Detecção e reparo interativo de imperfeição de pedido de evento em registros de processo

Prabhakar M. Dixit [1,2] (B), Suriadi Suriadi [2], Robert Andrews [2], Moe T. Wynn [2], Arthur HM ter Hofstede [2], Joos CAM Buijs [1], e Wil MP van der Aalst [1,2]

[1] Eind {hoven University of Technology, Eindhoven, Holanda

pmdixit, jcambuijs, wmpvdaalst} @ tue.nl

[2]

{ Queensland University of Technology, Brisbane, Austrália

Resumo. Muitas formas de análise de dados requerem informações de carimbo de data / hora para ordenar as ocorrências de eventos. o *mineração de processo* disciplina usa registros históricos de execuções de processos, chamados logs de eventos, para derivar insights sobre o comportamento e desempenho dos processos de negócios. Os eventos nos logs de eventos devem ser solicitados, normalmente obtidos usando carimbos de data / hora. A importância das informações de carimbo de data / hora significa que ela precisa ser de alta qualidade. Até onde sabemos, não existe suporte (semi-) automatizado para detectar e reparar problemas de imperfeição relacionados a pedidos em logs de eventos. Descrevemos um conjunto de indicadores baseados em carimbo de data / hora para detectar problemas de imperfeição na ordem de eventos em um log e nossa abordagem para reparar problemas identificados usando conhecimento de domínio. Por fim, avaliamos nossa abordagem implementada na estrutura de mineração de processos de código aberto, ProM, usando dois logs disponíveis publicamente.

Palavras-chave: Imperfeição do log de eventos · Ordem de eventos · Qualidade de dados

1. Introdução

Mineração de processos [1], uma forma de mineração de dados, utiliza registros históricos de execuções de processos para obter percepções sobre o desempenho e o comportamento dos processos de negócios. Central para todas as técnicas de mineração de processo é o “registro de eventos”. Um log de eventos é um conjunto de registros de etapas de processamento (eventos) executados durante várias instâncias de processo (casos). Cada evento é minimamente caracterizado por atributos que identificam a instância de processo a que pertence (*id do caso),* a etapa do processo que foi realizada (*atividade)* e um atributo, normalmente um *timestamp,* que permite que os eventos no caso sejam ordenados. Os registros de eventos podem conter opcionalmente outros atributos, como a pessoa que realizou a etapa do processo.

O ditado *lixo dentro - lixo fora* alude ao fato de que, para todas as formas de transformações de dados em informações, mineração de processos incluída, dados de entrada de baixa qualidade

J. Krogstie e HA Reijers (Eds.): CAiSE 2018, LNCS 10816, pp. 274–290, 2018. https://doi.org/10.1007/978-3-319-91563-0_17

leva a informações de baixa qualidade. Muitas formas de análise de dados, especialmente análises de séries temporais, por exemplo, previsão do mercado de ações e previsão do tempo, precisam de informações de data e hora de qualidade. Da mesma forma, dentro do domínio da mineração de processos, os registros de tempo são os principais meios para atividades de pedido dentro dos casos. A ordenação correta das atividades dentro de um caso é crítica para derivar modelos de processo precisos (descoberta), para determinar se as atividades foram realizadas de acordo com as expectativas organizacionais (conformidade) e para determinar se as atividades e casos são executados de forma eficiente (desempenho) . Assim, está claro que a qualidade dos resultados da mineração do processo depende da qualidade dos dados do carimbo de data / hora. Por exemplo, considere uma situação, onde as atividades em um registro de eventos são extraídas de dois sistemas de informação diferentes, um dos quais registra carimbos de data / hora em granularidade de nível de apenas um dia, enquanto o outro registra carimbos de data / hora em milissegundos. Os efeitos na análise de processo de mineração (ver Fig. 1 por exemplo), inclui modelos de processo descobertos que não refletem a ordem real dos eventos.

A importância de timestamps em processo de mineração analysis é bem reconhecido [2] Vários autores identificaram timestamp-dados relacionados qualproblemas de saúde e seu impacto no processo mineração [1 , 3 - 6] Como ordenação incorreta de eventos pode ter efeitos adversos sobre os resultados da análise de Process Mining, fornecemos um conjunto de indicadores relevantes para detectar problemas relacionados ao pedido em um registro de eventos para identificar as atividades que *poderia* ser ordenado incorretamente. [1] Em seguida, deixamos o usuário reparar o log de eventos injetando interativamente o conhecimento do domínio diretamente no log de eventos, com a ajuda de fragmentos de processo, bem como analisar o impacto do log reparado. Embora existam ferramentas para detecção automatizada e orientada pelo usuário e reparo de dados orientados ao tempo, por exemplo, TimeCleanser [7], até onde sabemos, esta é a primeira técnica que fornece mecanismos de detecção e reparo consolidados para lidar com problemas de pedido relacionados à qualidade de dados em logs de eventos.

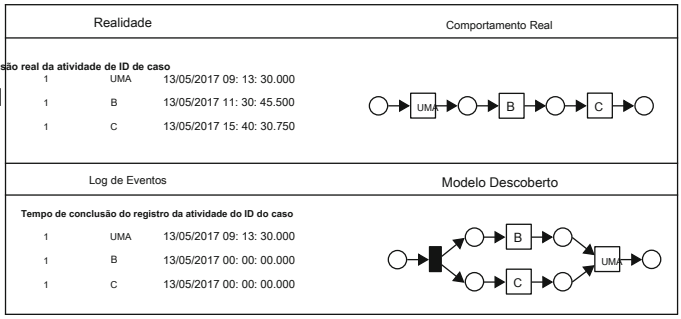


Figura 1. Efeito de carimbos de data / hora imprecisos na ordem de eventos.

Neste artigo, Sect. 2 descreva o *indicadores* de problemas de qualidade de dados de imperfeições de pedidos de eventos, Seção 3 descreve nossa abordagem para detectar, reparar e analisar o impacto do reparo do registro de eventos e do Sect. 4 descreve nossa implementação. Seção 5 avalia a ferramenta em relação a dois conjuntos de dados acessíveis ao público. Trabalhos relacionados são fornecidos na Seção 6 , seguido pela conclusão na seção 7 .

[1] Esses indicadores podem ser específicos para registros de eventos; no entanto, alguns dos indicadores que propomos são genéricos o suficiente para serem aplicáveis a outros dados ordenados pelo tempo.

2 Indicadores de imperfeição de pedido de evento

A abordagem proposta neste artigo visa capacitar especialistas de domínio a detectar e reparar imperfeições de ordenação de eventos que surgem devido a problemas relacionados ao carimbo de data / hora. Como primeiro passo, precisamos ser capazes de *localizar exatamente onde podem existir problemas de carimbo de data / hora em um registro* reconhecendo características comumente associadas à imperfeição na ordenação de eventos. Da literatura existente que descreve problemas de qualidade de dados em registros de eventos [1 , 2 , 4 , 5 , 8] em particular, e dados orientados ao tempo em geral [3], abstraímos três classes de indicadores que podem ser usados para localizar problemas de ordenação de eventos. Não é a intenção deste artigo fornecer uma lista abrangente de indicadores de imperfeição na ordenação de eventos; em vez disso, nosso objetivo aqui é destacar a importância de reconhecer os vários indicadores de imperfeição de pedido de evento de um log de eventos como um ponto de partida para nossa abordagem de reparo de log. Observe que a existência desses indicadores não significa automaticamente que o registro tem problemas de ordem de eventos. No entanto, a capacidade de destacar essas atividades ajudará os especialistas do domínio a selecionar as melhores ações de reparo, se necessário.

