente cinco indivíduos na população e sempre retorna o indivíduo mais apto;

(2) ***Crossover type***: define como dois pais (indivíduos selecionados) serão recombinados. Todos os tipos funcionam no nível da tarefa onde cada uma tem um conjunto de entrada e saída. Portanto, alguns tipos de *crossover* funcionam nos conjuntos completos de entrada / saída, outros em subconjuntos que estão nos conjuntos de entrada / saída de uma tarefa. Os tipos de *crossover* que estão presentes para escolha são:

(2a) ***Local One Point***: sempre troca os conjuntos de entrada e saída de uma tarefa;

(2b) ***Variable Local One Point*:** onde troca os conjuntos de entrada / saída em 50% das vezes. Outras vezes, apenas um desses conjuntos é trocado;

(2c) ***Fine Granularity***: troca subconjuntos no conjunto de entrada / saída de uma tarefa. Se os subconjuntos que foram trocados tiverem elementos de interseção com subconjuntos (no outro indivíduo) que não foram trocados, esse tipo de cruzamento vai escolher aleatoriamente se esses subconjuntos (os trocados e os não trocados) vão ser misturados ou se os elementos comuns são removidos dos subconjuntos que não foram trocados durante o *crossover*;

(2d) ***Enhanced***: troca subconjuntos no conjunto de entrada / saída de uma tarefa. Com igual probabilidade, os subconjuntos trocados podem ser incluídos no conjunto de entrada / saída, misturados com alguns subconjuntos não trocados nos conjuntos de entrada / saída, ou adicionar os conjuntos trocados e remover de alguns subconjuntos não trocados elementos de interseção que estão nos trocados. É recomendado o uso do tipo de crossover "*Enhanced*" (tipo padrão) porque ele incorpora os conceitos dos outros tipos de *crossover*;

(3) ***Crossover rate***: define a probabilidade de os dois pais serem recombinados para criar dois filhos para a próxima geração. Se a probabilidade for igual a 0, então, após o *crossover*, os filhos serão iguais aos pais. Como padrão o valor está definido como 0.8;

(4) ***Mutation type***: define como um indivíduo será modificado aleatoriamente, sendo o ponto de mutação uma tarefa. Como o *crossover*, também existem tipos de mutação e funcionam da seguinte maneira:

(4a) ***All Elements***: adiciona ou remove um elemento de um subconjunto no conjunto de entrada / saída da tarefa mutada;

(4b) ***Partition Redefinition***: vai reorganizar os subconjuntos do conjunto de entrada / saída da tarefa mutada;

(4c) ***Enhanced***: executa uma das seguintes operações com igual probabilidade: adicionar uma tarefa a um subconjunto no conjunto de entrada / saída da tarefa mutada, remover uma tarefa de um subconjunto no conjunto de entrada / saída de um conjunto mutado e reorganize os subconjuntos do conjunto de entrada / saída da tarefa mutada;

(5) ***Mutation rate***: define a probabilidade de um indivíduo levar com mutação.

Com as configurações presentes de cada algoritmo, o *Fuzzy Miner* não será utilizado para os resultados pelo facto de não ser possível converter o seu modelo para um modelo *Petri net*, concluindo a não possibilidade de comparar as qualidades de conformidade com os restantes algoritmos.

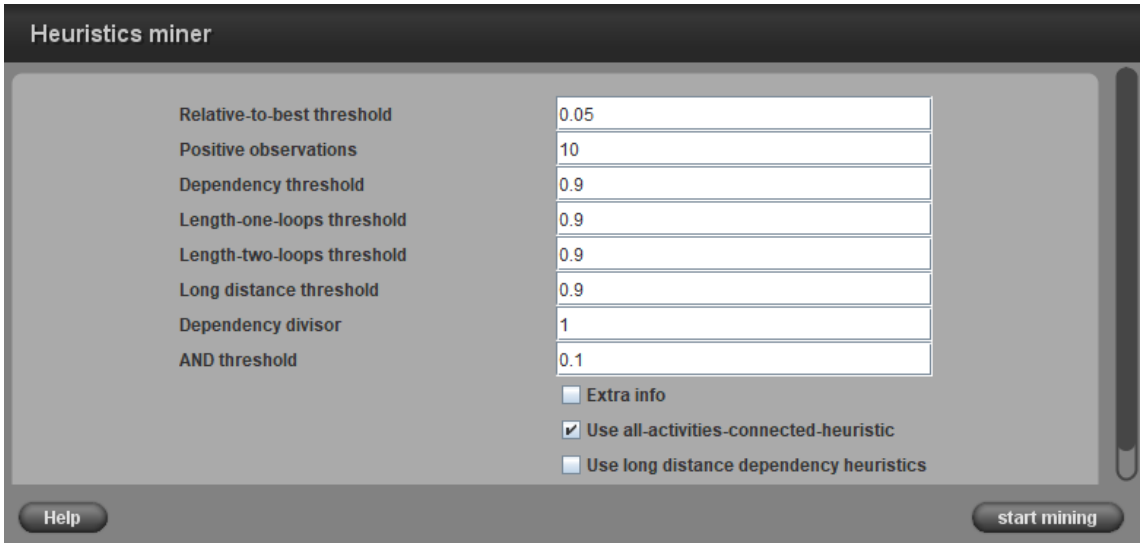


Figura 3.8 - Configuração presente no *Heuristic Miner*.

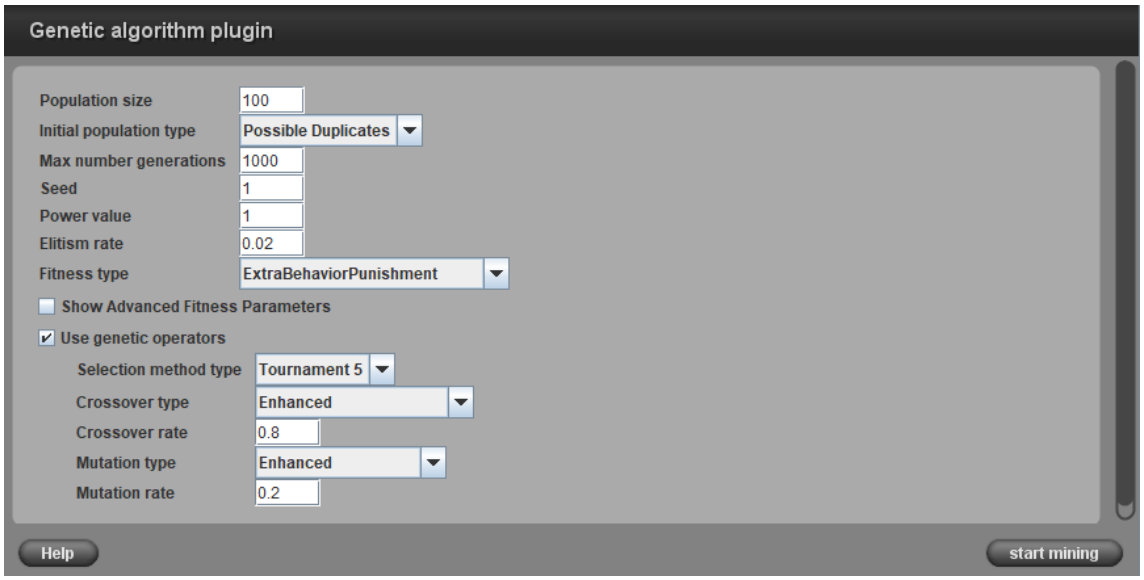


Figura 3.9 - Configuração presente no *Genetic Algorithm*.

CAPÍTULO 4

ANÁLISE DE RESULTADOS

4.1 DESCOBERTA DE MODELO

Nesta seção é calculado o modelo de cada algoritmo que analisa os *logs*. Os *logs* são os processos apresentados na Tabela 3.2 da seção 3.3. Nem todos os processos dessa tabela vão ser analisados, onde apenas é utilizado um de cada tamanho, sendo que os modelos de cada processo não apresentados estão em Anexo.

4.1.1 PROCESSOS

Os processos estão divididos em três tamanhos, sendo eles, pequeno, médio e grande. Com as configurações de cada algoritmo definidas na seção 3.5, os modelos gerados para cada algoritmo estão ilustrados nas Figuras 4.1, 4.2, 4.3 e 4.4, a começar no *Alpha Algorithm*, prosseguindo com o *Heuristic Miner* e *Genetic Miner*. O *Fuzzy Miner* não será utilizado para uma comparação geral de conformidade porque só apresenta uma dimensão de qualidade: *fitness*.

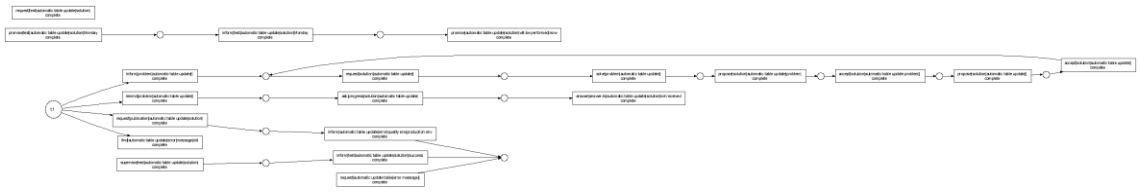


Figura 4.1 - Modelo do Processo_2_table gerado pelo *Alpha Miner*.

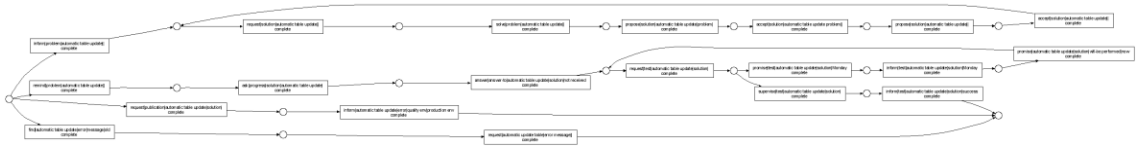


Figura 4.2 - Modelo do Processo_2_table gerado pelo *Heuristic Miner*.

