

Uma Implementação Distribuída em Névoa do Algoritmo de Detecção de Novidade em Fluxos de Dados MINAS

Luís Henrique Puhl de Souza

05 Julho 2021

Universidade Federal de São Carlos
Centro de Ciências Exatas e de Tecnologia
Departamento de Computação
Programa de Pós-Graduação em Ciência da Computação

Orientador: *Prof. Dr. Hermes Senger*

Obrigado CNPq pelo suporte financeiro (contrato 167345/2018-4).

Introdução

Contexto

- Crescimento do número de dispositivos IoT e riscos associados;
 - Heterogeneidade de dispositivos;
 - Falta de atualizações de *software*;
 - Exemplo: *Botnet* mirai, infectando cameras e roteadores, gerou 620 Gb/s (??).
- Detecção de intrusão em redes:
 - detecção por assinatura *versus* anomalia;
 - ambiente de névoa e redes IoT.

Contexto

- Crescimento do número de dispositivos IoT e riscos associados;
 - Heterogeneidade de dispositivos;
 - Falta de atualizações de *software*;
 - Exemplo: *Botnet* mirai, infectando cameras e roteadores, gerou 620 Gb/s (??).
- Detecção de intrusão em redes:
 - detecção por assinatura *versus* anomalia;
 - ambiente de névoa e redes IoT.

Proposta

- Um sistema para detecção de intrusão em Redes IoT implementando em névoa;
- A hipótese do trabalho é que o algoritmo MINAS pode ser distribuído em névoa reduzindo a latência sem redução na qualidade de classificação.

`figures/mfog-arch-fisica.png`

Ambientes de computação Distribuída

- **Computação em Nuvem** (*Cloud Computing*) é um modelo que permite acesso conveniente a recursos computacionais compartilhados (??).
Características: Auto-serviço sob demanda, Amplo acesso à rede, Agrupamento de recursos, Rápida elasticidade, Serviço mensurado;
- **Computação de Borda** (*Edge Computing*) refere-se a qualquer recurso computacional entre os dispositivos de borda e centro de dados hospedados em nuvem (??).
- **Computação em Névoa** (*Fog Computing*) é uma arquitetura horizontal a nível de sistema que distribui funções de computação, armazenamento, controle e rede próximos aos usuários no espaço contínuo nuvem-coisa (??).
Características: Mobilidade, Heterogeneidade, Baixa Latência, Distribuição geográfica, Alto número de nós, Interoperabilidade e federação, Uso de fluxo de dados e aplicações em tempo real.

Definição de Fluxo de Dados

Um fluxo de dados (*Data Stream*) é uma sequência massiva possivelmente ilimitada de exemplos multi-dimensionais $x_1, x_2, \dots, x_n, \dots$ recebidos em instantes associados $t_1, t_2, \dots, t_n, \dots$ (??).

Definição de Fluxo de Dados

Um fluxo de dados (*Data Stream*) é uma sequência massiva possivelmente ilimitada de exemplos multi-dimensionais $x_1, x_2, \dots, x_n, \dots$ recebidos em instantes associados $t_1, t_2, \dots, t_n, \dots$ (??).

Métodos Detecção de Novidade

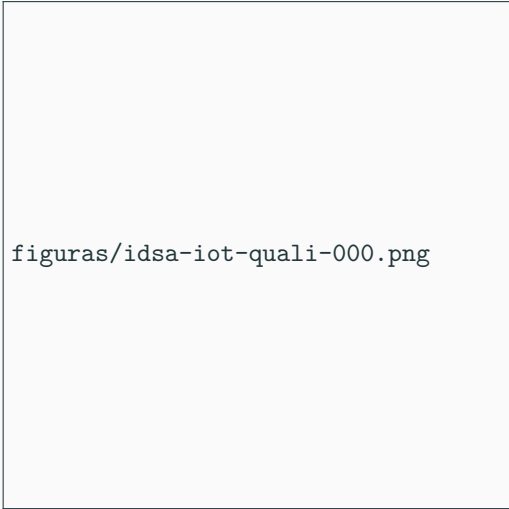
Métodos Detecção de Novidade (*Novelty Detection*) lidam com o reconhecimento e classificação de exemplos em padrões que diferem de padrões anteriores (??).