Granularidade. Um dos indicadores de imperfeição de ordenação de evento é a existência de granularidade grosseira de carimbo de data / hora (por exemplo *carimbo de data / hora impreciso [* 1 , 8 [2 , 3]). A mistura de granularidade de carimbo de data / hora variável pode resultar em eventos sendo ordenados incorretamente. Por exemplo, um evento *'Visto pelo Doutor'* pode ser registrado na granularidade de nível de dia, por exemplo, '05 Dec 2017 00:00:00 '. Dentro do mesmo caso, outro evento *'Registrar Paciente'*

pode ter granularidade de segundo nível, por exemplo, '05 Dec 2017 19:45:23 '. A ordenação desses dois eventos será *'Visto pelo Doutor'* Seguido por *'Registrar Paciente',* o que é incorreto, pois deveria ser o contrário.

Anomalia do pedido. A localização de eventos afetados por imperfeição de ordenação também pode ser realizada identificando eventos que exibem ordenação temporal incomum, por exemplo
entrada duplicada exatamente do mesmo evento [3] Dada uma atividade uma_1 que foi gravado duas vezes (devido a um usuário clicar duas vezes por engano em um botão em seu
tela), podemos observar uma ordem temporal incomum (infrequente) seguida diretamente
$uma_1 \rightarrow uma_1$, destacando uma possível imperfeição de pedido de evento. Problemas relacionados a eventos ausentes [1 , 4 , 8] e carimbos de data / hora incorretos devido a eventos sendo registrados
post-mortem [1] ou devido à entrada manual [8] também pode ser detectado por aprendizagem se existem outras formas de ordenações temporais incomuns entre as atividades.

Anomalia estatística. Extrair anomalias estatísticas mais genéricas, como aprender a posição temporal de uma determinada atividade no contexto de outras atividades, ou a distribuição de valores de carimbo de data / hora de todos os eventos em um log, pode indicar a existência de problemas relacionados ao carimbo de data / hora. Por exemplo, quando um log

é composto de eventos de vários sistemas, pode haver mais de uma maneira em que os carimbos de data / hora são formatados, o que pode levar ao problema de carimbo de data / hora 'incorreto' ou 'não ancorado' [3 , 5] em que os valores do carimbo de data / hora são interpretados incorretamente. Uma situação comum é quando os valores de carimbo de data / hora formatados como DD / MM / AAAA são interpretados como MM / DD / AAAA. Nessa situação, pode-se ver a distribuição desequilibrada dos valores de 'data' de todos os eventos no registro, em que todos os eventos terão o valor de data entre 1–12 apenas e nenhum para 13–31. Existem outros indicadores de problemas de pedido de evento que podem ser detectados por meio de anomalias estatísticas, como o uso de processamento em lote [3 , 5] e vários problemas de fuso horário [3]

3 Abordagem

Começamos com um log de eventos e adotamos uma abordagem iterativa (ver Fig. 2) para tratar da imperfeição da ordem do evento, passando por (i) detecção automatizada de problemas baseada em indicadores, (ii) reparo orientado pelo usuário, (iii) análise de impacto e (iv) atualização de log. Nesta seção, mostramos, por meio de exemplos, como nossos indicadores são usados para detectar possíveis problemas de ordem de evento, como problemas identificados são reparados usando fragmentos de processo infundidos de conhecimento de domínio e uma estratégia de alinhamento. Por último, descrevemos um conjunto de métricas para avaliar o impacto das ações de reparo no registro que podem orientar o usuário na aceitação ou rejeição dos reparos.

3.1 Detecção

Nesta seção, discutimos as três estratégias de detecção empregadas em nossa ferramenta para detectar os três indicadores apresentados anteriormente na Seção. 2 . A saída da detecção é uma lista consolidada de possíveis problemas de qualidade de dados orientados ao tempo, contendo o problema real e a descrição, a (s) atividade (s) afetada (s) e o número de instâncias afetadas. Nossa técnica não requer nenhuma entrada, exceto o log de eventos, para a detecção dos problemas. Deve-se notar que o objetivo de nossa técnica de detecção é fornecer dicas sobre os prováveis problemas relacionados ao pedido no log de eventos, e o usuário pode escolher ignorar, até mesmo *whitelist,* alguns dos itens detectados na lista. Esses itens da lista de permissões não serão exibidos nos ciclos de detecção subsequentes.

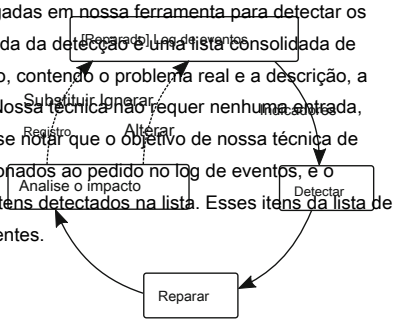


Figura 2. Abordagem de detecção / reparo

Tabela 1. Tabela de granularidade de amostra preenchida **Detecção baseada em granularidade** examinando um log de eventos.

Atividade	Ano	Mês	Dia	Hora	Minuto	Segundo
uma	0	0	0	1	2	500
b	0	0	0	450	20	0
c	0	0	0	0	5	350
d	0	0	0	2	7	448
⋮	⋮	⋮	⋮	⋮	⋮	⋮

faz uso de uma tabela onde as linhas correspondem às atividades e as colunas correspondem à granularidade dos carimbos de data / hora. Cada célula representa a frequência (por granularidade de gravação mais precisa) de uma atividade particular no registro de eventos. Mesa 1 dá um exemplo de tal tabela, que claramente indica aquela atividade *b* geralmente é registrado em uma granularidade mais grosseira (hora) em comparação com as outras atividades. Cada um *b* é adicionado à lista de problemas detectados.

Detecção baseada em pedidos é conceitualmente semelhante ao desafio enfrentado pelas técnicas de descoberta de processos, que tentam deduzir a ordem correta das atividades usando um log de eventos. Aqui, usamos relações causais de pares entre atividades para *acho* a ordem correta de atividades com base nos limites de frequência. Para conseguir isso, preenchemos duas tabelas: uma contendo o diretamente *segue* relações e o outro contendo o diretamente *precede* relações entre as atividades no registro de eventos.

Mesa 2. Um exemplo *segue relações* instantâneo da tabela (esquerda) e *precede relações* instantâneo da tabela (direita).

