FAS-Bus: Um Sistema de Análise da Frota de Ônibus Urbanos

Fernando D. M. Silva[1], **Matheus F. Tavares**[1], **Pedro Cruz**[1,2], **Rodrigo S. Couto**[1] **e Luís Henrique M. K. Costa**[1]

[1]Universidade Federal do Rio de Janeiro - PEE-COPPE/DEL-Poli/GTA
Caixa Postal 68.504 – 21.941-972 – Rio de Janeiro – RJ – Brasil

[2]Institut National de Recherche en Sciences et Technologies du Numérique - Inria
Saclay – Ile de France – França

{fernandodias,felinto,cruz,rodrigo,luish}@gta.ufrj.br

Abstract. *In order to evaluate the performance of public transit bus systems, the constant monitoring of each bus line is nedded. This process, when done manualy, inflicts in costs and ineficiency. This article proposes FAS-Bus (Fleet Analysis System for urban Buses), a system that can expedite this process by taking advantage of the bus tracking by GPS. The system consists of modules that collect, store, correct and allows visualization of the trajectory data. The results are statistical data of a bus system that can be used to measure, for example, the changes in fleet size and service level for a given line. The system is demonstrated with the use of the Rio de Janeiro city's fleet data.*

Resumo. *Para avaliar o desempenho do sistema de transporte de ônibus urbanos, é necessário o monitoramento constante da operação de cada linha. Esse processo, executado manualmente, é custoso e ineficiente. Neste artigo é proposto o FAS-Bus (Fleet Analysis System for urban Buses), um sistema que visa agilizar esse processo tirando proveito da localização dos ônibus por GPS. O FAS-Bus é composto de módulos que fazem a coleta, armazenamento, correção e visualização das trajetórias. Os resultados são os dados estatísticos de um sistema de ônibus, que podem ser usados para avaliar, por exemplo, mudanças de tamanho de frota e nível de serviço das linhas. O sistema é demonstrado com o uso dos dados da frota de ônibus da cidade do Rio de Janeiro.*

1. Introdução

Com o crescimento das cidades, seus sistemas de transporte por ônibus tornam-se cada vez maiores e mais complexos, o que dificulta e encarece processos de auditoria. O uso de recursos humanos para a análise da eficiência desses sistemas de transporte pode acarretar em imprecisões nos dados e lentidão. Dentro do contexto da Internet das coisas (IoT), é possível utilizar dispositivos de baixo custo conectados à Internet para a coleta de dados. Com o auxílio desses dispositivos, os sistemas de transporte podem ser analisados facilmente e de forma mais econômica. A posição dos ônibus pode ser coletada de maneira regular para ser disponibilizada em outros serviços. Além disso, o armazenamento desses dados permite a obtenção de parâmetros estatísticos que fornecem uma visão geral do desempenho da frota como um todo.

O baixo custo de dispositivos IoT vem com a desvantagem a sua baixa confiabilidade, o que pode penalizar a análise. Outro fator a se considerar é se a intervenção

humana é necessária para configuração de valores, o que a torna propensa a erros. Por exemplo, a linha servida por um ônibus é frequentemente preenchida de maneira manual. Esses fatores põem em risco a análise precisa, e para isso é necessária uma forma de validação mais confiável antes de os dados serem usados para fins estatísticos.

Com o objetivo de gerenciar e validar os dados coletados, desenvolve-se o FAS-Bus (*Fleet Analysis System for urban Buses*), um sistema de análise estatística para frotas de ônibus. O sistema é responsável por gerenciar o processo de coleta e processamento de dados provenientes dos veículos rastreados. Além disso, o sistema oferece uma interface de visualização livre para que usuários possam conferir os resultados de análise. A equipe de administração do FAS-Bus pode passar os dados por um processo de correção de erros de classificação proveniente dos dispositivos IoT e, assim, prover uma análise mais precisa. A correção é feita com o uso de um algoritmo vetorizado, utilizando bibliotecas que permitem o uso de placas gráficas para acelerar o processo de correção. O FAS-Bus é modular, podendo ser distribuído e replicado em diversas máquinas e GPUs, aumentando a robustez do sistema para suportar o processo de correção de frotas de tamanho elevado.