- Evolução de Conceito (*Concept Evolution*): surgimento de um conceito durante o fluxo;
- Mudança de Conceito (*Concept Drift*, deriva ou desvio): modificação da distribuição de um padrão conhecido. A modificação pode ser repentina, incremental ou recorrente;
- Ruído e *Outliers*: que não pertencem a um conceito ou pertencem a um conceito porém estão fora da distribuição conhecida.

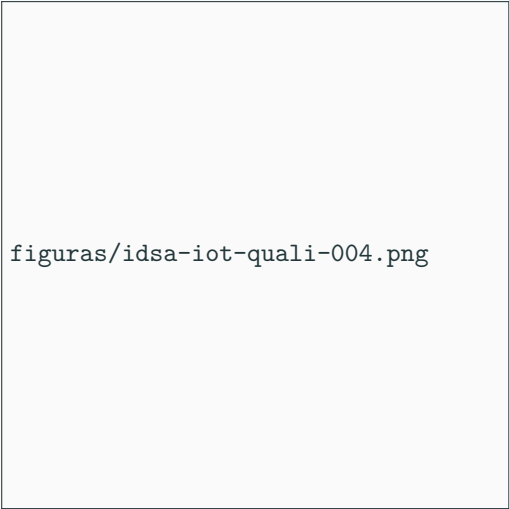
Estado da Arte e Trabalhos Relacionados

Sistemas de detecção de intrusão em redes

- Ferramenta BigFlow (??):
 - Sistema de detecção de intrusão por anomalia para redes de alta velocidade;
 - + Integração da extração dos descritores de fluxo à emissão de alarmes;
 - + Capacidade de tratamento de grandes volumes;
 - Atualização semanal com avaliação de um especialista;
 - Execução somente em nuvem.
- Ferramenta CATRACA (????):
 - Sistema de monitoração e detecção de ameaça com processamento de fluxos e NVF;
 - + Divisão em camadas alocadas em nuvem e névoa;
 - + Modelo de decisão baseado em árvore de decisão;
 - Extração dos descritores de fluxo é feita em névoa, classificação e detecção é feita em nuvem.
- Arquitetura IDSA-IoT (??):
 - + Avaliação do algoritmo MINAS, ECSMiner e AnyNovel;
 - + Distribuição das tarefas em nuvem e névoa focada em IoT;
 - Implementação e detalhamento da arquitetura em aberto.



figuras/idsa-iot-quali-000.png



figuras/idsa-iot-quali-004.png

Figura 2: Distribuição de serviços da arquitetura IDSA-IoT.

Fonte: ??).

Proposta

O algoritmo MINAS

- Análise no espaço \mathbb{R}^d ;
- Aprendizado *Offline-Online*;
- Classificação com *Clusters* e distância euclideana;

figuras/FariaMinas2015-off.png

figuras/FariaMinas2015-on.png

O algoritmo MINAS

- Agrupamento para identificação de novos padrões, tratando recorrência, extensão e novidade;

`figuras/FariaMinas2015-ndd.png`

```
1 Função MinasOnline(Modelo, fluxoEntrada, fluxoSaida, janelaLimpeza, gatilhoDetecçãoNov):
2   Desconhecidos  $\leftarrow \emptyset$ ; ModeloAntigo  $\leftarrow \emptyset$ ; últimaLimpeza  $\leftarrow 0$ ; proximaNovidade  $\leftarrow 0$ ;
3   para cada exemploi  $\in$  fluxoEntrada faça
4     maisPróximo  $\leftarrow$  clusterMaisPróximo (exemplo, Modelo);
5     se maisPróximo.distância < maisPróximo.cluster.raio então
6       exemplo.rótulo  $\leftarrow$  maisPróximo.cluster.rótulo;
7       maisPróximo.cluster.últimoUso  $\leftarrow i$ ;
8     senão
9       exemplo.rótulo  $\leftarrow$  “desconhecido”;
10    Desconhecidos  $\leftarrow$  Desconhecidos  $\cup$  exemplo;
11    se  $| \text{Desconhecidos} | \geq \text{gatilhoDetecçãoNov}$  então
12      Modelo  $\leftarrow$  Modelo  $\cup$  DetecçãoNovidade (Modelo  $\cup$  ModeloAntigo, *Desconhecidos);
13    se  $i > (\text{últimaLimpeza} + \text{janelaLimpeza})$  então
14      Modelo  $\leftarrow$  moveModeloAntigo (Modelo, *ModeloAntigo, últimaLimpeza);
15      Desconhecidos  $\leftarrow$  removeExemplosAntigos (Desconhecidos, últimaLimpeza);
16      últimaLimpeza  $\leftarrow i$ ;
17  fluxoSaída.adicione(exemplo);
```