→	*abc*	*d*	*e*	*f*
⋮	⋮	⋮	⋮	⋮
c	3 0 0 150 200 25			
⋮	⋮	⋮	⋮	⋮

←	*ab*	*c*	*d*	*ef*
uma	0 3	3	450 0 0	
⋮	⋮	⋮	⋮	⋮
f	0 0 25		0	0 0

Cada célula contém o número de vezes que uma atividade de uma linha correspondente foi diretamente seguida / precedida por um atividade da coluna correspondente no log de eventos. Por exemplo, em a *segue* relações de mesa 2 atividade *c* é seguido por atividades *a, d, e* e *f:* 3, 150, 200 e 25 vezes, respectivamente. As discrepâncias na ordenação de cada atividade são analisadas. Por exemplo, considere a atividade *c* na tabela 2 e um valor limite (definido pelo usuário) de 0,8. A primeira etapa é filtrar o menor conjunto de atividades que são seguidas diretamente por *c* resultando em pelo menos 80% do total de ocorrências de *c.*

No *segue* mesa (mesa 2), essas são atividades *d* e *e.* Em seguida, para cada atividade restante com frequência diferente de zero (*uma* e *f)* fora do limite, verificamos o correspondente *precede* tabela de relações com os valores limite para relações de atividade não frequentes diferentes de zero. No caso de *uma,* essas seriam atividades *b* e *c.* Uma vez que ambos seguem e precedem diretamente as relações (*c* → *uma* e *uma* ← *c)* são infrequentes, concluímos que há um problema na ordem entre as atividades *uma* e *c,* e, portanto, essas atividades são adicionadas à lista de problemas detectados. No entanto, para atividade *f,* atividade *c* está dentro do limite, ou seja, todas as ocorrências de *f* são precedidos por *c.* Desde a atividade *f* é altamente infrequente (em comparação com *c),* a ordenação entre *c* e *f* é considerado correto. Portanto, considerando tanto o diretamente *segue* e *precede* relações nós *detectar* apenas aquelas infrequências

que estão relacionados com pedidos. Uma abordagem semelhante é seguida para, eventualmente *segue* e *precede* relações para explorar relações infrequentes de longo prazo.

Estatístico anomalia pode ser usada para detectar imperfeições na ordem de eventos. Neste artigo, investigamos um tipo de anomalia estatística útil na detecção de problemas de ordenação de eventos devido a *mis-fieldding (* veja seita. 2) Por exemplo, dados extraídos como MM / DD / AA do sistema pode ser processado incorretamente como DD / MM / AA formato no log de eventos. Nesse cenário, apenas testemunharíamos dias no intervalo de 1 a 12 e não no intervalo de 13 a 31. Para avaliar tais questões, utilizamos uma técnica estatística direcional - o teste de Kuiper. O teste de Kuiper é especialmente útil nos cenários em que os dados são circulares. Ou seja, ao analisar a distribuição horária da distribuição de atividades, os valores de registro de data e hora 23:59 he 00:01 seriam considerados próximos. Para mais detalhes sobre o teste de Kuiper, referimos o leitor a [9] (pág. 99). Empregamos o teste de Kuiper para comparar a distribuição de cada atividade com todas as outras atividades no registro de eventos, em 6 níveis: hora do dia, minuto da hora, segundo do minuto, dia da semana, dia do mês e mês do ano. A atividade cuja distribuição é estatisticamente signi fi cativamente diferente (p = 0,05) em comparação com todas as outras atividades, seria então exibida na lista de detecção.

3.2 Reparação

As percepções da fase de detecção são combinadas com o conhecimento do domínio para reparar o log de eventos. Uma única atividade é reparada por vez. O fl uxo de trabalho de reparo consiste em quatro etapas, conforme mostrado na Fig. 3 e explicado nas subseções a seguir.

Fig. 3. As quatro etapas do fluxo de trabalho de reparo do processo

Fragmentos do processo de modelagem permite ao usuário especificar o conhecimento do domínio como um fragmento de rede de Petri de livre escolha por meio de um editor de rede de Petri interativo. No mínimo, a rede de Petri contém apenas a atividade que deve ser reparada. O usuário pode adicionalmente incluir “outras atividades” na rede de Petri em relação às quais a atividade deve ser reparada. Por exemplo, considere que o modelo original esperado da Fig. 1 está indisponível. No entanto, o usuário está ciente de que a atividade *C* deve acontecer somente após a atividade *UMA.* Na fase de detecção, o usuário saberia que há um problema com a granularidade da atividade *C.* Portanto, o usuário pode decidir reparar a atividade *C* em relação à atividade *UMA,* modelando uma rede de Petri sequencial simples. Deve-se notar que em muitos cenários da vida real, o log de eventos contém várias atividades e o usuário pode não estar ciente da relação entre *tudo* as atividades. Assim, gostaríamos de apoiar o usuário na modelagem de fragmentos parciais de Petri

redes, especificando explicitamente as relações entre as atividades das quais o usuário tem conhecimento. A modelagem por meio de redes de Petri permite a possibilidade de incluir fragmentos gráficos complexos, como simultaneidade, escolhas, loops, duplicações e introdução *silencioso*
Atividades. As atividades duplicadas na rede de Petri são numeradas de forma exclusiva. Depois de modelar o fragmento da rede de Petri, a próxima etapa é a configuração das opções de reparo.

Tabela 3. As opções de configuração da atividade de reparo.

Atividade	Descrição
Atividade para reparar	Especifique a atividade a ser reparada do fragmento da rede de Petri. Para ocorrências duplicadas na rede de Petri, selecione a instância de atividade (numerada exclusivamente)
Adicionar eventos?	Especifique se as novas atividades devem ser inseridas artificialmente no log de eventos. Se definido como verdadeiro, a técnica pode inserir eventos arti fi ciais (correspondentes à atividade a ser reparada) no registro de eventos
Remover eventos?	Especifique se as atividades devem ser removidas do log de eventos. Se definido como verdadeiro, a técnica pode remover eventos existentes (correspondentes à atividade a ser reparada) do registro de eventos

Configuração de reparo. A configuração de reparo permite ao usuário especificar as configurações para corrigir a ordem de uma atividade particular e consiste em: (i) a configuração da atividade de reparo e (ii) a configuração do contexto de reparo. A configuração da atividade de reparo trata da atividade a ser reparada e é mostrada na Tabela 3 . A configuração do contexto de reparo fornece as informações contextuais de como corrigir uma atividade em relação a outra atividade, usando as chamadas ações de reparo. Cada *ação de reparo* consiste em:

(Eu) **Âncora:** O carimbo de data / hora da atividade a ser reparada seria corrigido
baseado em uma atividade chamada âncora. A atividade âncora é uma atividade da rede de Petri que não pode ser igual à atividade a ser reparada. Alternativamente, a âncora pode ser o início de um caso (primeira atividade) ou o final de um caso (última atividade).

(ii) **Posição:** Selecione se a atividade a ser reparada deve ocorrer *antes*
a âncora ou *depois de* a atividade âncora. (iii) **Valor:** Especifique o valor do novo carimbo de data / hora da atividade a ser reparada
em relação à âncora. Este poderia ser um valor absoluto, por exemplo, 4 h, ou um valor médio / mediano de todas as relações de conformidade do registro de eventos entre a atividade da âncora e a atividade a ser reparada pela rede de Petri modelada.