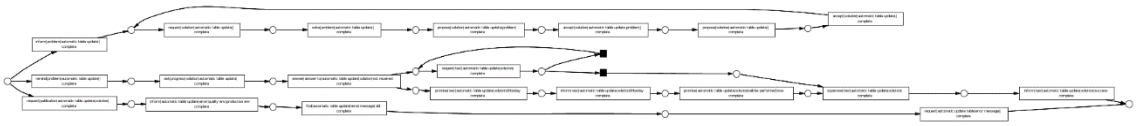


Figura 4.3 - Modelo do Processo_2_table gerado pelo *Genetic Miner*.

No caso do *Heuristic Miner* e do *Genetic Miner* os seus modelos gerados foram convertidos em *Petri Net* para análise de conformidade. Só é possível esta análise se os *logs* estiverem modelados dessa forma. O *Fuzzy Miner* tem o seu próprio modelo (*Fuzzy model*) que não permite converter para outro tipo de modelação, ao contrário dos restantes algoritmos.

4.2 VERIFICAÇÃO DE CONFORMIDADE

A partir do *ProM 5.2* a análise dos processos é feita a partir de *Conformance Checker*, em que os modelos gerados têm obrigatoriamente estar convertidos em *Petri net*, porque caso contrário não é possível ter presente esta análise. Este tipo de análise demonstra até que ponto existe uma boa conformidade entre o *log* e o modelo construído. A verificação de conformidade é colocada no contexto mais amplo das técnicas de *Process Mining*. Enquanto o objetivo da descoberta é a extração automática de um modelo de processo a partir dos dados do *log*, a verificação de conformidade preocupa-se com a comparação de um modelo de processo existente e um *log* correspondente (van der Aalst & Rozinat, 2008). Para a verificação de conformidade é necessário o cálculo de diferentes tipos de medidas de qualidade já explicadas na secção 2.3. Dentro da análise do *Conformance Checker* presente no *ProM 5.2* é possível o cálculo de *Fitness*, *Precision* e *Simplicity*. A métrica de qualidade *Generelization* não está presente no *ProM 5.2*, logo não é possível o seu cálculo. A partir da Figura 4.5 é possível visualizar as opções padrão para o cálculo das métricas de qualidade na verificação de conformidade do *ProM*. A configuração padrão são normalmente as mais adequadas para uma boa análise.

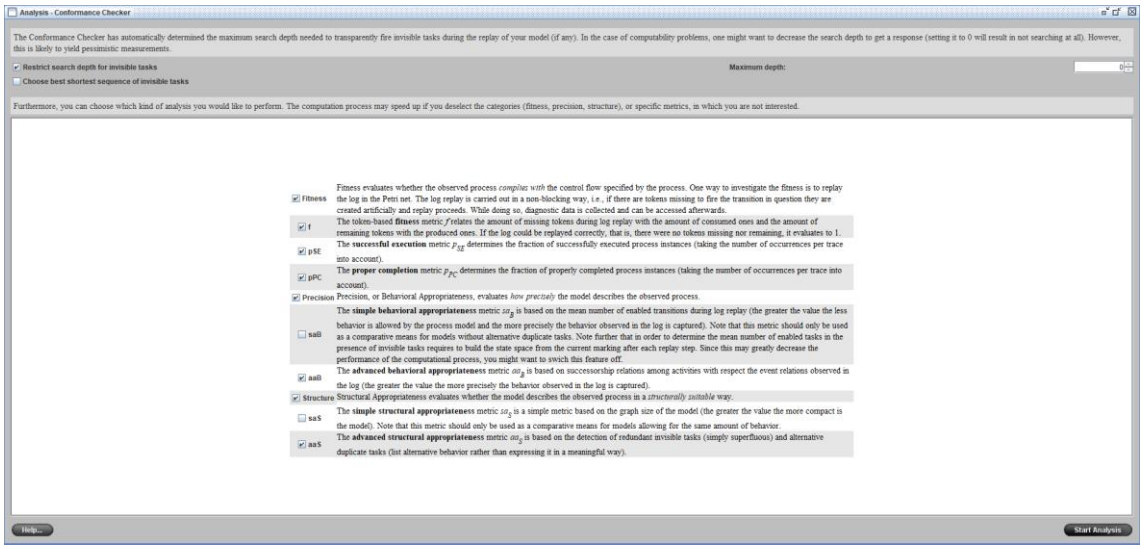


Figura 4.4 - Análise de verificação de conformidade no *ProM* 5.2.

4.2.1 MÉTRICAS DE QUALIDADE

Atualmente, várias métricas que medem *Fitness*, *Precision*, *Simplicity* e *Generalization* de um modelo descoberto são implementadas. As métricas são importantes para poder analisar o comportamento entre um modelo descoberto e um *log*. A partir do *ProM* é possível calcular essas métricas desde que o modelo esteja convertido em uma *Petri Net* (van der Aalst & Rozinat, 2008). A primeira métrica, referente a *Fitness*, é baseada num *token* f que relaciona: (1) a quantidade de *tokens* ausentes com a de consumidos e (2) a quantidade de *tokens* restantes com a de produzidos. Se o *log* puder ser reproduzido corretamente, sem ausência ou restos de *tokens* "perdidos", o seu valor será 1, caso contrário a medida tende a ser avaliada como 0.

Para *Precision*, existem duas métricas diferentes, sendo elas, *Simple Behavioral Appropriateness* e *Advanced Behavioral Appropriateness*. A primeiras das duas formas é calculado com o número médio de transições durante a reprodução do *log*. Quanto mais perto do valor 1, menos comportamento é permitido pelo modelo e ao mesmo tempo será mais preciso (van der Aalst & Rozinat, 2008). Não é aplicado quando o modelo é sequencial através de tarefas duplicadas e tem como limitação a presença de *loops* e tarefas duplicadas alternativas. A segunda forma tem como base analisar mais ao pormenor as relações de atividades entre o *log* e o modelo. Esta métrica é usada como uma medida absoluta e não apenas como uma medida comparativa (caso da métrica *Simple Behavioral Appropriateness*). Existe um ponto ótimo (valor 1) que está

relacionado entre *Precision* e *Generalization* levando à integridade do *log* de eventos. Isto quer dizer que com a presença de *Generalization* podem existir caminhos que não estão presentes no *log* onde a distância entre as atividades é determinada globalmente (as atividades entre si não precisam seguir ou preceder diretamente). Existe assim uma relação de "Segue" e "Precede" entre as atividades, estando dividida em quatro casos possíveis (Rozinat, 2010; van der Aalst & Rozinat, 2008; Fahland et al., 2017):

(i) **Sempre precede**: visualiza as atividades que sempre se precedem no *log*, mas algumas vezes só se precedem de acordo com o modelo;

(ii) **Nunca precede**: visualiza as atividades que nunca se precederam no *log*, mas algumas vezes se precedem no modelo;

(iii) **Sempre segue**: visualiza as atividades que sempre se seguiram no *log*, mas apenas algumas vezes se seguem nas relações de modelo

(iv) **Nunca segue** - visualiza as atividades que nunca se seguiram no *log*, mas apenas algumas vezes se seguem no modelo. Se num modelo a soma das relações de "Algumas vezes" existirem numa mesma quantidade de "Segue" e "Precede", ou se o *log* conter apenas um caminho (mas o modelo permitir mais), *Precision* será avaliado em 0, levando a uma precisão indefinida.

Tem como limitação as atividades que fazem parte de um *loop* maior, não serem refletidas pelas relações "Sempre" e "Nunca", mas sim pelas relações "Às vezes" porque o processo pode voltar a esse ponto no modelo ou não. Sendo assim, relações globais podem ser benéficas para caracterizar melhor o comportamento em *loops*, até certo ponto. Além da limitação descrita anteriormente, tarefas duplicadas alternativas dentro de um *loop* não serão detetadas e modelos grandes vão necessitar de uma exploração muito exaustiva (Rozinat, 2010; Fahland et al., 2017).

Por fim para *Simplicity* também está dividida em duas métricas diferentes de cálculo no *ProM* (ou Struture), sendo elas, *Simple Structural Appropriateness* e *Advanced Structural Appropriateness*. A primeira forma é uma medida simples com base no tamanho do gráfico (quanto maior o valor, mais compacto é o modelo), onde deve ser usada apenas como um meio comparativo para modelos que permitem a mesma quantidade de comportamento (van der Aalst & Rozinat, 2008). Já a segunda forma para que o seu valor seja maior não pode existir no modelo tarefas duplicadas alternativas (que

nunca ocorrem na mesma sequência de execução) e tarefas invisíveis redundantes (tarefas que ao serem removidas do modelo, não alteram o comportamento deste). Se nenhum deste tipo de tarefas existirem no modelo, então o valor de *Simplicity* é 1. Tem em comum com a métrica *Advanced Behavioral Appropriateness* a limitação de tarefas duplicadas dentro de um *loop* não serem detetadas (Rozinat, 2010).

4.3 RESULTADOS

Para facilitar e simplificar a interpretação dos resultados, a partir da Tabela 4.1 a cada algoritmo e *log* foi utilizado um termo específico.

Tabela 4.1 - Simplificação do nome de cada algoritmo e *log* com um termo específico.

Nome Original	Termo Atribuído
Alpha Algorithm	M1
Heuristic Miner	M2
Genetic Miner	M3
Processo_2_table	L1
Processo_8_cards	L2
Processo_16_campaign	L3
Processo_7_projectlist	L4
Processo_11_docfunction	L5
Processo_13_webservice	L6
Processo_1_meeting	L7
Processo_3_appintegration	L8
Processo_4_supplierappclass	L9

Tabela 4.2 – Resultados de *Fitness* e *Precision* Simples/Avançada entre os *logs* e modelos utilizados.

		Pequeno			Médio			Grande		
		L1	L2	L3	L4	L5	L6	L7	L8	L9
M1	*f*	0.9317	1.0	0.9365	0.8947	0.8824	0.9778	0.7821	0.5488	0.6482
	Ab	0.7628	0.8472	0.8619	0.6163	0.6179	0.7739	0.4539	0.7256	0.8082
	A'b	0.0	0.0	0.0	0.0312	0.0312	0.0	0.0	1.0	0.0
M2	*f*	0.8928	0.9231	0.8965	0.9153	0.8000	0.9454	0.8217	0.6301	0.9043
	Ab	0.9454	0.9506	0.9816	0.8112	0.9257	0.9275	0.7678	0.6899	0.8699
	A'b	0.4664	1.0	1.0	0.7915	0.5871	1.0	0.8871	0.5799	0.5864
M3	*f*	0.7931	0.9629	0.9687	0.9180	0.7912	0.9210	0.8155	0.7671	0.7993
	Ab	0.9430	0.9551	0.9816	0.8112	0.9222	0.9151	0.7609	0.7656	0.7461
	A'b	0.4700	0.7215	0.6582	0.7915	0.6527	0.6964	0.8558	0.8440	0.7825

Com a Tabela 4.2 é possível encontrar todos os valores de cada métrica calculada a partir da análise de conformidade presente no *ProM*, quando o modelo primeiramente está convertido numa *Petri Net*. Para uma melhor discussão dos resultados, nas Figuras 4.5, 4.6 e 4.7, são apresentados os comportamentos de cada modelo com o *log* referente ao *Fitness* e *Precision*.