Algoritmo 1: Interpretação do algoritmo MINAS *online* (??).

- Arquiteturas *Lambda* e *Kappa*, focadas em aplicações tradicionais;

A diagrama da arquitetura Lambda, que mostra o fluxo de dados entre diferentes componentes de processamento distribuído.

figuras/lambda.png

Figura 5: Arquitetura *Lambda* com Kafka, Storm, Hadoop, SGBD tradicional e aplicação consumidora.
Fonte: ??).

- Mineração de Dados:

Pergunta de Pesquisa

- É viável paralelizar e distribuir o algoritmo MINAS seguindo a arquitetura IDSA-IoT?
- Quais são os efeitos na qualidade de classificação se distribuir o algoritmo MINAS?

Proposta da Pesquisa

- Implementar uma versão distribuída do algoritmo MINAS conforme arquitetura IDSA-IoT;
- Paralelizar o método de classificação do algoritmo MINAS;
- Avaliar a implementação quanto à viabilidade e qualidade.

Proposta da Pesquisa

- Implementar uma versão distribuída do algoritmo MINAS conforme arquitetura IDSA-IoT;
- Paralelizar o método de classificação do algoritmo MINAS;
- Avaliar a implementação quanto à viabilidade e qualidade.

Método

- Plataforma de processamento distribuído;
- Estratégias de implementação da arquitetura IDSA-IoT;
- Experimentação com a distribuição do algoritmo MINAS em ambiente específico;
- Métricas de qualidade de classificação para validação da implementação;
- Métricas de escalabilidade.

Primeira Implementação com *Python* e *Apache Kafka*

- *Python* é acessível e fornece bibliotecas diversas;
- *Apache Kafka* é um sistema de mensagens distribuído;
 - Interface de programação com cliente produtor e consumidor;
 - Mensagens organizadas em tópicos que são distribuídos em partições;
- A hipótese de que a carga seria distribuída entre os consumidores, uma vez que o consumidor pode selecionar uma partição para leitura;



Figura 6: Partições em Apache Kafka.
Fonte: ??).

- Em experimento com um produtor, 8 partições e 8 consumidores, observou-se que um

Segunda Implementação com *Apache Flink*

- Implementação escrita em Scala ou Java;
- Processamento de fluxos *Stateful*;



`figuras/dataflow-code-flink.png`

Terceira Implementação com MPI

- Maior controle sobre a implementação e execução;
- Implementado em linguagem C, OpenMPI 4.0.4, seguindo a técnica SPMD;
- Dividido em 2 módulos e 4 tarefas.
- `mpirun` cria processos, o processo de 0 executa o módulo raiz e os demais processos executam o módulo folha;
- Módulo raiz, com as tarefas Fonte e Detector, trata dos fluxos de entrada e saída além de gerenciar o conjunto de desconhecidos e a detecção de novidade;
- Módulo folha, com as tarefas Classificador e Atualiza Modelo, trata da classificação de cada exemplo e manutenção do modelo local de cada instância.

figures/lifecycle-uml-svg.pdf

Métricas e Ambientes

- Métricas de qualidade de classificação:
 - Avaliação do fluxo de saída do classificador em uma matriz de confusão própria;
 - Taxa de desconhecidos, acurácia e erro por classe.