Para aplicar o FAS-Bus em um estudo de caso real, neste trabalho são utilizados os dados de localização dos ônibus da cidade do Rio de Janeiro. Os dados das trajetórias dos ônibus são coletados desde outubro de 2018 e parte desses dados são utilizados para fazer a avaliação do sistema.

Este trabalho está estruturado da seguinte forma. A Seção 2 discute trabalhos relacionados. A Seção 3 provê uma visão geral das funcionalidades do sistema. A Seção 4 detalha a arquitetura do sistema, funções e requisitos de cada módulo. A Seção 5 apresenta um teste de desempenho e mostra o sistema em execução. As conclusões são descritas na Seção 6.

2. Trabalhos Relacionados

O FAS-Bus se baseia na integração dos ônibus à Internet das Coisas. É possível obter atualizações constantes da posição de todos os ônibus da frota com sensores de localização GPS e conectividade via LoRAWAN ou GSM, por exemplo. Exemplos desse sistema são encontrados em [Farooq et al. 2010], [Boshita et al. 2018] e [Kadam et al. 2018].

O FAS-Bus tem o mesmo objetivo de [Nguyen et al. 2018], que descreve um sistema que avalia as condições da frota dos ônibus de Los Angeles, EUA, via métricas estatísticas. O FAS-Bus se diferencia por receber dos ônibus menos informações por entrada e se basear nessas entradas para inferir em nuvem outros dados como linha percorrida. Assim, o FAS-Bus é mais adequado para infraestruturas IoT de baixo custo e que têm alta taxa de erros em sua classificação.

O FAS-Bus pode servir como uma ferramenta para gerenciar sistemas que utilizem a frota para implementação de novos serviços em cidades inteligentes. Trabalhos como [Cruz et al. 2018] e [Zambada et al. 2015] podem utilizar o FAS-Bus como base para agregação de informações, e adaptar a estrutura de dados para incorporar dados específicos de cada trabalho. A correção oferecida pelo sistema pode ser usada por [Celes et al. 2020] para auxiliar no processo de interpolação da movimentação dos ônibus em sistemas que a classificação de linhas esteja incorreta ou inexistente.

3. Funcionalidades do FAS-Bus

O FAS-Bus gerencia todo o processo de armazenamento, correção e visualização de dados proveniente de uma frota de ônibus rastreável. A sua arquitetura consiste em diferentes aplicações separadas em contêineres que exercem as diferentes etapas desse processamento. Graças a essa separação, o FAS-Bus pode ser facilmente adaptado para servir em diferentes estruturas de rede.

Processos de pré-processamento que visam mitigar erros simples, como localizações e IDs inválidos, podem não ser suficientes para o tratamento dos dados do FAS-Bus. Classificações mais complexas, como a linha que o ônibus percorre, requerem um algoritmo de detecção e correção mais robusto. O processo de correção dessas trajetórias, pensado para frotas de larga escala, faz o uso de uma nova implementação do algoritmo proposto em [Silva et al. 2020]. A nova implementação utiliza vetorização e aceleração por hardware com o uso de GPUs para reclassificar todas as entradas de ônibus, baseando-se apenas na sua trajetória. Esse processo pode ser distribuído em múltiplos sistemas e múltiplas GPUs, o que permite o ajuste de recursos de modo a acompanhar a demanda de frotas de ônibus de larga escala.

O FAS-Bus possui uma interface API que serve como consulta dos resultados obtidos. Essa interface oferece uma forma segura e simples de expor os resultados publicamente. A API oferece diferentes tipos de consultas definidas pela equipe de administração do sistema, que também é responsável por criar novas análises caso o sistema seja adaptado para servir frotas com diferentes disposições de dados.

O processo de administração do sistema deve considerar a adaptação do módulo de inserção para ser compatível com seu respectivo cenário. Uma vez implementado, deve-se gerenciar a política de armazenamento dos dados e controle da parcela destinada à correção. O resultado pode ser visualizado por qualquer usuário por meio da API do FAS-Bus.

4. Arquitetura do Sistema

O FAS-Bus é divido nos seguintes módulos: inserção, banco de dados, visualização e correção. Cada módulo é construído com um ou mais contêineres Docker[1] separados que permitem que a aplicação possa ser distribuída e paralelizada. Com isso as únicas dependências das máquinas que terão o sistema implementado são possuir sistema operacional Linux e Docker. Para a máquina que fará o processamento, tem-se o adicional de necessitar de uma GPU NVIDIA[2] compatível com as bibliotecas CUDA[3]. É possível visualizar a estrutura na Figura 1.