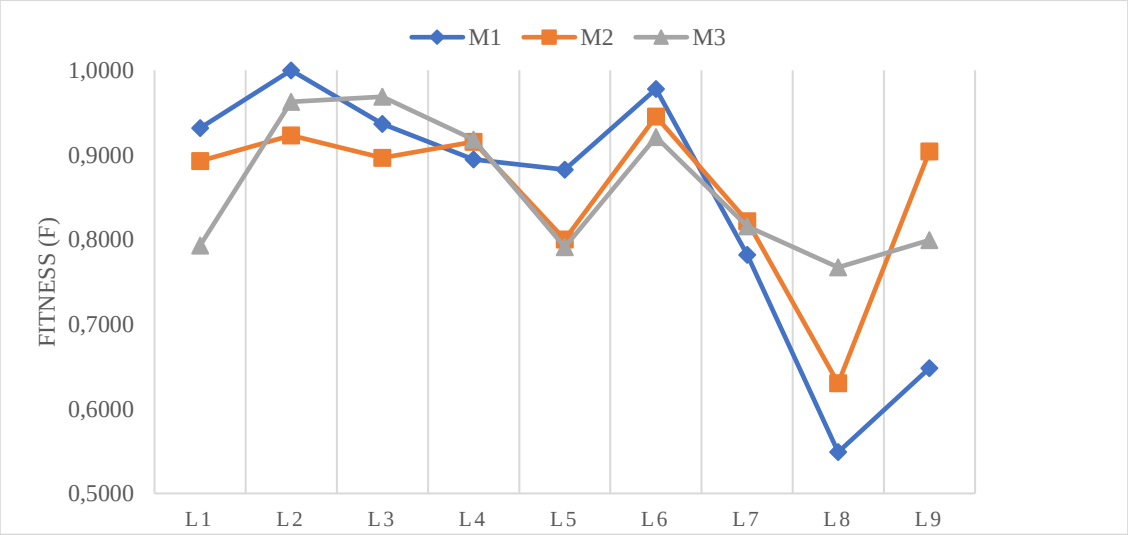


Figura 4.5 - Comportamento do modelo em relação ao *log* em termos de *Fitness*.

Visualizando a Figura 4.5 o *Alpha Miner* conseguiu no geral obter melhores valores de *Fitness* nos primeiros seis *logs* (L1 até L6) conseguindo o valor ótimo de 1.0 em L2. Nos restantes *logs* o seu valor baixou drasticamente pelo facto de estes terem mais dados, que podiam conter maior ruido e presença de mais tarefas duplicadas em relação aos anteriores. Os restantes algoritmos tiveram valores muito parecidos entre ambos sofrendo também uma diminuição de *fitness* nos últimos três logs. Concluindo, todos os algoritmos tiveram um comportamento bom nos primeiros seis *logs*, mas a partir de L7

os valores de *Fitness* baixaram, com *Alpha Miner* a ser o mais visado. Como teoricamente, este último apesar de acabar por ter resultados sensivelmente bons em termos de *Fitness*, acaba por não ser um algoritmo fiável para ser utilizado na prática e isso pode ser visto a seguir, nos resultados de *Precision*.

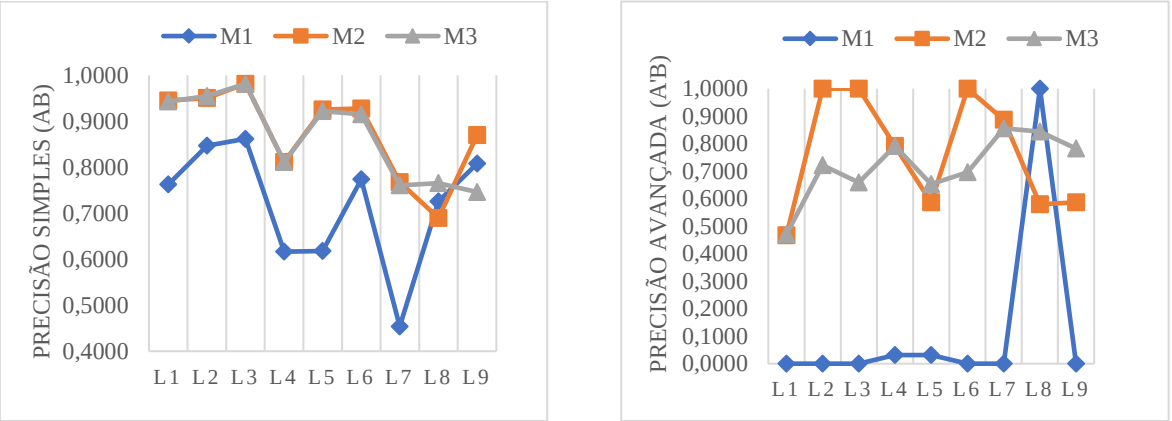


Figura 4.6 - Comportamento do modelo em relação ao *log* em termos de *Precision* Simples e Avançada.

A partir da Figura 4.6 é possível notar que na *Precision* Simples *Ab Heuristic Miner* e *Genetic Miner* têm ambos valores muito idênticos (ambos acabam por ter o mesmo valor em L3 e L4) com o *Heuristic Miner* a conseguir ter ligeiramente melhores resultados, onde apenas em L8 e L9 tiveram uma maior diferença. Para L8 tiveram uma diferença de 0.0757 com M3 a ter vantagem, mas para L9 foi M2 a conseguir ficar por cima por uma diferença de 0.1238. O *Alpha Miner* acaba por resultar em valores mais baixos, apenas tendo um resultado razoável (0.8082) em L9.

Na *Precision* Avançada *A'b* o algoritmo *Heuristic Miner* consegue por três vezes obter o valor ótimo de 1.0 em L2, L3 e L6. Além disso é o algoritmo que apresenta melhores resultados na maior parte dos *logs* presentes. Os algoritmos *Heuristic Miner* e *Genetic Miner* conseguem obter o mesmo valor (0.7915) em L4. Para esta métrica o resultado é praticamente nulo (valor 0) com o *Alpha Miner*, apesar de conseguir um valor ótimo de 1.0 em L8.

Concluindo, o *Alpha Miner* acabou por ter resultados muito baixos em ambas as métricas, acabando o *Heuristic Miner* e *Genetic Miner* obterem resultados mais satisfatórios, com o primeiro destes dois conseguir ser ligeiramente melhor. De notar também que os valores, para ambas as métricas, tenderam a baixar nos últimos três logs, como aconteceu com *Fitness*.

Tabela 4.3 – Resultados da Simplicity Simples/Avançada entre os *logs* e modelos utilizados.

		Pequeno			Médio			Grande		
		L1	L2	L3	L4	L5	L6	L7	L8	L9
M1	*As*	0.6470	0.6000	0.5952	0.7500	0.7143	0.6406	0.8611	0.6522	0.6242
	A's	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
M2	*As*	0.5946	0.5609	0.5555	0.5957	0.5556	0.5857	0.5581	0.6000	0.5621
	A's	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
M3	*As*	0.5358	0.5476	0.5102	0.5600	0.5405	0.5857	0.5217	0.6250	0.6050
	A's	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0

Na Tabela 4.3 é possível concluir que os valores para a Simplicity Simples *As* são muito idênticos entre todos os logs e modelos. Apenas no *Alpha Miner* os valores tiveram maior variação como o caso de L4, L5 e L7 que obtiveram resultados mais altos, entre 0.7100 e 0.8700, em comparação com os restantes que ficaram compreendidos entre 0.5100 e 0.6500. Na *Simplicity* Avançada *A's* todos os valores deram valor ótimo de 1.0 chegando à conclusão que por esta métrica não é possível avaliar qual dos algoritmos tiveram melhor comportamento pelo facto de terem todos o mesmo resultado.

O melhor modelo descoberto não pode ser escolhido apenas através de uma medida de qualidade, sendo importante ter o peso de pelo menos duas medidas, como *Fitness* e *Precision*, apesar de *Simplicity* também poder ser usada (no *ProM* 5.2 são as medidas possíveis de serem calculadas). Colocando um peso igual para cada medida, foi necessário somar os valores de pelo menos duas medidas calculadas (*Fitness* e Precision) e dividir pelo número de medidas utilizadas, resultando num valor médio que serviu para concluir qual seria o algoritmo apropriado para cada *log*.

No cálculo da Média entre as medidas de conformidade, a *Simplicity* Simples não foi posta em conta pelo facto de:

(i) apresentar valores muito baixos;
(ii) os valores entre todos os modelos e logs são muito idênticos;
(iii) neste caso não interfere na escolha do modelo para cada log;
(iv) das medidas de conformidade calculadas é colocada como menos importante para a decisão do modelo.

Em relação a *Simplicity* Avançada, tendo em conta que todos os algoritmos dão valor ótimo de 1.0, também não vai influenciar em nada a escolha do modelo, ficando

indefinido qual o melhor entre todos os modelos. Posto isto, a Média foi calculada em duas tabelas, numa o *Fitness* e *Precision* Simples e em outra o *Fitness* e *Precision* Avançada.

Tabela 4.4 – Resultados da Média Simples de *Fitness* e *Precision* Simples.