Múltiplas ações de reparo podem ser especi fi cadas para reparar uma atividade. A ordem das ações de reparo determina a seqüência em que os reparos são executados no registro de eventos, de forma que, se a primeira ação falhar, a segunda será aplicada e assim por diante.

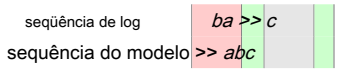
Execute reparos. O reparo real é realizado usando o resultado da técnica de conformidade baseada em alinhamentos [10] Demonstramos o uso da estratégia de alinhamentos para realizar o reparo real do log de eventos com um exemplo. Considere a ordem de uma atividade *b* está incorreto em um log de eventos e precisa ser corrigido. Vamos supor que a ordem correta de atividades *b* em relação a duas atividades *uma* e *c* é uma relação sequencial: *uma → b → c,* ie atividade *uma* deve ser seguido por *b,* que deve ser seguido por *c.* Essa relação pode ser facilmente modelada usando uma rede de Petri. Como primeiro passo, o log de eventos original é duplicado. Vamos ligar

este log de eventos duplicado *eu* D. Este registro de evento duplicado e o fragmento da rede de Petri são usados como entrada para os alinhamentos. A conformidade baseada em alinhamentos

A técnica encontra um caminho de navegação ideal em um modelo de rede de Petri para uma sequência particular de atividades. Ele faz isso atribuindo um *custo* para cada atividade, que é então usada para determinar a ordem correta com o objetivo de minimizar o custo geral. Todas as atividades que não estão presentes no fragmento da rede de Petri são atribuídas a um custo zero. Em nosso exemplo, como *b* é a atividade a ser reparada, presume-se que o posicionamento da atividade *b* é impreciso em comparação com *uma* e

c. Conseqüentemente *uma* e *c* são atribuídos custos mais elevados em comparação com *b,* ou seja, é preferível alinhar *uma* ou *c* ao custo de *b.* Agora, considere que há um caso no log de eventos *eu* D, de modo que todas as três atividades (*a, b* e *c)* acontecer uma vez e no mesmo dia, Contudo *b* acontece às 08:00, *uma* acontece às 09:00 e *c* acontece às 10:00. É fácil ver que este caso não se encaixa no *esperado* comportamento sequencial (*uma → b → c).*

O resultado dos alinhamentos para tal caso seria, portanto:



No exemplo de alinhamentos acima, atividades *uma* e *c* Exibir *síncrono* comportamento de acordo com o fragmento da rede de Petri e o registro de eventos. A primeira ocorrência de *b (* vermelho) é um *mova no log,* indicando um comportamento no log de eventos que não pode ser reproduzido na rede. O segundo *b (* cinza) é um *mova o modelo,* indicando uma ocorrência esperada de *b* de acordo com o fragmento da rede de Petri que não ocorreu no registro de eventos. Agora, a ordem de *b* é corrigido usando cada ação de reparo da caixa de configuração de contexto da seguinte forma:

1. Para cada *seguir em frente* do *b* no alinhamento de um caso particular, mude
   o carimbo de data / hora de um evento correspondente a um *mover no log* do *b* para esse caso. O novo carimbo de data / hora é definido com base na ação de reparo atual. Um carimbo de data / hora de um evento pode ser atualizado apenas uma vez para uma ação de reparo específica. Deve-se notar que, no caso de padrões de rede de Petri, como *rotações,* o mais perto *síncrono*
   a atividade âncora é escolhida (caso a âncora não seja início ou fim). Em nosso exemplo acima, o carimbo de data / hora do evento correspondente ao primeiro *b* é alterado com base na ação de reparo configurada.
2. Se *adicionar eventos* é definido como verdadeiro: Let *n* seja a diferença entre o número de *seguir em frente* do *b* e o número de *mover no log* do *b* para um caso particular. E se *n* é um número positivo diferente de zero, então adicione *n* número de eventos arti fi ciais de atividade *b,* com base na ação de reparo escolhida.
3. Se *remover eventos* é definido como verdadeiro: Let *n* seja a diferença entre o número do *mover no log* do *b* e o número de *seguir em frente* do *b* para um caso. E se *n* é

número positivo diferente de zero, em seguida, remova *n* número de eventos de atividade *b* cujos valores de carimbo de data / hora não foram alterados (na etapa (a)), e cuja etapa de alinhamento corresponde a *mover no log*.

4. Realize alinhamentos no registro de eventos reparado, e refaça e replique todos aqueles casos com ordem correta de atividade *b* no log de eventos eu_D. Crio um novo log de eventos usando eu_D contendo apenas os casos que têm ordem incorreta de atividade *b*. Realize alinhamentos neste log de eventos recém-criado, e repita os passos (a) - (d) até que todas as ações de reparo sejam realizadas. Alternativamente, pare se o posicionamento da atividade *b* para todos os casos está correto.

3.3 Análise de Impacto

Tendo reparado o log de eventos, o usuário recebe informações sobre o impacto do reparo no log de eventos. O usuário pode escolher substituir o atual cópia de trabalho do log de eventos com o log reparado (eu_D), ou opte por ignorar o log de eventos reparado. As métricas apresentadas ao usuário para analisar o impacto são: (i) editar distância: distância de Levenshtein entre o registro de eventos reparado e o registro de eventos original, (ii) aptidão [10] do registro de eventos original e do registro de eventos reparado para as atividades do fragmento da rede de Petri, (iii) o número total de casos impactados, (iv) o número total de eventos adicionados, (v) o número total de eventos removidos, e (vi) o número total de eventos com um valor de carimbo de data / hora alterado.

4 Implementação

As técnicas propostas foram implementadas no ProM [2], e pode ser usado a partir do pacote “Mineração de processo interativo” na versão noturna do ProM. Figura 4 mostra a tela de detecção de nossa ferramenta. O usuário é apresentado a todos os problemas de qualidade detectados (junto com sua descrição, frequências, etc.) em um formato tabular (painel A). Ao selecionar um problema da tabela de detecção, o usuário recebe o registro de eventos filtrado (painel B) que contém apenas os eventos relevantes para o problema detectado. Além disso, o usuário também recebe um histograma agregado (painel C) mostrando a distribuição de todos os eventos afetados com base no tipo de problema detectado.

Figura 5 mostra a tela de reparo de nossa ferramenta. O usuário modela uma rede de Petri interativamente adicionando uma atividade por vez (painel A). Esta rede de Petri especifica a relação entre a atividade a ser reparada com outras atividades. O usuário especifica a estratégia de reparo na visualização de configuração de reparo (painel B). Ao realizar o reparo, o impacto é apresentado ao usuário (painel C) que pode ser utilizado para a tomada de decisão de manter ou descartar as alterações.

[2] http://www.processmining.org/prom/start .