Cada módulo possui uma função e característica única. São eles:

- **Inserção:** Módulo que faz a coleta e inserção dos dados no banco de dados.
- **Banco de dados:** Tabela relacional que serve de estrutura para armazenar as trajetórias de todos os ônibus e todas as linhas de ônibus.
- **Correção:** Módulo que realiza a reclassificação da linha percorrida pelos ônibus para cada entrada.

[1]https://www.docker.com/
[2]https://www.nvidia.com/
[3]https://developer.nvidia.com/cuda-zone/

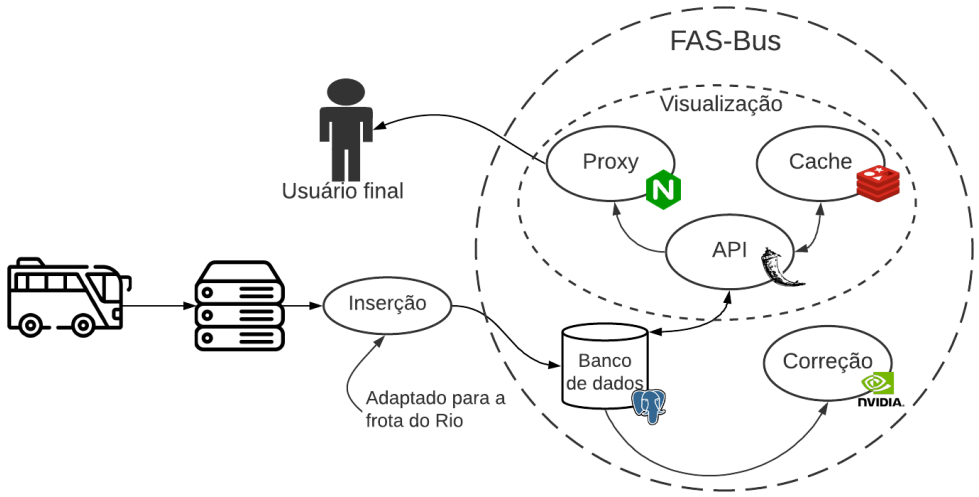

Figura 1. Arquitetura do sistema FAS-Bus.

- **Visualização:** Módulo que fornece uma API para visualização dos dados disponíveis no banco de dados, um cache para os resultados e um site estático com informações da ferramenta.

4.1. Banco de dados

O módulo de banco de dados é responsável pelo armazenamento dos dados, permitindo a organização dos dados coletados. Além disso, o banco de dados permite que todos os resultados estatísticos de interesse possam ser obtidos através de consultas diretas. Para a implementação no FAS-Bus, foi utilizado o banco de dados PostgreSQL[4].

A tabela que armazena a trajetória dos ônibus é dividida em partições para facilitar a busca de dados e permitir seu gerenciamento com mais facilidade e rapidez, ficando à escolha da administração a quantidade de dados que são retidos na tabela. A estrutura do banco de dados e alguns exemplos de dados podem ser visualizados na Figura 2.

O banco de dados também armazena informações importantes para o módulo de correção, como o itinerário das linhas. Na administração do sistema, pode-se reestruturar as tabelas para adicionar os dados que se tenha interesse.

4.2. Módulo de correção

O FAS-Bus é voltado para sistemas IoT de baixo custo que são passíveis de erros na hora de classificar as trajetórias dos dados enviados. Com esse cenário, o módulo de correção faz a reclassificação da informação de linha percorrida das entradas na tabela. A correção é alcançada ao fazer a comparação entre todas as trajetórias de ônibus e de linhas de ônibus e definir, para cada coordenada do ônibus, a qual linha ele pertence.