Média Simples	L1	L2	L3	L4	L5	L6	L7	L8	L9
M1	0.8298	0.9236	0.8992	0.7555	0.7501	0.8758	0.6180	0.6372	0.7282
M2	**0.9191**	0.9368	0.9390	0.8632	**0.8628**	**0.9364**	**0.7947**	0.6600	**0.8871**
M3	0.8680	**0.9590**	**0.9751**	**0.8646**	0.8567	0.9180	0.7882	**0.7663**	0.7727
Melhor desempenho	M2	M3	M3	M3	M2	M2	M2	M3	M2

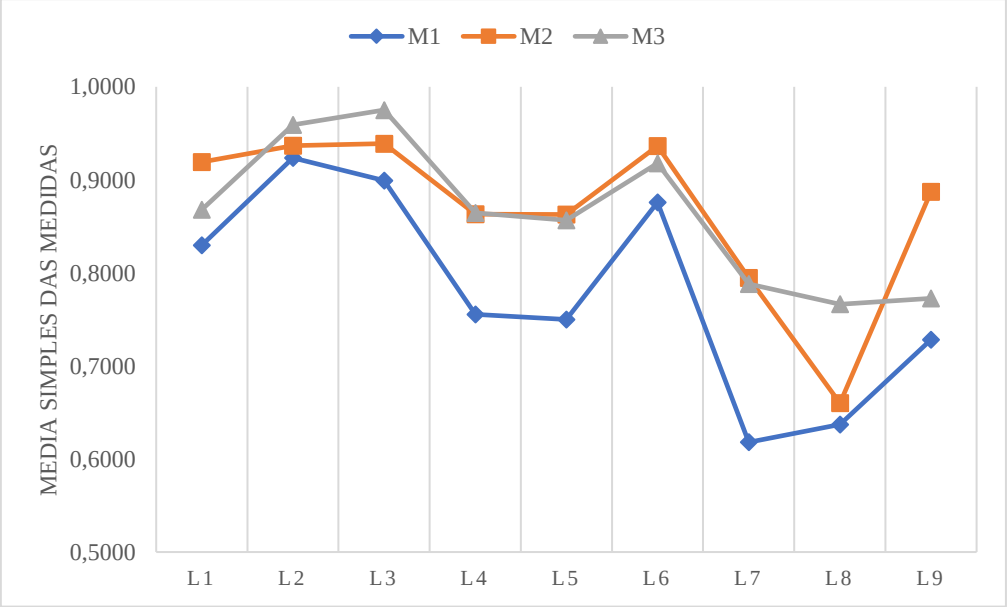


Figura 4.7 – Comportamento da Média Simples de *Fitness* e Precision Simples.

A partir da Tabela 4.4 foi possível construir o gráfico presente na Figura 4.7 onde denota melhor o comportamento da Média Simples de *Fitness* e *Precision* Simples entre os *logs* e os modelos gerados pelos algoritmos utilizados.

Tabela 4.5 – Resultados da Média Avançada de *Fitness* e *Precision* Avançada.

Média Avançada	L1	L2	L3	L4	L5	L6	L7	L8	L9
M1	0.4658	0.5000	0.4682	0.4629	0.4568	0.4889	0.3910	0.7744	0.3241
M2	**0.6796**	**0.9615**	**0.9482**	0.8534	0.6935	**0.9727**	**0.8544**	0.6050	0.7453
M3	0.6315	0.8422	0.8134	**0.8547**	**0.7219**	0.8087	0.8356	**0.8055**	**0.7909**
Melhor desempenho	M2	M2	M2	M3	M3	M2	M2	M3	M3

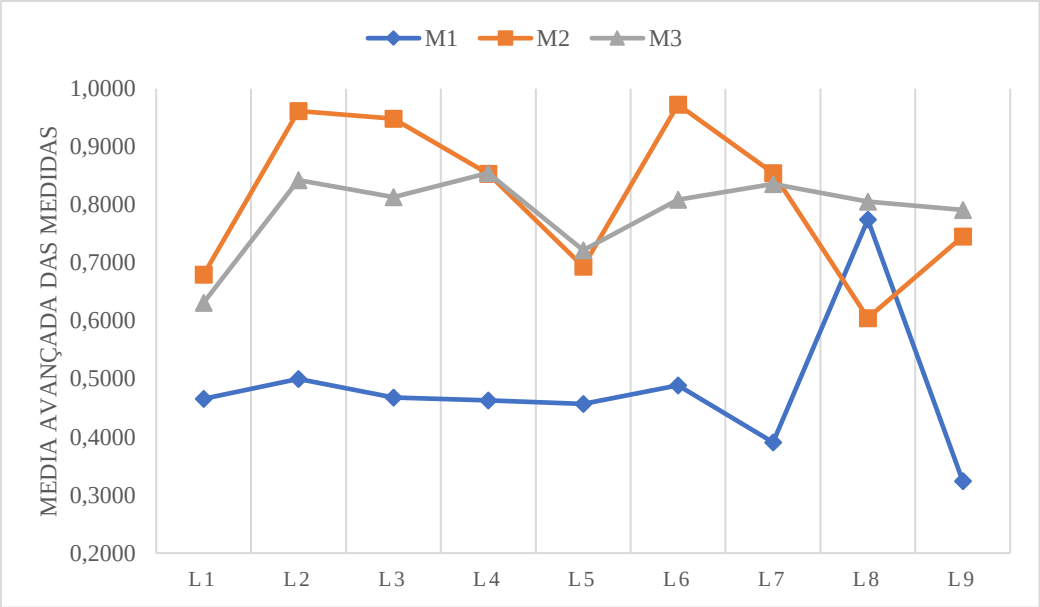


Figura 4.8 - Comportamento da Média Avançada de *Fitness* e *Precision* Avançada.

Com a Tabela 4.5 foi possível construir o gráfico presente na Figura 4.8 onde denota melhor o comportamento da Média Avançada de *Fitness* e *Precision* Avançada entre os *logs* e os modelos gerados pelos algoritmos utilizados.

O algoritmo que apresentou melhores resultados, em ambas as Médias, foi *Heuristic Miner* (conseguiu ter melhor resultado em cinco *logs*, no total de nove). Em segundo foi *Genetic Miner* (conseguiu ter melhor resultado em quatro logs) e por último, como já esperado, *Alpha Miner* não sendo escolhido para nenhum *log*. Apesar do deste último não ser escolhido para nenhum *log*, apresentou bons resultados de *Fitness* (em quatro ocasiões) e de *Simplicity* simples (nove vezes). Sabendo apresenta bons resultados nas medidas mencionadas anteriormente não significa que este seja de confiança na prática, pelo facto de só ser útil para *logs* simples que não contenham ruído e não tolerar *event logs* reais. É mais aconselhável para estudos teóricos por ser simples do ponto de vista científico.

Para ambas as Médias calculadas, acontece que para alguns logs com a Média Simples acabou por ser escolhido um certo modelo e para a Média Avançada esses mesmos *logs* acabaram por ficar com um modelo diferente. Os modelos e logs que não tiveram mudança nas duas Médias foram: M2L1; M3L4; M2L6; M2L7 e M3L8. De uma forma mais simples com apoio do que foi estudado na teoria, é importante avaliar algumas das escolhas entre *logs* e modelos nas Médias Simples e Avançada.

Avaliando os modelos M2 e M3 com L3, em termos de *Fitness*, M2 (0.89) tem pior resultado que M3 (0.96) porque o *log* contém *loops* de uma mesma atividade que acontece duas vezes seguidas (tarefa duplicada) e M2 não consegue criar um *loop* com esse comportamento. Sendo assim existem *tokens* que acabam por não ser consumidos, diminuindo o *Fitness*. O que acaba por decidir a escolha de M3 com a Média Simples (M2 e M3 têm o mesmo valor de *Precision* Simples, 0.9816) é o *Fitness* e a escolha de M2 com a Média Avançada é mesmo *Precision* Avançada conseguir ter o valor ótimo de 1.0, enquanto M3 apresenta um baixo valor de 0.6582. Uma das causas de M3 ter um baixo valor nesta métrica é a presença de *loops*, sendo que a *Precision* Avançada tem limitações nesse aspeto.

Com os mesmos modelos para L4 conclui-se que o comportamento de ambos é idêntico pelo facto de os resultados de *Precision* Simples e Avançada serem iguais. O que decide qual é o melhor modelo é o *Fitness* de M3 ser ligeiramente melhor (0.9153 de M2 para 0.9180 de M3)

No caso de L6, apesar de o *log* apresentar *loops* curtos, M2 acaba por ter melhor *Fitness* que M3, apesar da diferença ser muito pequena (0.9454 de M2 e 0.9210 de M3). Acontece que M3 acabou por ter um maior número de *tokens* à espera de serem consumidos em comparação com M2 onde uma das causas de isso acontecer foi por ter iniciado uma sequência a partir de uma tarefa que não era suposto (*ask*|*problema*|*evictions web servise*|*problem*). A *Precision* Simples é muito idêntica entre os dois modelos, mas a *Precision* Avançada de M2 consegue um valor ótimo de 1.0 (enquanto M3 obtém um valor de 0.6964) o que conclui M2 ser o mais apropriado para este *log*.

CAPÍTULO 5

CONCLUSÃO E TRABALHO FUTURO

Este trabalho teve como objetivo descobrir técnicas de análise de dados a partir de registo de ações, dar a conhecer melhor o *Process Mining* e uma melhor abordagem aos algoritmos existentes e possíveis para a realização deste trabalho. Antes da análise teve primeiramente a organização dos registos de ações (*logs*) e posteriormente a utilização de algoritmos que serviam para descobrir e comparar os modelos nos *logs*. Ao descobrir os modelos a partir da ferramenta *ProM*, para ser possível uma comparação entre eles, houve necessidade de análise de conformidade. A partir daí obteve-se os resultados entre algoritmos e *logs*, sendo assim possível chegar a uma conclusão mais pormenorizada.

Com os resultados obtidos, utilizando as ferramentas propostas, pode-se concluir que no geral apesar do algoritmo *Heuristic Miner* ter conseguido melhores resultados, o algoritmo *Genetic Miner* acabou por apresentar resultados tão bons como o anterior. É possível afirmar que o algoritmo para ser utilizado não vai depender só de si mesmo, mas também das características que o *log* apresenta. Com a receção dos dados, mesmo depois de organizados em vários *logs*, foi questionado se estes poderiam ter algum erro de informação mal traduzida para o ficheiro *Excel*. Acaba por ser normal quando se trabalha com um variado número de dados em que estes contêm informação de muitos eventos diferentes, resultando numa noção básica chamada de ruído. Além desse motivo, o ruído também pode ser definido por um comportamento raro e pouco frequente, que não costuma ser típico do *log*.