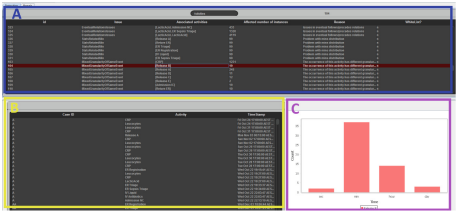


Fig. 4. A guia de detecção mostrando os três painéis de detecção: (A) a lista de problemas detectados, (B) a visualização do log de eventos e (C) a visualização do gráfico mostrando as distribuições para o problema selecionado.

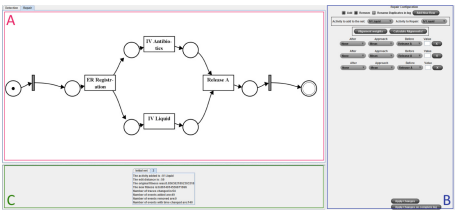


Fig. 5. A guia de reparo mostrando os três painéis de reparo: (A) a rede de Petri modelada interativamente pelo usuário, (B) a visualização da configuração do reparo e (C) o impacto de uma (sequência de) reparo (s).

5 Avaliação

Detectar e reparar anomalias de pedido de evento envolve a reestruturação do registro de eventos para corrigir o aspecto do fluxo de controle. Conseqüentemente, para cada log de eventos, demonstramos a capacidade de detecção de ordenar problemas relacionados no log de eventos, seguido pelo reparo dos logs de eventos usando os insights da detecção junto com o conhecimento do domínio. O resultado final é avaliado com base nos modelos de processo descobertos usando o registro de eventos reparado. Os registros de eventos usados são: (i) registro de eventos de sepse - em que o modelo de processo descoberto a partir de um registro de eventos reparado é comparado com o modelo de processo de verdade, (ii) registro de eventos BPIC 2015 - em que os resultados das técnicas de descoberta de processo são comparados antes e após o reparo do log de eventos. A partir de [11] e os relatórios enviados ao BPIC 2015, sabemos que ambos os registros de eventos apresentam problemas de qualidade relacionados à solicitação de eventos. No entanto, os problemas e reparos exatos eram desconhecidos e / ou indisponíveis.

Conforme detalhado na Seção 3 , acesso a *conhecimento de domínio* é essencial em nossa abordagem. Para a avaliação de nossa abordagem usando o registro Sepsis, o conhecimento do domínio foi adquirido por meio de um analista de processo que teve ampla consulta com especialistas de domínio relevantes [11] Para a avaliação usando o log do BPIC 2015, o conhecimento do domínio foi adquirido por meio da consulta aos relatórios submetidos ao BPIC 2015. Reconhecemos que nossa avaliação pode ser limitada pela ausência de *especialistas de domínio*

que seria capaz de con fi rmar a validade da tora reparada e lançar luz sobre qualquer fenômeno peculiar, mas ainda válido, dentro do processo sendo analisado que pode ser perdido devido às ações de reparo aplicadas. No entanto, mesmo sem acesso a especialistas no domínio, nossa abordagem é capaz de melhorar a qualidade do registro, conforme evidenciado pela capacidade de gerar um modelo de processo mais próximo do modelo de processo de verdade conhecido (para o registro da Sepse) e de melhor qualidade do modelo de processo (para o registro de desafio BPIC 2015). Os detalhes da avaliação de nossa abordagem são fornecidos abaixo.

5.1 Sepse

O registro da sepse [3] [11] (1.050 pacientes / casos e 16 atividades) contém o processo de tratamento da sepse de pacientes em um hospital holandês. O modelo de processo de "verdade fundamental" (ver Fig. 6) foi desenhado à mão em consulta com especialistas do domínio (consulte [11] para detalhes).

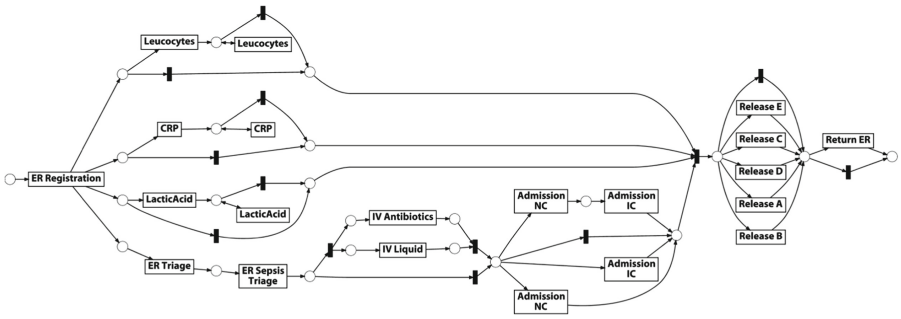


Fig. 6. Modelo original de sepse para o registro de eventos, conforme modelado em [11]

O resultado da detecção de nossa ferramenta indicou que 96% dos problemas estavam relacionados à granularidade e ao pedido e os 4% restantes eram problemas decorrentes de anomalias estatísticas. Entre as questões relacionadas com pedidos, quase 50% (14 de 28) dos *direto* problemas de pedido tiveram pelo menos uma das três atividades:
Antibióticos IV, líquido IV e / ou *Admissões NC,* o que sugeriu um possível problema de pedido com relação a essas atividades.

Aqui, damos um exemplo das etapas seguidas para reparar uma das atividades (*Admissões NC).* O fragmento da rede de Petri mostrado na Fig. 7 foi modelado com base no domínio de conhecimento sobre protocolos de sepse da literatura. Para reparar a atividade *Admissões NC,* a posição do *Admissões NC* foi configurado para ser posterior (a duração média de todos os casos de adaptação) *IV Antibióticos*

ou *IV Líquidos* para os casos desalinhados. No total, a ordem do carimbo de data / hora de seis atividades foi corrigida (ver Tabela 4) Depois de reparar os valores de tempo (e remover duplicatas) das 6 atividades, Inductive miner-incomplete [12] foi usado para descobrir automaticamente um modelo de processo do log de eventos reparado final (Fig. 9) Deve-se notar que o modelo de processo (ver Fig. 8) descoberto a partir do
o log de eventos original usando miner-incompleto indutivo inclui muito paralelismo e permite auto-loops para quase todas as atividades. Ao comparar os modelos de processo nas Figs. 6 e 9 , é bastante evidente que, usando nossa ferramenta para reparar o log de eventos em apenas 6 etapas, fomos capazes de chegar muito perto do modelo de processo descoberto manualmente pelos autores de [11] Conseguimos isso apesar da suposição de *ausência* de um modelo normativo completo para começar. Em vez disso, olhamos para o *problema* atividades da fase de detecção, e usado

[3] https://data.4tu.nl/repository/uuid:915d2bfb-7e84-49ad-a286-dc35f063a460 .

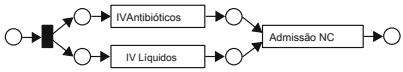


Fig. 7. *Admissão NC* deveria estar opcional (ver Fig. 6), mas de acordo com o conhecimento de domínio disponível, só se sabe que se *Admissão NC* ocorre, deve ocorrer sempre (apenas uma vez) após *IV Antibióticos* e *IV Liquid.*

Tabela 4. Atividades reparadas no diário de sepse.