O processo descrito anteriormente torna-se árduo em sistemas de transporte de larga escala. Para lidar com esse cenário, o FAS-Bus implementa o algoritmo de correção em Python3[5] com o uso da biblioteca CuPy[6], que realiza o processamento paralelo com

[4]https://www.postgresql.org/
[5]https://www.python.org/
[6]https://cupy.dev/

Tabela ônibus					
Data/Hora	Identificador	Latitude	Longitude	Linha reportada	Linha detectada
2021-03-05T15:40:03	"AA0001"	-22.85964	-43.22882	"001"	"002"
2021-03-05T15:40:03	"BB0030"	-22.85964	-43.22882	"040B"	"040B"

Tabela linhas				
Identificador	Direção	índice	Latitude	Longitude
"001"	Ida	2	-22.85964	-43.22882
"002"	Volta	13	-22.85964	-43.22882

Figura 2. Estrutura da tabela do banco de dados com exemplos.

uso de GPU. O módulo pode ser distribuído e aproveitar o processamento de múltiplas instâncias executando em diferentes máquinas. É necessário que em cada máquina exista uma placa e os respectivos drivers atualizados.

4.2.1. Algoritmo de correção

O algoritmo consiste em fazer a comparação ponto-a-ponto de cada trajetória do ônibus com trajetória de linha, entre todos os ônibus e linhas. Isso produz um resultado para cada entrada que previamente estava armazenada no banco de dados. Para isso, o processo é dividido em duas etapas: detecção e correção.

A detecção consiste em uma análise por ônibus, na qual a trajetória de cada ônibus é comparada com as coordenadas de todas as linhas. Para cada linha, calcula-se um fator de pertencimento, que é a razão entre a extensão da linha na qual o ônibus percorreu e seu comprimento total. Quando essa razão se encontra acima de um nível de tolerância, define-se a linha como pertencente àquele ônibus. O resultado dessa etapa é uma lista de comparações que é passada para o algoritmo de correção para que esse possa ser feito em um espectro bem menor de comparações. Esse processo foi publicado em [Silva et al. 2020].

O uso da GPU é fundamental na etapa de detecção. A GPU aproveita o fato de as comparações entre coordenadas retornarem resultados independentes. Dessa forma, os dados são organizados em matrizes, que podem ser processadas paralelamente com o uso das bibliotecas CUDA.

Uma vez na etapa de correção, faz-se a comparação temporal, para cada ônibus, entre cada coordenada dele e a trajetória de cada linha. Essa comparação resulta em um vetor de identificação para cada linha. Esse vetor determina, para cada coordenada da trajetória, seu pertencimento ou não à trajetória. Após essa etapa, os diferentes grupos de linha são comparados. Assim, realiza-se a resolução de conflitos, para eliminar dupla detecção (proveniente de linhas com trechos similares). Além disso, é feita a determinação de começo e fim de atuação baseado no deslocamento inicial do ônibus.

O resultado final desse processo substitui na íntegra (ou adiciona uma nova) a coluna de linha que o ônibus percorria no banco de dados de trajetória de ônibus. Esse atributo é então utilizado para fazer consultas que o necessitem como, por exemplo, de-

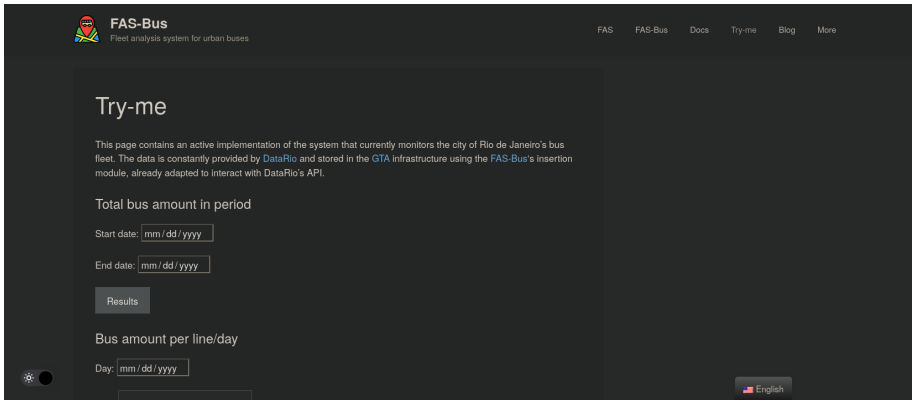

Figura 3. Exemplo da página de consulta disponibilizada pelo módulo de API.

terminar em um intervalo do dia quantos ônibus percorriam aquela linha.