De acordo com a literatura, o *Alpha Miner* revelou bastantes limitações quando analisado na ferramenta *ProM*. Em comparação com algoritmos mais complexos, como *Heuristic Miner* e *Genetic Miner*, o *Alpha Miner* apresenta resultados inferiores, justificando-se pela sua dificuldade de análise em *logs* que contenham ruído e quando a complexidade dos dados aumenta, uma vez que esta constitui a sua principal desvantagem.

No que diz respeito aos algoritmos *Heuristic Miner* e *Genetic Miner*, ambos apresentam o seu modelo através de uma rede heurística, sendo possível depois converter para uma *Petri Net* para serem avaliados através das medidas de conformidade. No caso do algoritmo *Genetic Miner* verificou-se uma maior duração de processamento do modelo final para *logs* grandes, usando cada vez mais recursos do computador. O algoritmo *Fuzzy Miner* foi inicialmente tido em consideração para obtenção de resultados excelentes neste estudo pelo facto de ser robusto, mas acabou por ser desprezado pela não possibilidade de ser convertido para uma *Petri Net*.

Para trabalho futuro nesta área de estudo seria a possibilidade de utilizar um maior número de algoritmos para conseguir ter um maior número de comparações e resultados entre modelos diferentes. Ter uma maior atenção nos *logs* que estão prontos para análise para que qualquer erro existente seja o menor possível e assim os resultados serem mais satisfatórios. Analisar mais aprofundadamente as configurações existentes no *ProM* para cada algoritmo de modo a verificar se é possível melhorar os resultados com configurações não padrão. A partir do que foi dito anteriormente, analisar o desempenho do modelo depois de este ter sido descoberto e a análise de conformidade ter sido verificada. Descobrir uma possibilidade de conseguir resultados (análise de conformidade e desempenho) a partir do algoritmo *Fuzzy Miner*, mesmo tendo conhecimento neste trabalho que este não é possível ser convertido para uma *Petri Net*. Por fim analisar resultados não apenas com a ferramenta *ProM*, mas também com outras que tenham a possibilidade de descobrir e analisar vários modelos de diferentes algoritmos.

REFERÊNCIAS BIBLIOGRÁFICAS

Aalst, W. Van Der, Adriansyah, A., Arcieri, F., & Baier, T. (2011). *Process Mining Manifesto*. *August*. https://doi.org/10.1007/978-3-642-28108-2

Aalst, W M P Van Der, & Hofstede, A. H. M. (2005). *YAWL : yet another workflow language*. *30*, 245–275. https://doi.org/10.1016/j.is.2004.02.002

Aalst, W M P Van Der, Medeiros, A. K. A. De, & Weijters, A. J. M. M. (2005). *Genetic Process Mining*. *i*.

Aalst, Wil M P Van Der. (2007). *Challenges in Business Process Analysis*. *June*. https://doi.org/10.1007/978-3-540-88710-2

Aalst, Wil M P Van Der. (2013). *Process Mining in the Large: A Tutorial*.

Aalst, Wil M P Van der, Dongen, B. F. van, Rozinat, A., Verbeek, H. M. W., & Weijters, A. J. M. M. (2009). *ProM : The Process Mining Toolkit . ProM : The Process Mining Toolkit*. *January*.

Ashraf, I. (2014). *Data Mining Algorithms and their applications in Education Data Mining*.

Azeroual, O., & Theel, H. (2018). *The Effects of Using Business Intelligence Systems on an Excellence Management and Decision-Making Process by Start-Up Companies: A Case Study*. *4*(3), 30–40. https://doi.org/10.18775/ijmsba.1849-5664-5419.2014.43.1004

Bach, M. P., Vugec, D. S., & Vuksic, V. B. (2019). *BPM and BI in SMEs : The role of BPM / BI alignment in organizational performance*. *11*, 1–16. https://doi.org/10.1177/1847979019874182

Balogh, Z., & Kuchárik, M. (2019). *Modeling of Uncertainty with Petri Nets Modeling of Uncertainty with Petri Nets*. *January*. https://doi.org/10.1007/978-3-030-14799-0

Batyuk, A., Voityshyn, V. V. (2018). *PROCESS MINING: APPLIED DISCIPLINE AND SOFTWARE IMPLEMENTATIONS*. 22–36. https://doi.org/10.20535/1810-0546.2018.5.146178

Becker, T., & Intoyoad, W. (2017). Context Aware Process Mining in Logistics. *Procedia CIRP*, *63*, 557–562. https://doi.org/10.1016/j.procir.2017.03.149

Bramer, M. (2016). *Principles of Data Mining* (Issue January 2007). https://doi.org/10.1007/978-1-84628-766-4

Buijs, J. C. A. M., Dongen, B. F. Van, & Aalst, W. M. P. Van Der. (2012). *On the Role of Fitness, Precision, Generalization*. 305–322.

Celik, U., Akcetin, E., & Yaldir, A. (2016). *Analysis of Volvo IT ' s Closed Problem Management Processes By Using Process Mining Software ProM and Disco*. *4*(2).

Gunther, Christian W. , & Aalst, W. M. P. Van Der. (2007). *Fuzzy Mining – Adaptive Process Simplification Based on Multi-Perspective Metrics*.

Christoph F. Strnadl (2006) Aligning Business and It: The Process-Driven Architecture Model, Information Systems Management, 23:4, 67-77, DOI: 10.1201/1078.10580530/46352.23.4.20060901/95115.9

Devi, K. L., & Suryakala, M. (2014). *Educational Process Mining-Different Perspectives*. *16*(1), 57–60.

Dey, A. K., & Abowd, G. (2015). *Towards a Better Understanding of Context and Context-Awareness*. *March*.

Dumas, M., Rosa, M., Mendling, J., & Reijers, H. A. (2018). *Fundamentals of Business Process Management*.

Georgakopoulos, D., & Hornick, M. (1995). *An Overview of Workflow Management : From Process Modeling to Workflow Automation Infrastructure*. *153*, 119–152.

Golfarelli, M., & Rizzi, S. (2004). *Beyond Data Warehousing : What ' s Next in Business Intelligence ?*

Grigorova, K., Malysheva, E., & Bobrovskiy, S. (2017). *Application of Data Mining and Process Mining approaches for improving e-Learning Processes*. 115–121.

Grigorova, K., Mironov, K., & Malysheva, E. Y. (2018). *Applying process mining techniques and neural networks to creating and assessment of business process models*.

Gupta, E. (2015). *Process mining algorithms*. *March*.

Gupta, S. (2007). *Workflow and Process Mining in Healthcare*.

Han, J. e M. Kamber, Data Mining: Concepts and Techniques, Morgan Kaufmann, SanFrancisco, 2000

Han, K. H., Han, S. W., & Choi, S. (2010). *process-based performance measurement model Business Activity Monitoring System Design Framework Integrated With Process-Based Performance Measurement Model*. *January 2015*.

Hernaus, T., Bach, M. P., & Vuksic, V. B. (2012). *Influence of strategic approach to BPM on financial and non-financial performance*. *October*. https://doi.org/10.1108/17465261211272148

Jangvaha, K., Palangsantikul, P., Porouhan, P., & Premchaiswadi, W. (2017). *Analysis of Emergency Room Service using Fuzzy Process Mining Technique*. 2–6.

Jans, M., Soffer, P., & Jouck, T. (2019). *Building a valuable event log for process mining: an experimental exploration of a guided process*. *March*. https://doi.org/10.1080/17517575.2019.1587788

Janssenswillen, G., Donders, N., Jouck, T., & Depaire, B. (2017). *A comparative study of existing quality measures for process discovery*. *71*, 1–15. https://doi.org/10.1016/j.is.2017.06.002

Jassim, M. A., & Abdulwahid, S. N. (2021). *Data Mining preparation: Process , Techniques and Major Issues in Data Analysis*. https://doi.org/10.1088/1757-899X/1090/1/012053

Jovic, A., Brkic, K., & Bogunovic, N. (2015). *An overview of free software tools for general data mining*. *March*. https://doi.org/10.1109/MIPRO.2014.6859735

Kerremans, M; Kitson, N. (2012). *Aligning Business Process Management and Business Intelligence to Achieve Business Process Excellence*.

Ko, R. K. L. (2009). *A Computer Scientist's Introductory Guide to Business Process Management (BPM)*. *15*(4), 11–18.

Koosawad, K., Palangsantikul, P., Porouhan, P., & Saguansakdiyotin, N. (2018). Improving Sales Process of an Automotive Company with Fuzzy Miner Techniques. *2018 16th International Conference on ICT and Knowledge Engineering (ICT&KE)*, 1–6. https://doi.org/10.1109/ICTKE.2018.8612390

Kopceková, A., Kopcek, M., & Tanuska, P. (2013). *Business Intelligence in Process Control*. *October 2014*. https://doi.org/10.2478/rput-2013-0039

Kurniati, A. P., Kusuma, G., & Wisudiawan, G. (2016). *Implementing Heuristic Miner for Different Types of Event Logs*. *11*(8), 5523–5529.

Mayorga, H., & Garcia, Nicolás. (2015). *Minería de procesos: desarrollo, aplicaciones y factores críticos*. *28*(50), 137–157. https://doi.org/10.11144/Javeriana.cao28-50.mpda.

Medeiros, A. K. A. De. (2006). *Genetic Process Mining*.

Mendling, J. (2008). *Metrics for Process Models*.

Mishra, B. K., Hazra, D., Tarannum, K., & Kumar, M. (2016). *Business Intelligence using Data Mining Techniques and Business Analytics*.

Mostafa, A. (2018). Review of Data Mining Concept and its Techniques. February.

Yang, C., Cheng, H., & Juan, Y. (2014). *An Integrated mining approach to discover business process models with parallel structures: towards fitness improvement*. *March 2015*, 37–41. https://doi.org/10.1080/00207543.2014.974847

Premchaiswadi, W., & Porouhan, P. (2018). Process Modeling, Behavior Analytics and Group Performance Assessment of e-Learning Logs via Fuzzy Miner Algorithm. *2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC)*, *01*, 304–309. https://doi.org/10.1109/COMPSAC.2018.10247

R'bigui, H., & Cho, C. (2016). *THE STATE-OF-THE-ART OF BUSINESS PROCESS MINING CHALLENGES*. https://doi.org/10.1504/IJBPIM.2017.10009731

Rozinat, A., & Gunther, C. W. (2012). *Disco: Discover Your Processes*. *940*(2012), 40–44.