Atividade reparada	Editar distância	% traços impactados
Triagem ER	15	0.9
Triagem de sepse ER	46	2,3
IV líquido	124	6,2
IV Antibióticos	22	1,1
Admissão NC	407	32,3
IC de admissão	17	1,2

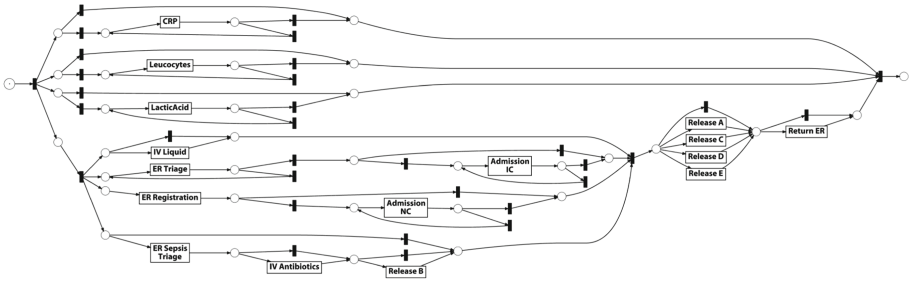


Fig. 8. Modelo de sepse descoberto a partir do registro de eventos original usando o algoritmo de descoberta de minerincomplete indutivo.

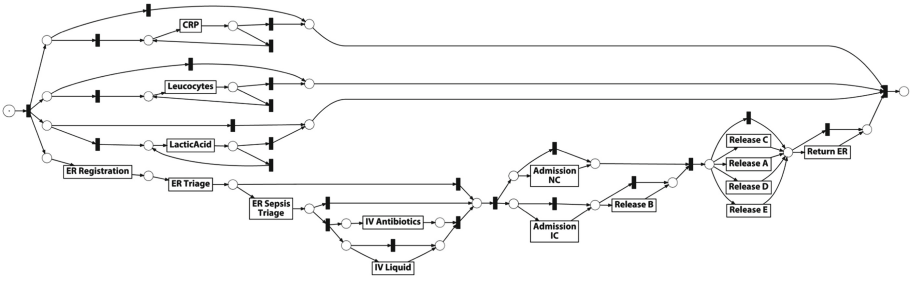


Fig. 9. Modelo de sepse descoberto a partir do registro de eventos reparado usando o algoritmo de descoberta de minerincomplete indutivo.

os protocolos clínicos relativos a tais atividades como estão, especificadas por fragmentos de rede de Petri, a fim de reparar o registro de eventos.

5.2 BPIC 2015

Também avaliamos a nossa abordagem em relação ao Desafio BPI 2015 [4] registros de eventos que tratam de pedidos de licença de construção de cinco municípios nos Países Baixos. Usamos o registro de eventos de um dos municípios (município 1) que

[4] http://www.win.tue.nl/bpi/2015/challenge .

contém quase 400 atividades. Para tornar os resultados compreensíveis, filtramos o registro para selecionar as 25 atividades mais frequentes pertencentes ao processo de aplicação principal e enfocamos quase 700 *fechadas* casos.

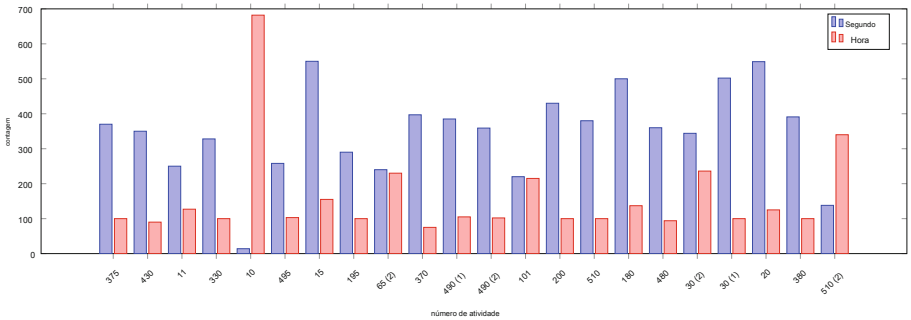


Fig. 10. Número de eventos por atividade com granularidade horária vs. segunda.

A técnica de detecção de nossa ferramenta indicou um provável problema de granularidade (50% de todos os problemas). É claro que os problemas de granularidade podem influenciar o pedido e, portanto, havia um grande número de problemas relacionados ao pedido também (47%), em comparação com 3% das estatísticas problemas de anomalia.

Entre os problemas de granularidade, mais de 10 atividades tinham valores de tempo multigranulares (consulte a Figura 10) e muitas atividades tiveram alto número de eventos registrados em *de hora em hora* granularidade resultando na perda de informações de pedidos no log de eventos. Portanto, reparamos o log de eventos para corrigir manualmente a ordem das 10 principais atividades com alto número de granularidade por hora. As atividades no log de eventos filtrado

consiste na seguinte estrutura de nomenclatura: 01 HOOFD ### (algumas atividades têm um ## adicional no final). A partir do conhecimento de domínio disponível, sabe-se que normalmente os três últimos dígitos de 01 HOOFD ### denotam a sequência de ordenação das atividades (no entanto, pode nem sempre ser o caso). Usamos esta informação para *reparar* o posicionamento das atividades que possuem um elevado número de eventos com granularidade horária, em relação às atividades que possuem elevado número de eventos com granularidade de segundos. Ao fazer isso repetidamente, aumentamos a granularidade de quase 45% dos eventos que inicialmente tinham granularidade horária. Porque não temos *terra*

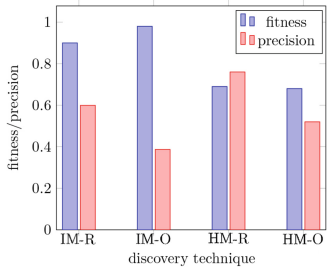


Fig. 11. Fitness e precisão (log original). IM / HM: minerador indutivo infrequente / minerador heurístico algoritmos. -R / -O inicial denota registros reparados e originais usados para descoberta.

verdade para o log BPIC 2015, nós usar as dimensões de qualidade da qualidade de mineração de processo (ou seja, a fidelidade do

modelo para o log) e precisão (ou seja, até que ponto os comportamentos não vistos no log são permitidos pelo modelo) para avaliar os resultados. Geralmente, um modelo de processo com maior aptidão e precisão é desejável [1] Nós descobrimos modelos de processo usando dois algoritmos de descoberta de processo de última geração (minerador heurístico [13] e minerador indutivo [12]) usando o log de eventos original (não reparado) nas configurações padrão. Em seguida, usamos os mesmos algoritmos para descobrir modelos de processo usando o log de eventos reparado. Todos os quatro modelos de processo descobertos são avaliados em relação ao registro de eventos original (não reparado) para avaliar usando a aptidão [10] e precisão [14] dimensões (ver Fig. 11) A partir da fase de detecção, ficou claro que havia problemas de granularidade (e, portanto, problemas de ordem de eventos) no log de eventos, ao reparar o log de eventos usando o conhecimento de domínio disponível, melhoramos significativamente a precisão dos modelos, restringindo assim o modelo de permitir muito comportamento não observado, com pouco ou nenhum impacto sobre a adequação do modelo com o registro de eventos.