$$\mathbf{C} = \{c_1, c_2, \dots, c_m\} \quad (1)$$

$$\mathbf{Y} = \{y_1, y_2, \dots, y_k\} \quad (2)$$

$$\mathbf{L} = \{l_1, l_2, \dots, l_n\} = \mathbf{C}' \cup \{"-\"} \cup \mathbf{Y} \quad (3)$$

$$\mathbf{E}_x = (e_{ij}) \in \mathbb{N}^{m \times n} \quad (4)$$

$$A(l_j) = \begin{cases} \nexists & \text{se } l_j = "-"} \\ c_i & \text{se } \exists c_i = l_j : c_i \in \mathbf{C}' \\ c_i & \text{se } e_{ij} = \max\{e_{aj} : j \in [0, m]\} \end{cases} \quad (5)$$

$$UnkR_{x,i} = \frac{e_{ij} : l_j = "-"}{\sum_{j=1}^n e_{ij}} \quad (6)$$

$$tp_i = \sum_{j=1}^n e_{ij} \text{ se } l_j \neq "- \text{ e } A(l_j) = c_i \quad (7)$$

$$fn_i = \sum_{j=1}^n e_{ij} \text{ se } l_j \neq "- \text{ e } A(l_j) \neq c_i \quad (8)$$

$$acc_x = \frac{1}{m} \sum_{i=1}^m \frac{tp_i}{fn_i + tp_i} \quad (9)$$

$$err_x = \frac{1}{m} \sum_{i=1}^m \frac{fn_i}{fn_i + tp_i} \quad (10)$$

Métricas e Ambientes

- Métricas de escalabilidade:
 - Número e tipo de processadores;
 - Uso de memória;
 - Tempo de processamento;
 - Latência, tempo entre a entrada e saída de cada descritor de fluxo.
- Ambientes de teste:
 - Computador Pessoal (para desenvolvimento);
 - Névoa composta de SBC (*Single Board Computer*) ARM 4 núcleos;
 - Conjunto de dados para IDS, Kyoto 2006+, segmento dezembro de 2015 como estabelecido por ??).

Resultados


Tabela 1: Listagem dos principais experimentos.

Experimento	Programa	Características
<i>a-Referência</i>	MINAS referência 2013	Raio é a distância máxima.
<i>b-Sequencial</i>	MINAS sequencial para validação	Raio é o desvio padrão das distâncias; Modelo único; Remoção de desconhecidos mais agressivo.
<i>c-Paralelo</i>	M-FOG 1 nó, 4 processadores	Classificadores paralelos; Detecção de novidade assíncrona.
<i>d-Distribuído</i>	M-FOG 3 nós, 12 processadores	Mais processadores; Comunicação em rede.



experiments/revise-java-log.png


(a) Experimento *a-Referência*, implementação de referência do algoritmo MINAS.



experiments/online-nd-log.png

(b) Experimento *b-Sequencial*, M-FOG sequencial.

Figura 9: Visualização de fluxo do conjunto *Kyoto* Dez. 2015.



experiments/tmi-base-log.png

(a) Experimento *c-Paralelo*, M-FOG com 1 nó e 4 núcleos.



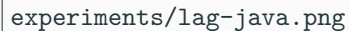
experiments/tmi-n12-log.png

(b) Experimento *d-Distribuído*, M-FOG com 3 nós de 4 núcleos cada.

Tabela 2: Sumário das métricas extraídas dos experimentos principais.

Experimento Métrica	<i>a-Referência</i>	<i>Offline</i>	<i>b-Sequencial</i>	<i>c-Paralelo</i>	<i>d-Distribuído</i>
unk	0.018333		0.043717	0.023521	0.023718
hit	0.305618		0.298438	0.312416	0.312478
err	0.676049		0.657843	0.664061	0.663802
Novidades	12		9	5	5
Tempo (s)	2761.83	194.12	80.79	522.10	207.14
Sistema (s)	7.15	0.075	11.51	47.77	157.61
Decorrido (s)	2772.07	194.27	93.03	145.04	95.38
Latência (s)	$4.24 \cdot 10^{-3}$		$1.42 \cdot 10^{-4}$	$2.22 \cdot 10^{-4}$	$1.46 \cdot 10^{-4}$
Processadores	1	1	1	4	12
<i>Speedup</i>				0.6414092	0.9753617
Eficiência				0.1603523	0.0812801

experiments/speedup-clean.pdf



experiments/lag-java.png

(a) Implementação de referência.



experiments/lag-serial.png

(b) Implementação sequencial.