Rozinat, A., Medeiros, A. K. A. de, Gunther, C. W., Weijters, A. J. M. M., & Aalst, W. M. P. van der. (2007). *The Need for a Process Mining Evaluation Framework in Research and Practice*.

Rozinat, A, & Aalst, W. M. P. Van Der. (2008). *Conformance checking of processes based on monitoring real behavior*. *33*, 64–95. https://doi.org/10.1016/j.is.2007.07.001

Rozinat, Anne. (2010). *Process Mining : Conformance and Extension*.

Rudnitckaia, J. (2014). *Process Mining. Data science in action*.

Saravanan, M. S. (2011). *Process Mining in Dyeing Unit Using Organizational Perspective: A Case Study*. *3*(3), 1344–1350.

Saylam, R., & Sahingoz, O. K. (2013). *PROCESS MINING IN BUSINESS PROCESS MANAGEMENT : Department of Computer Engineering Turkish Air Force Academy*. 131–134.

Shankar, R., & Duraisamy, S. (2018). *Analysis of Data Mining Tasks, Techniques, Tools,Applications And Trends*. *20*(5), 12–19. https://doi.org/10.9790/0661-2005021219

Song, M., & Aalst, W. M. P. Van Der. (2008). *Towards Comprehensive Support for Organizational Mining*.

Sree, R., & Saravanan. (2010). *PROCESS MINING IN HEALTHCARE USING CONTROL FLOW PERSPECTIVE: A CASE STUDY*.

Tax, N., Lu, X., Sidorova, N., Fahland, D., & Aalst, W. M. P. Van Der. (2017). *The Imprecisions of Precision Measures in Process Mining*. https://doi.org/10.1016/j.ipl.2018.01.013

Verbeek, H. M. W. E., & Bose, R. P. J. C. (2010). *ProM 6 Tutorial*.

Vergidis, K., Tiwari, A., & Majeed, B. (2008). *Business Process Analysis and Optimization : Beyond Reengineering*. *March 2014*.

https://doi.org/10.1109/TSMCC.2007.905812

W.M.P., van der Aalst (2011). *Process Mining Discovery, Conformance and Enhancement of Business Processes*.

Wil M.P. van der Aalst. Business Process Management: A comprehensive survey. ISRN Software Engineering Volume 2013, Article ID 507984, 37 pages, Hindawi Publishing Corporation, http://dx.doi.org/10.1155/2013/507984

Weerdt, J. De, Backer, M. De, Vanthienen, J., & Baesens, B. (2012). A multi-dimensional quality assessment of state-of-the-art process discovery algorithms using real-life event logs. *Information Systems*, *37*(7), 654–676. https://doi.org/10.1016/j.is.2012.02.004

Weerdt, J. De, Schupp, A., Vanderloock, A., & Baesens, B. (2012). Computers in Industry Process Mining for the multi-faceted analysis of business processes — A case study in a financial services organization. *Computers in Industry*, *64*(1), 57–67. https://doi.org/10.1016/j.compind.2012.09.010

Weijters, A. J. M. M., & Aalst, W. M. P. Van Der. (2007). *Genetic process mining : An experimental evaluation*. *April*. https://doi.org/10.1007/s10618-006-0061-7

Weijters, A. J. M. M., Aalst, W. M. P. Van Der, & Medeiros, A. K. A. De. (2006). *Process Mining with the HeuristicsMiner Algorithm*.

Weske, M. (2007). *Business Process Management*.

Weske, M., & Aalst, W. M. P. Van Der. (2004). *Advances in business process management*. *50*, 1–8. https://doi.org/10.1016/j.datak.2004.01.001

Zacarias, Marielba. “Conceptual Framework based on Agents and Contexts for the Alignment between Individuals and Organizations”. PhD Thesis. Instituto Superior Tecnico. July, 2008

Zaiane, O. R. (1999). *Introduction to Data Mining*. 1–15.

Houthoofd, D (2015). Data mining vs. process mining: what's the difference? Acedido a 20 de Novembro de 2020 em: https://www.horsum.be/en/blog/power-bi/data-mining-vs-process-mining-whats-difference

Rozinat, A (2010). ProM Tips — Which Mining Algorithm Should You Use? Acedido a 17 de Outubro de 2020 em: https://fluxicon.com/blog/2010/10/prom-tips-mining-algorithm/

Günther, C. W. (2010). Why We Hate ProM 6. Acedido a 17 de Outubro de 2020 em: https://fluxicon.com/blog/2010/11/why-we-hate-prom-6/

ANEXO

Através da ferramenta *ProM* foram gerados os modelos de cada processo escolhido. Através dos modelos transformados em rede *Petri net* foi possível calcular a verificação de conformidade na secção 4.2 e conseguir resultados a partir do Capítulo 4.

Os modelos dos processos pequenos podem ser visualizados em baixo:

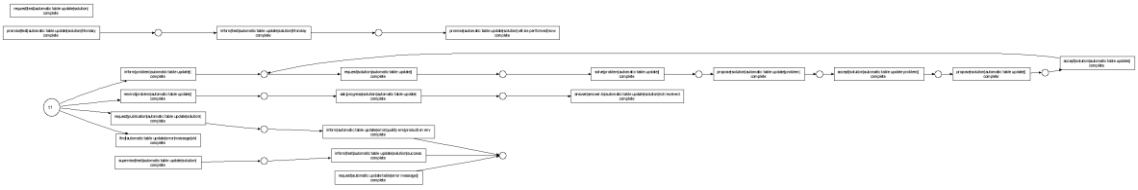


Figura A.1: Modelo *Alpha Miner* do ficheiro “Alpha_Processo_2_table”

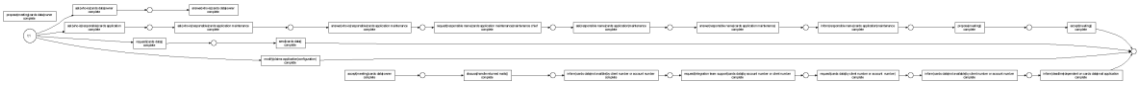


Figura A.2: Modelo *Alpha Miner* do ficheiro “Alpha_Processo_8_cards”

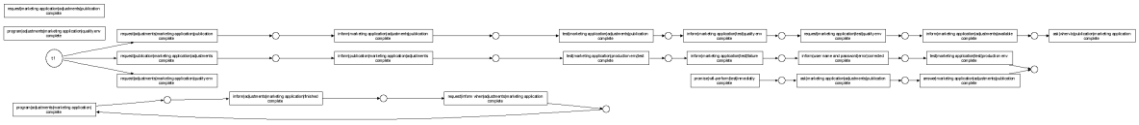


Figura A.3: Modelo *Alpha Miner* do ficheiro “Alpha_Processo_16_campaign”

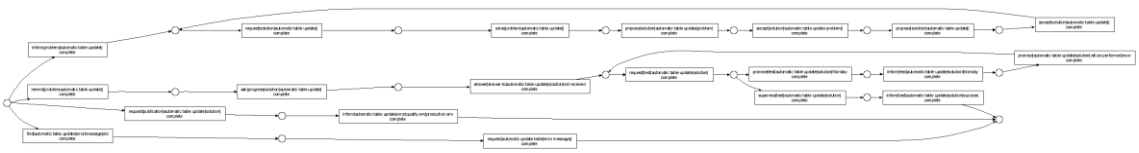


Figura A.4: Modelo *Heuristic Miner* do ficheiro “Heuristic_Processo_2_table”

Figura A.5: Modelo *Heuristic Miner* do ficheiro “Heuristic_Processo_8_cards”

Figura A.6: Modelo *Heuristic Miner* do ficheiro “Heuristic_Processo_16_campaign”

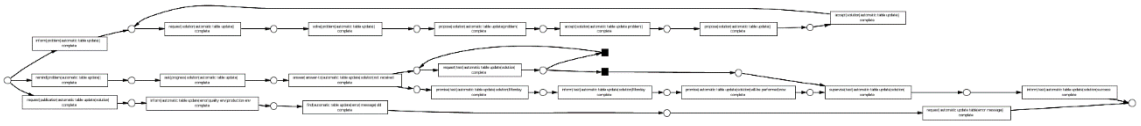


Figura A.7: Modelo *Heuristic Miner* do ficheiro “Heuristic_Processo_2_table”

Figura A.8: Modelo *Genetic Miner* do ficheiro “Heuristic_Processo_8_cards”

Figura A.9: Modelo *Genetic Miner* do ficheiro “Heuristic_Processo_16_campaign”

Os modelos dos processos médios podem ser visualizados em baixo:

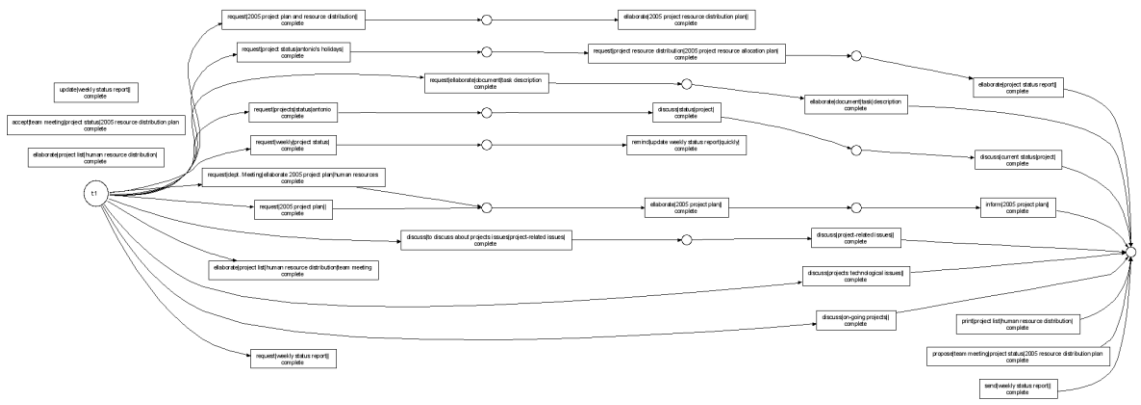


Figura A.10: Modelo *Alpha Miner* do ficheiro “Alpha_Processo_7_projectlist”