6 Trabalhos Relacionados

Nesta seção, consideramos noções de qualidade de dados em geral, qualidade de dados para logs de eventos (timestamps em particular) e discutimos o trabalho relacionado à detecção e reparo de problemas de qualidade de dados relacionados a pedidos de eventos. A qualidade dos dados tem uma longa história de pesquisa ativa (consulte [15 - 17]). A qualidade dos dados é geralmente descrita como sendo um conceito multidimensional com dimensões como exatidão / correção, integridade, compreensibilidade, atualidade e precisão [18 , 19] sendo freqüentemente mencionado. A qualidade dos dados para logs de eventos, seu impacto na mineração de processos e a importância particular dos carimbos de data / hora para os dados do evento foram considerados pela primeira vez em [2] Em linha com a discussão geral sobre qualidade de dados, a consideração da qualidade do log de eventos também usa estruturas multidimensionais e adota dimensões semelhantes. Por exemplo, Mans et al. [6] descrevem a qualidade do registro de eventos como um espectro bidimensional com a primeira dimensão preocupada com o nível de abstração dos eventos e a segunda relacionada com a precisão do carimbo de data / hora (em termos de (i)

granularidade, (ii) *objetividade do registro (* ou seja, a moeda da gravação do carimbo de data / hora) e (iii) *correção)* fazendo assim *explícito* a importância dessas dimensões de qualidade para a ordenação temporal adequada de eventos.

Notamos que muitas técnicas de mineração de processo *implicitamente* detectar problemas de qualidade no log de eventos. Por exemplo, técnica de verificação de conformidade por [10] combina a expectativa (modelada por uma rede de Petri) com a realidade (de acordo com os registros de eventos) para detectar comportamento em conformidade e desvio. Assim, os problemas de qualidade de dados relacionados ao pedido apareceriam como comportamento divergente. No entanto, tais técnicas *exigir* um modelo de processo para fazer a análise. Nossa abordagem não requer informações básicas durante a fase de detecção. Muitos algoritmos de descoberta de processo (por exemplo, [12 , 13 , 20]) implicitamente tenta detectar ruído no registro de eventos, que às vezes pode ser atribuído a imperfeições de ordenação de eventos. No entanto, as decisões e detecções feitas durante a fase de descoberta são implicitamente incorporadas no modelo de processo descoberto, mas não explicitamente apresentadas ao usuário. Nossa abordagem apresenta explicitamente ao usuário as anomalias individuais relacionadas às imperfeições de ordenação de eventos.

Algumas técnicas da literatura quantificam automaticamente a qualidade de um registro de eventos. [21] é um pacote em R que fornece informações agregadas sobre a estrutura e comportamento de um log de eventos. Similarmente [22] discute várias métricas para quantificar a qualidade geral do log de eventos. No entanto, essas técnicas geralmente fornecem uma medida global da qualidade dos dados e não indicam a lista exata de problemas no log de eventos. Nossa abordagem se concentra em fornecer ao usuário uma lista de problemas de qualidade relacionados ao tempo no registro de eventos.

Ao contrário da descoberta de processos e da análise de desempenho, o reparo de problemas de qualidade de dados em logs de eventos é um território quase inexplorado no campo de mineração de processos. Os autores de [23 , 24] descreve técnicas para corrigir o posicionamento de eventos com base em redes de Petri (temporizadas) e probabilidades derivadas de alinhamentos. Dentro [25] os autores descrevem uma estratégia de reparo heurístico baseada na decomposição da rede de Petri para um reparo eficiente do registro de eventos. Na realidade, entretanto, um modelo de processo padrão de fato raramente está disponível. Em comparação com essas abordagens que pressupõem a presença de um modelo de processo, nossa abordagem não requer um modelo de processo de ponta a ponta para o reparo do log de eventos. Nossa abordagem requer apenas conhecimento de domínio suficiente para reconhecer as fontes de problemas de carimbo de data / hora a partir dos indicadores de qualidade de carimbo de data / hora detectados e, então, construir fragmentos de processo apropriados.

Dentro [26 , 27], os autores descrevem uma abordagem baseada em restrições de negação e uma abordagem baseada em restrições temporais, respectivamente. para reparar automaticamente as marcações de tempo de eventos no registro de eventos. Novamente, essas abordagens exigem que os usuários pré-especifiquem as restrições, ao passo que, em nosso caso, o usuário pode primeiro analisar os problemas detectados e, então, escolher agir de acordo com eles. Dentro [7] é discutido um aplicativo guiado visual para reparar problemas relacionados ao tempo no log de eventos. Ao contrário das abordagens baseadas em restrições e abordagens orientadas por visualizações para reparo de log de eventos, nossa abordagem permite que o usuário especifique de forma flexível o conhecimento de domínio usando fragmentos de processos gráficos, que normalmente são mais intuitivos e permitem a especificação de comportamentos de processos complexos, como simultaneidade, loops, escolhas , duplicação de atividades etc. Em [28] é proposta uma técnica para remover automaticamente o comportamento “ruidoso” do log de eventos, sem qualquer envolvimento do usuário. No entanto, em nosso caso, apresentamos esses problemas prováveis ao usuário, em vez de removê-los automaticamente do registro de eventos. O usuário pode reparar temporariamente os problemas, analisar o impacto e, opcionalmente, tornar as alterações permanentes.

7. Conclusão

Apresentamos uma técnica integrada para detectar e reparar imperfeições na ordem de eventos em um log de eventos. Os indicadores de imperfeições na ordenação de eventos foram discutidos, seguidos de uma estratégia abrangente para detectar tais indicadores. A abordagem de reparo proposta permite ao usuário desenvolver uma verdade global (projetada como um fragmento de processo), que pode ser aplicada localmente em cada caso no log de eventos. Isso torna nossa abordagem de reparo muito mais fácil em comparação com uma estratégia de reparo caso a caso. Além disso, fragmentos de processos gráficos intuitivos podem ser projetados interativamente pelos usuários para fácil incorporação do conhecimento do domínio. Aplicamos nossas técnicas de detecção e reparo em dois registros de eventos da vida real.

Pudemos mostrar que podemos detectar anomalias e repará-las, levando à descoberta de melhores modelos de processo. Este trabalho focou principalmente nas questões de qualidade de dados de uma perspectiva de fluxo de controle de mineração de processo. No futuro, gostaríamos de abordar a perspectiva de desempenho e conformidade, além de explorar outros problemas de qualidade de dados na mineração de processos.

Reconhecimento. As contribuições para este artigo de R. Andrews, MT Wynn e AHM ter Hofstede foram apoiados por meio do ARC Discovery Grant DP150103356.