4.3. Módulo de visualização

O módulo de visualização utiliza diversos contêineres para fornecer uma API que se comunica com o módulo de banco de dados e provê uma interface de livre acesso para a pesquisa dos resultados. Sua estrutura consiste em um contêiner Nginx[7] para servir de proxy reverso ao contêiner da API, que contém a CGI (Common Gateway Interface) em Python3 e a biblioteca Flask[8]. Essa última processa as requisições e faz as consultas ao banco de dados para retornar os dados. Uma vez feita a consulta, os dados são salvos no contêiner de cache, que usa o Redis[9] para agilizar as próximas consultas.

As consultas ao banco de dados são feitas com as informações básicas de identificação do ônibus e linha percorrida, para cada entrada dos ônibus no banco de dados. É responsabilidade da equipe de administração adaptar as consultas à tabela para servir outros tipos de frota que possam enviar diferentes dados. Com a estrutura da cidade do Rio de Janeiro e com a correção dos dados de linha é possível obter os seguintes dados:

- **Contingente da frota**: Mostra a contagem total de veículos que operaram em um dia junto com a quantidade máxima de ônibus operando simultaneamente.
- **Quantidade de ônibus por linha**: Mostra, ao longo de um dia, a quantidade de ônibus que participaram dessa linha. Esse gráfico possui duas curvas para mostrar a versões criadas com os dados diretamente reportados e (quando disponível) os dados corrigidos.
- **Parcela atuante por consórcio**: Um gráfico que mostra, em um dia específico, a quantidade de ônibus atuante separada por consórcios, que podem ser identificados usando uma parcela comum dos ônibus.

Além de oferecer a própria API, o módulo de proxy serve um site estático que oferece informações básicas do sistema, mas que pode ser alterado para servir a qualquer propósito desejado. A API tem a opção de retornar um gráfico interativo para ser exibido em páginas HTML usando a biblioteca Bokeh[10], como pode ser visto na Figura 4.

[7]https://www.nginx.com/
[8]https://flask.palletsprojects.com/en/2.0.x/
[9]https://redis.io/
[10]https://bokeh.org/

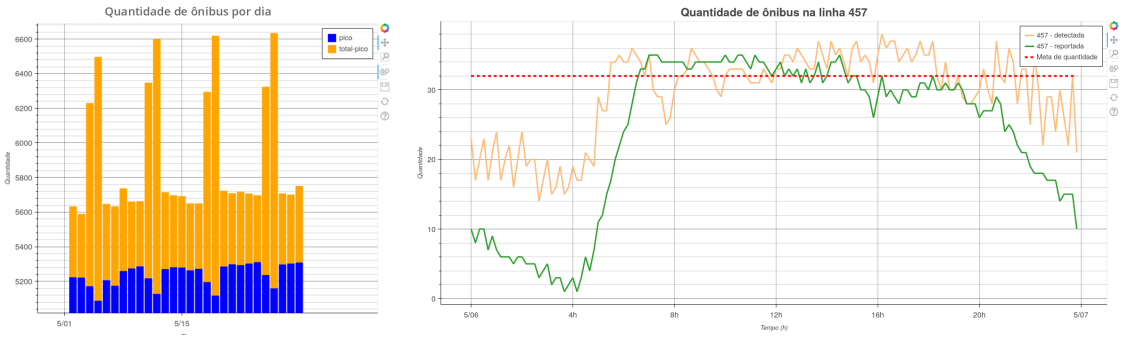

Figura 4. Gráficos de resultados obtidos pela ferramenta.

O site do FAS-Bus contém formulários[11] e código próprio para permitir que usuários interajam diretamente com uma implementação da API. Esta está situada no mesmo domínio, no diretório `/rj/`.

5. Análise do Sistema

O sistema possui maior demanda computacional em dois momentos: na consulta dos resultados estatísticos ao banco de dados e na correção dos dados. São feitas duas análises para avaliar o tempo de execução dessas tarefas com o uso dos dados da frota do Rio de Janeiro. Desse modo, é possível avaliar o sistema no cenário de aplicações de larga escala.