Figura A.11: Modelo *Alpha Miner* do ficheiro “Alpha_Processo_11_docfunction”

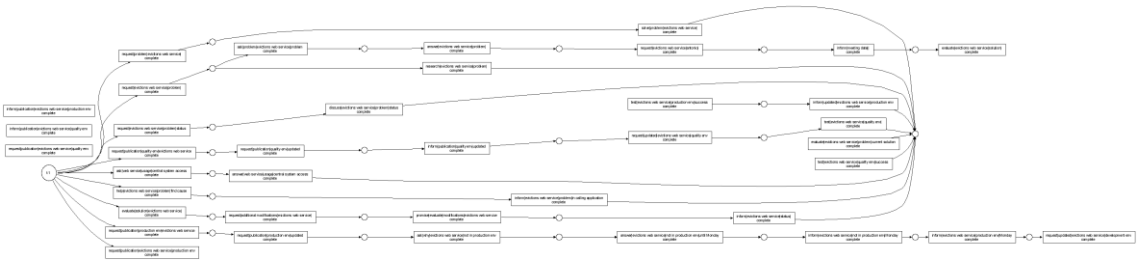


Figura A.12: Modelo *Alpha Miner* do ficheiro “Alpha_Processo_13_webservice”

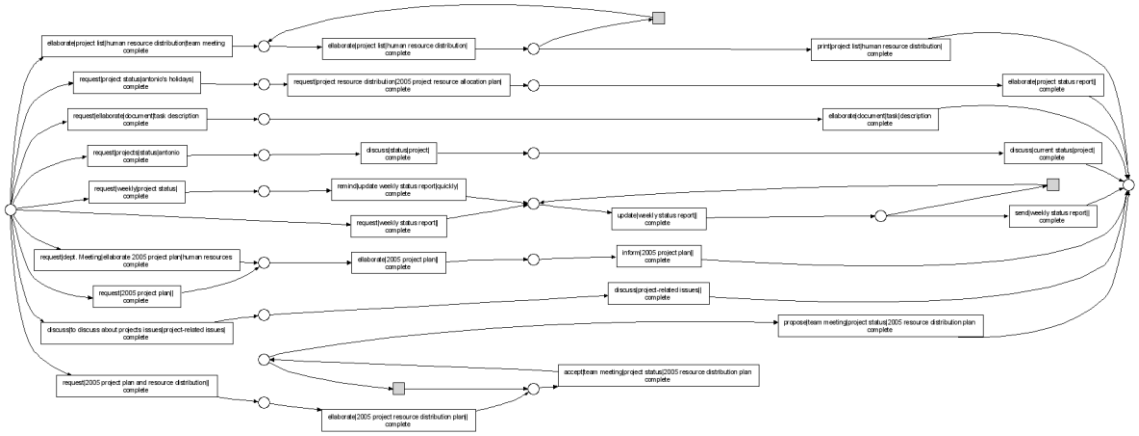


Figura A.13: Modelo *Heuristic Miner* do ficheiro “Heuristic_Processo_7_projectlist”

Figura A.14: Modelo *Heuristic Miner* do ficheiro “Heuristic_Processo_11_docfunction”

Figura A.15: Modelo *Heuristic Miner* do ficheiro “Heuristic_Processo_13_webservice”

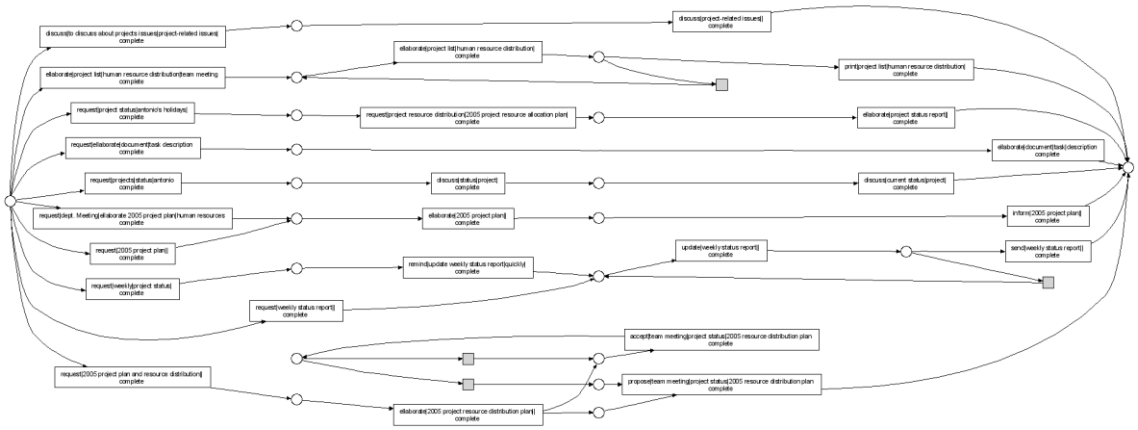


Figura A.16: Modelo *Genetic Miner* do ficheiro “Genetic_Processo_7_projectlist”

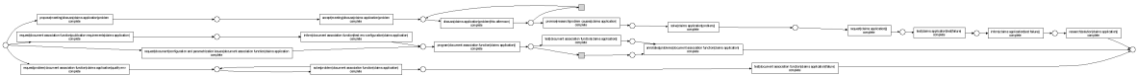


Figura A.17: Modelo *Genetic Miner* do ficheiro “Genetic_Processo_11_docfunction”

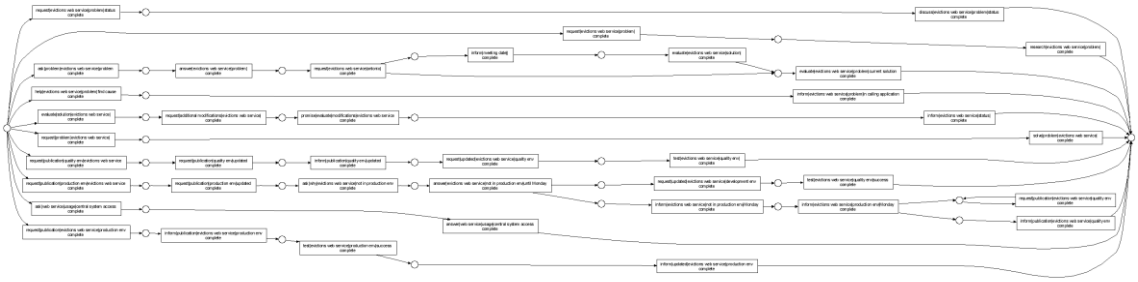


Figura A.18: Modelo *Genetic Miner* do ficheiro “Genetic_Processo_13_webservice”

Os modelos dos processos grandes podem ser visualizados em baixo:

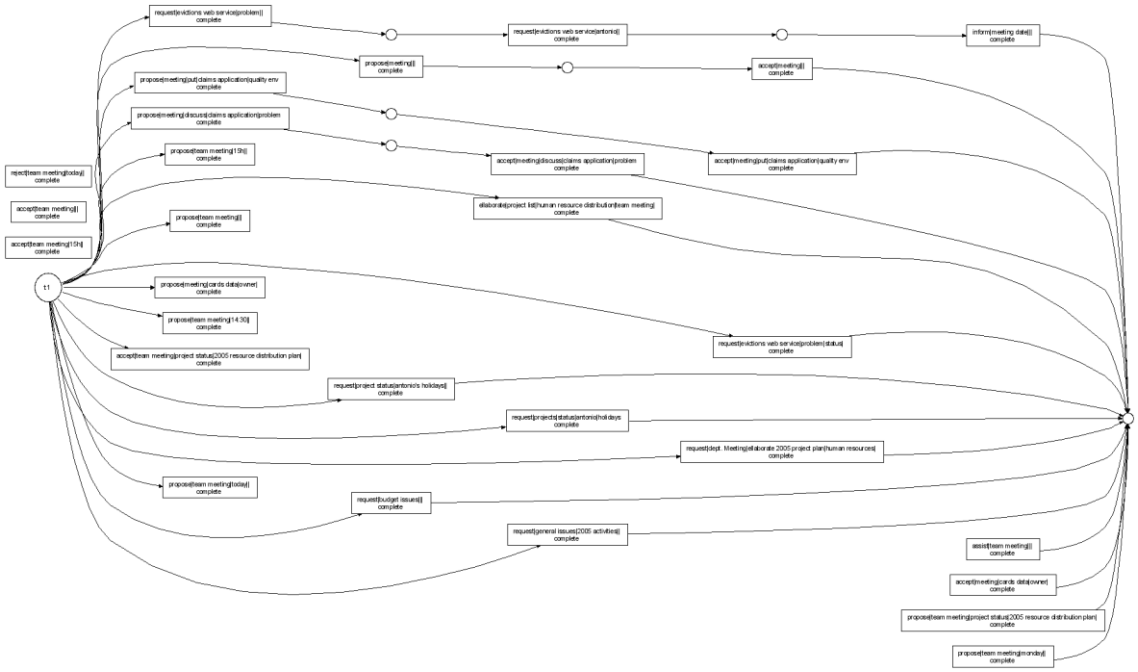


Figura A.19: Modelo *Alpha Miner* do ficheiro “Alpha_Processo_1_meeting”

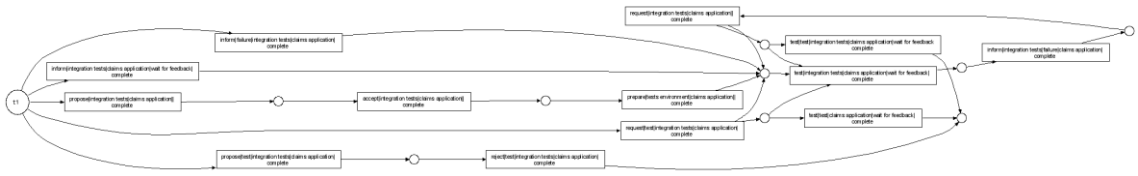


Figura A.20: Modelo *Alpha Miner* do ficheiro “Alpha_Processo_3_appintegration”