Referências

1. van der Aalst, WMP: Process Mining - Data Science in Action, 2ª ed. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-
2. van der Aalst, W., Adriansyah, A., de Medeiros, AKA, Arcieri, F., Baier, T., Blickle, T., Bose, JC, van den Brand, P., et al .: Processo manifesto de mineração. Em: Daniel, F., Barkaoui, K., Dustdar, S. (eds.) BPM 2011. LNBIP, vol. 99, pp. 169–194. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-28108-2 19
3. Gschwandtner, T., G¨ ärtner, J., Aigner, W., Miksch, S .: A taxonomy of dirty time-dados orientados. Em: Quirchmayr, G., Basl, J., You, I., Xu, L., Weippl, E. (eds.) CDARES 2012. LNCS, vol. 7465, pp. 58–72. Springer, Heidelberg (2012). https: // doi. org / 10.1007 / 978-3-642-32498-7 5
4. Bose, JC, Mans, RS, van der Aalst, WMP: Quer melhorar os resultados de mineração de processo? IEEE CIDM **2013,** 127–134 (2013)
5. Suriadi, S., Andrews, R., ter Hofstede, AHM, Wynn, MT: Padrões de imperfeição do log de eventos para mineração de processos: em direção a uma abordagem sistemática para limpar logs de eventos. Inf. Syst. **64,** 132-150 (2017)
6. Mans, RS, van der Aalst, WMP, Vanwersch, RJB, Moleman, AJ: Mineração de processos na área da saúde: desafios de dados ao responder perguntas frequentes. Em: Lenz, R., Miksch, S., Peleg, M., Reichert, M., Riaño, D., ten Teije, A. (eds.) KR4HC / ProHealth -2012. LNCS (LNAI), vol. 7738, pp. 140-153. Springer, Heidelberg (2013). https://doi.org/10.1007/978 10
7. Gschwandtner, T., Aigner, W., Miksch, S., G¨ ärtner, J., Kriglstein, S., Pohl, M., Suchy, N .: TimeCleanser: uma abordagem de análise visual para limpeza de dados orientados por tempo. In: i-KNOW, p. 18. ACM (2014)
8. Mans, RS, van der Aalst, WMP, Vanwersch, RJB: Problemas de qualidade de dados. Process Mining in Healthcare. SBPM, pp. 79-88. Springer, Cham (2015). https://doi.org/ 10.1007 / 978-3-319-16071-9 6
9. Mardia, KV, Jupp, PE: Estatísticas Direcionais. Wiley, Chichester (1999)
10. Adriansyah, A., van Dongen, BF, van der Aalst, WMP: Rumo a verificação de conformidade robusta. In: zur Muehlen, M., Su, J. (eds.) BPM 2010. LNBIP, vol. 66, pp. 122–133. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-64220511-8 11
11. Mannhardt, F., Blinde, D .: Analisando as trajetórias de pacientes com sepse usando mineração de processo. RADAR + EMISA **1859,** 72–80 (2017)
12. Leemans, SJJ, Fahland, D., van der Aalst, WMP: Descobrindo modelos de processos estruturados em blocos de logs de eventos contendo comportamento infrequente. In: Lohmann, N., Song, M., Wohed, P. (eds.) BPM 2013. LNBIP, vol. 171, pp. 66–78. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-06257-0 6

13. van der Aalst, WMP, Weijters, AJMM, Maruster, L .: Mineração de fluxo de trabalho: descobrindo modelos de processo a partir de logs de eventos. IEEE Trans. Knowl. Data Eng. **16 (** 9), 1128-1142 (2004)

14. Muñoz-Gama, J., Carmona, J .: Um novo olhar sobre a precisão na conformidade do processo. Em: Hull, R., Mendling, J., Tai, S. (eds.) BPM 2010. LNCS, vol. 6336, pp. 211–226. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15618-2 16
15. Wang, RY, Storey, V., Firth, C .: A framework for analysis of data quality research. IEEE Trans. Knowl. and Data Eng. **7 (** 4), 623-640 (1995)
16. Batini, C., Palmonari, M., Viscusi, G .: Abrindo o mundo fechado: um levantamento da pesquisa de qualidade da informação na natureza. Em: Floridi, L., Illari, P. (eds.) The Philosophy of Information Quality. SL, vol. 358, pp. 43-73. Springer, Cham (2014). https: // doi.org/10.1007/978-3-319-07121-3 4
17. Laranjeiro, N., Soydemir, SN, Bernardino, J .: Um inquérito sobre a qualidade dos dados: classificação de dados insatisfatórios. In: PRDC, pp. 179-188. IEEE (2015)
18. Wand, Y., Wang, RY: Ancorar dimensões de qualidade de dados em fundações ontológicas. Comum. ACM **39 (** 11), 86-95 (1996)
19. ISO / IEC FDIS 25012: Engenharia de software - requisitos e avaliação de qualidade de produto de software - modelo de qualidade de dados (2008)
20. Günther, CW, van der Aalst, WMP: Fuzzy mining - simplificação de processo adaptativo com base em métricas de múltiplas perspectivas. In: Alonso, G., Dadam, P., Rosemann, M. (eds.) BPM 2007. LNCS, vol. 4714, pp. 328-343. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-75183-0 24
21. Swennen, M., Janssenswillen, G., Jans, M., Depaire, B., Vanhoof, K .: Capturando o comportamento do processo com métricas de processo baseadas em log. In: SIMPDA (2015)
22. Kherbouche, MO, Laga, N., Masse, PA: Para uma melhor avaliação da qualidade dos logs de eventos. Em: 2016 IEEE SSCI, pp. 1-8, dezembro de 2016
23. Rogge-Solti, A., Mans, RS, van der Aalst, WMP, Weske, M .: Melhorar a documentação reparando os logs de eventos. Em: Grabis, J., Kirikova, M., Zdravkovic, J., Stirna, J. (eds.) PoEM 2013. LNBIP, vol. 165, pp. 129–144. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-41641-5 10
24. Rogge-Solti, A., Mans, RS, van der Aalst, WMP, Weske, M .: Reparando logs de eventos usando modelos de processo cronometrados. In: Demey, YT, Panetto, H. (eds.) OTM 2013. LNCS, vol. 8186, pp. 705–708. Springer, Heidelberg (2013). https://doi.org/10. 1007 / 978-3-642-41033-8 89
25. Song, W., Xia, X., Jacobsen, HA, Zhang, P., Hu, H .: Recuperação heurística de eventos ausentes em logs de processo. In: ICWS, pp. 105-112, junho de 2015
26. Chu, X., Ilyas, IF, Papotti, P .: Limpeza holística de dados: colocando as violações no contexto. In: IEEE ICDE, pp. 458-469, abril de 2013
27. Song, S., Cao, Y., Wang, J .: Limpando carimbos de data / hora com restrições temporais. Proc. VLDB Endow. **9 (** 10), 708-719 (2016)
28. Conforti, R., Rosa, ML, ter Hofstede, AHM: Filtrando comportamento infrequente de logs de eventos de processos de negócios. IEEE Trans. Knowl. Data Eng. **29 (** 2), 300-314 (2017)