Para a primeira análise, utilizam-se os dados reais de trajetória de linhas e ônibus da frota do Rio de Janeiro para medir o tempo gasto. O dia escolhido foi o dia 6 de maio de 2019. Os dados processados continham 5.208 ônibus e 2.857 linhas, com o total aproximado de 4,7 milhões e 1,4 milhões de coordenadas inseridas no banco de dados, respectivamente. A máquina utilizada possui uma GPU RTX 2060 de 6GB de memória e um processador Intel I7-9770K. Os dados do banco de dados estão salvos em um SSD. O tempo total gasto é de aproximadamente 5 horas e 40 minutos, entre os processos de detecção e correção. Esse tempo é longo, mas é suficiente para fazer uma avaliação completa de todas as entradas de dados geradas pelos ônibus da cidade.

Outra avaliação se dá no tempo gasto para processar e gerar os resultados estatísticos. Sob as mesmas condições, o tempo máximo que cada consulta demora é de aproximadamente 4 segundos. Períodos de tempo maiores somam o tempo de um dia para serem processados. Porém, graças ao módulo de cache, esse tempo só é percebido no momento da primeira consulta e as consultas subsequentes são respondidas em menos de um segundo. Exemplos desses resultados podem ser visualizados na Figura 4.

6. Conclusão

Este trabalho apresentou o FAS-Bus, um sistema de gerenciamento, classificação de trajetórias e análise estatística dos dados de trajetória de frotas de ônibus que disponibilizam dados de localização GPS. Foram apresentadas suas características principais, como modularidade do sistema e interface para consulta pública.

O sistema é capaz de fazer a correção dos dados de classificação de linha por um algoritmo de correção, que é implementado usando vetorização e bibliotecas de

[11] https://fasbus.gta.ufrj.br/try-me/

aceleração por GPU. Mostrou-se que esse método robusto pode ser implementado e suportar a demanda de uma frota de ônibus de larga escala, como a do Rio de Janeiro.

Para a apresentação da ferramenta, o sistema será instalado em uma das máquinas do Grupo de Teleinformática e Automação e a apresentação será transmitida por meios virtuais. Os espectadores poderão ter acesso ao sistema. Mais informações, documentação e vídeo de instalação podem ser obtidos em https://fasbus.gta.ufrj.br e o código pode ser encontrado a partir do repositório https://github.com/Projeto-Onibus/FAS-Bus.

Referências

Boshita, T., Suzuki, H., and Matsumoto, Y. (2018). Iot-based bus location system using lorawan. In *21st Int. Conf. on Intelligent Transportation Systems (ITSC)*, pages 933–938. IEEE.

Celes, C., Boukerche, A., and Loureiro, A. A. F. (2020). Calibrating bus mobility data for bus-based urban vehicular networks. In *Proceedings of the 23rd International ACM Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, MSWiM '20, page 207–214, New York, NY, USA. Association for Computing Machinery.

Cruz, P., Silva, F. F., Pacheco, R. G., Couto, R. S., Velloso, P. B., Campista, M. E. M., and Costa, L. H. M. K. (2018). Sensingbus: Using bus lines and fog computing for smart sensing the city. *IEEE Cloud Computing*, pages 1–11.

Farooq, U., u. Haq, T., Amar, M., Asad, M. U., and Iqbal, A. (2010). Gps-gsm integration for enhancing public transportation management services. In *2nd International Conf. on Computer Engineering and Applications*, volume 2, pages 142–147.

Kadam, A. J., Patil, V., Kaith, K., Patil, D., and Sham (2018). Developing a smart bus for smart city using iot technology. In *2nd Int. Conf. on Electronics, Communication and Aerospace Technology (ICECA)*, pages 1138–1143.

Nguyen, K., Yang, J., Lin, Y., Lin, J., Chiang, Y.-Y., and Shahabi, C. (2018). Los angeles metro bus data analysis using gps trajectory and schedule data (demo paper). SIGSPATIAL '18, page 560–563, New York, NY, USA. Association for Computing Machinery.

Silva, F. D. M., Cruz, P., Couto, R. S., and Costa, L. H. M. K. (2020). Redução de inconsistências no monitoramento da frota de Ônibus urbanos. In *XXXVIII Simpósio Brasileiro de Telecomunicações e Processamento de Sinais*. SBrT.

Zambada, J., Quintero, R., Isijara, R., Galeana, R., and Santillan, L. (2015). An iot based scholar bus monitoring system. In *IEEE 1st Int. Smart Cities Conference (ISC2)*, pages 1–6. IEEE.