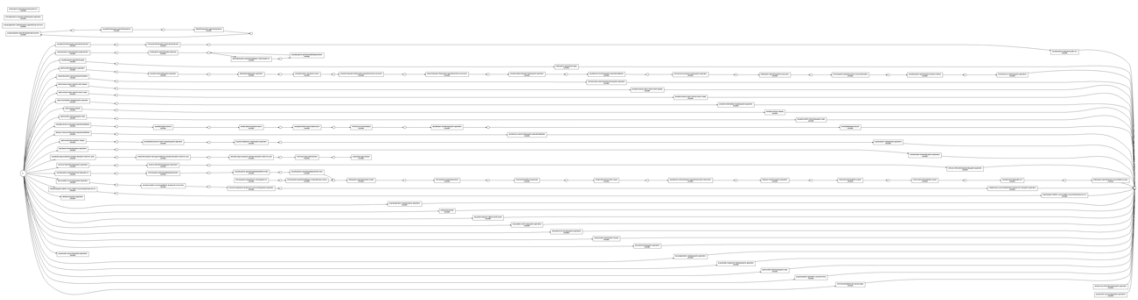


Figura A.21: Modelo *Alpha Miner* do ficheiro “Alpha_Processo_4_supplierappclasses”

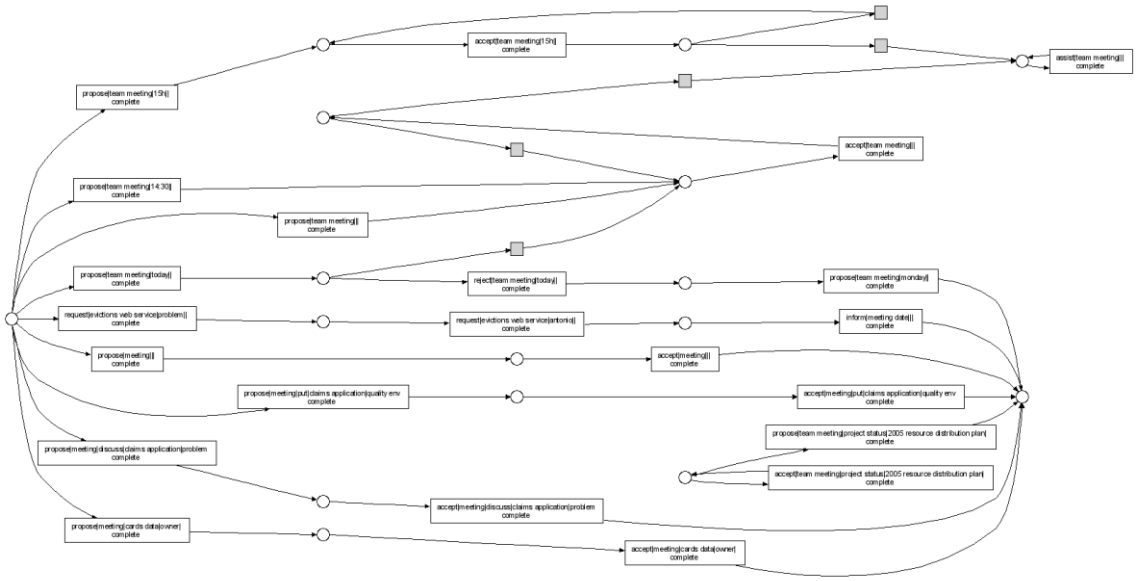


Figura A.22: Modelo *Heuristic Miner* do ficheiro “Heuristic_Processo_1_meeting”

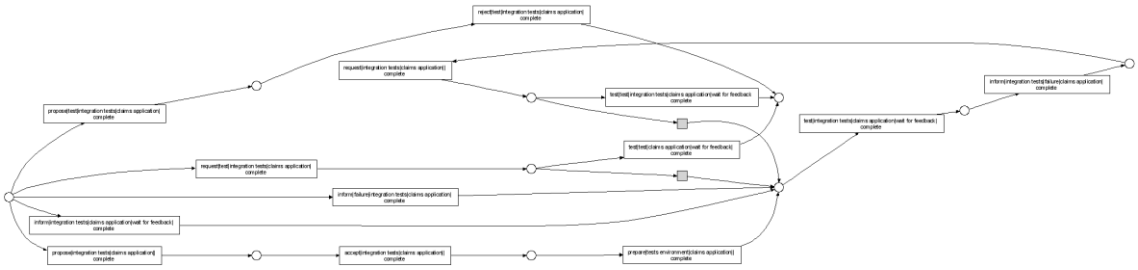


Figura A.23: Modelo *Heuristic Miner* do ficheiro
“Heuristic_Processo_3_appintegration”

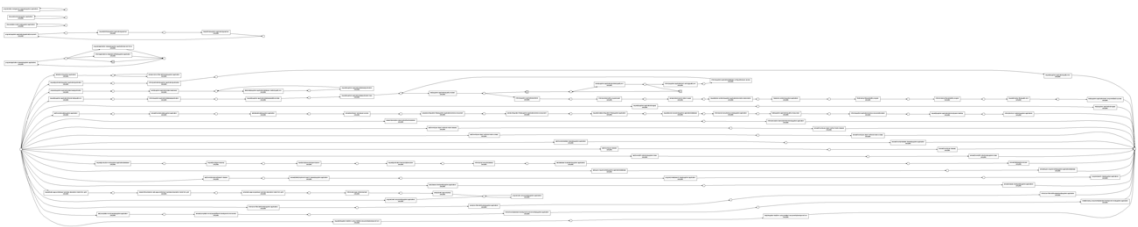


Figura A.24: Modelo *Heuristic Miner* do ficheiro
“Heuristic_Processo_4_supplierappclasses”

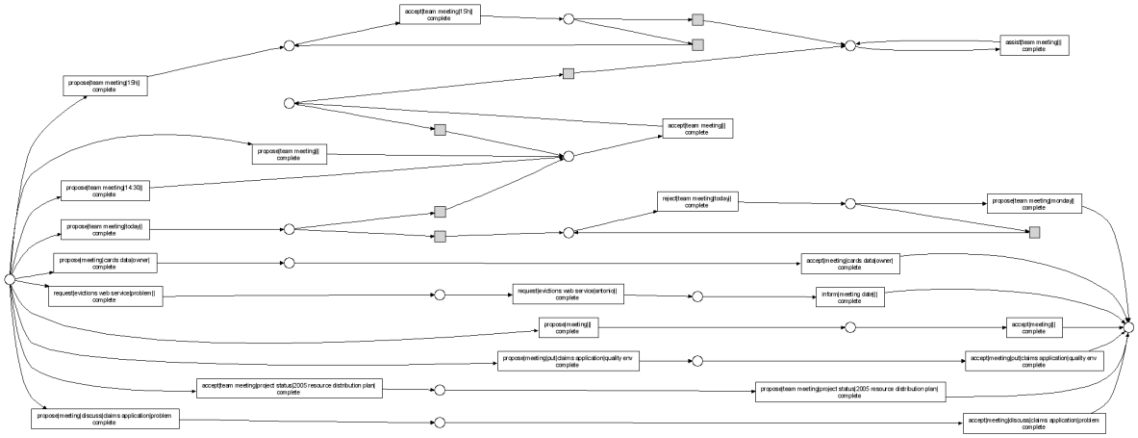


Figura A.25: Modelo *Genetic Miner* do ficheiro “Genetic_Processo_1_meeting”

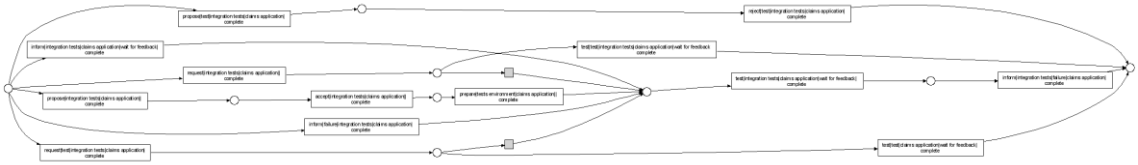


Figura A.26: Modelo *Genetic Miner* do ficheiro “Genetic_Processo_3_appintegration”

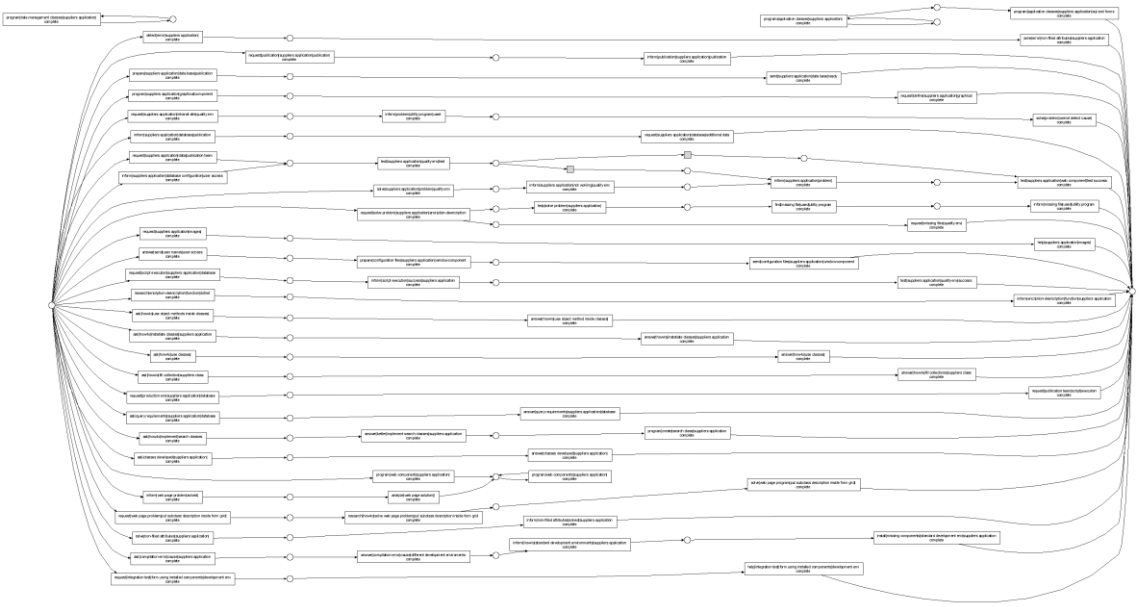


Figura A.27: Modelo *Genetic Miner* do ficheiro
“Genetic_Processo_4_supplierappclasses”