Figura 12: Visualização de Latência.

Fonte: O autor.

experiments/lag-mfog.png

Conclusão

Resultados obtidos:

- Algoritmo MINAS distribuído e a arquitetura IDSA-IoT implementada com modificações;
- Distribuição tem pequeno efeito sobre as métricas de qualidade;
 - Maior efeito é a redução de etiquetas novidade na versão distribuída;
- Resultados mostram que a implementação M-FOG não atinge escala pelo CCR e eficiência;

Resultados obtidos:

- Algoritmo MINAS distribuído e a arquitetura IDSA-IoT implementada com modificações;
- Distribuição tem pequeno efeito sobre as métricas de qualidade;
 - Maior efeito é a redução de etiquetas novidade na versão distribuída;
- Resultados mostram que a implementação M-FOG não atinge escala pelo CCR e eficiência;

Trabalhos futuros:

- Da arquitetura: Distribuição do modelo entre redes distintas (conjuntos aditivos);
- Na implementação:
 - Outros algoritmos de agrupamento (CluStream);
 - Estratégia de otimização da distribuição de carga (micro ou mini batching);
 - Outras plataformas de processamento otimizadas para o ambiente névoa;
- No algoritmo:
 - Explorar distribuição espacial dos clusters (polígonos sem sobreposição, árvore de busca);
 - Algoritmo com modelo de tamanho fixo (máxima precisão com recursos disponíveis);

- Artigo aceito na trilha principal da 21ª Conferência Internacional em Computação Científica e suas Aplicações (ICCSA 2021, <https://iccsa.org/>) em Cagliari, Itália, Setembro 13-16 2021 (??);
- Código fonte com experimentos e métodos publicamente disponíveis em <https://github.com/luis-puhl/minas-flink>.

Obrigado!

figures/arq-mfog.png

figures/mfog-arch-fisica.svg.pdf

figures/lifecycle-uml-svg.pdf

experiments/params.png

experiments/revised-java-log.png

Tabela 3: Experimento *a-Referência*, Matriz de confusão, *Kyoto* Dez. 2015.

Rótulos	-	N	1	2	3	4	5	6	7	8	9	10	11	12
Classes														
A	3 774	438 750	123	145	368	8	52	165	1	1 046	161	2 489	71	26
N	8 206	193 030	0	79	44	0	0	0	229	181	154	4 066	289	0
Associação	-	N	A	A	A	A	A	A	N	A	A	N	N	A
Hits (<i>tp</i>)	0	193 030	123	145	368	8	52	165	229	1 046	161	4 066	289	26

experiments/online-nd-log.png

Tabela 4: Experimento *b-Sequencial*, Matriz de confusão, *Kyoto* Dez. 2015.

Rótulos	-	N	0	1	2	4	5	6	7	8	10
Classes											
A	16 086	429 765	94	995	104	0	23	3	29	46	34
N	12 481	193 642	3	94	0	47	0	0	0	11	0
Associação	-	N	A	A	A	N	A	A	A	A	A
Hits (<i>tp</i>)	0	193 642	94	995	104	47	23	3	29	46	34

experiments/tmi-base-log.png

Tabela 5: Experimento *c-Paralelo*, M-FOG com 1 nó e 4 núcleos, Matriz de confusão, *Kyoto Dez.* 2015.

Rótulos	-	N	0	1	2	3	4
Classes							
A	12 282	433 797	147	952	0	0	1
N	3 088	203 019	40	99	27	5	0
Associação	-	N	A	A	N	N	A
Hits (<i>tp</i>)	0	203 019	147	952	27	5	1

experiments/tmi-n12-log.png

Tabela 6: Experimento *d-Distribuído*, M-FOG com 3 nós de 4 núcleos cada, Matriz de confusão, *Kyoto* Dez. 2015.

Rótulos	-	N	0	1	2	3	4
Classes							
A	12 378	433 631	117	886	0	162	5
N	3 121	202 916	40	96	105	0	0
Associação	-	N	A	A	N	A	A
Hits (<i>tp</i>)	0	202 916	117	886	105	162	5

experiments/speedup-clean.pdf

experiments/lag-java.png

experiments/lag-serial.png

experiments/lag-mfog.png