WiBus: Um Sistema de Monitoramento de Transportes Públicos Usando Redes IEEE 802.11

Vitor Borges C. da Silva, Tatiana Sciammarella, Miguel Elias M. Campista e Luís Henrique M. K. Costa *

[1]GTA/PEE-COPPE/DEL-Poli – Universidade Federal do Rio de Janeiro (UFRJ)
Caixa Postal 68504 – 21.941-972 – Rio de Janeiro – RJ – Brasil

{borges,tatiana,miguel,luish}@gta.ufrj.br

Abstract. *A great challenge today is to deal with the constant traffic jams related to excessive use of vehicles. It is believed that more people would adopt public transportation if these were more reliable. Thus, this paper proposes WiBus, which is a system to estimate buses times of arrival, based on information from opportunistic contacts in IEEE 802.11 networks. Estimates are provided to users through graphical interfaces of mobile devices. WiBus adjusts bus trajectories with an algorithm for dynamic creation and maintenance. The system was implemented and its performance was analyzed via emulation of a real scenario. Experimental results show that WiBus can meet demands of large cities with accumulated error on the order of a few minutes in the worst case.*

Resumo. *Um grande desafio atual é lidar com os engarrafamentos relacionados ao uso excessivo de veículos. Acredita-se que mais pessoas adotariam transportes públicos se estes fossem mais confiáveis. Nesse sentido, este trabalho propõe o WiBus, um sistema para estimar os horários de chegada dos ônibus, baseado em informações de contatos oportunistas em redes IEEE 802.11. As estimativas são disponibilizadas aos usuários através de interfaces gráficas de dispositivos móveis. O WiBus ajusta as trajetórias das linhas de ônibus dinamicamente. O sistema desenvolvido foi analisado através da emulação de um cenário real. Os experimentos mostram que o WiBus pode atender demandas de grandes cidades com erro acumulado de poucos minutos no pior caso.*

1. Introdução

As grandes metrópoles brasileiras vêm sofrendo com engarrafamentos constantes gerados pelo aumento do volume de veículos particulares. Em apenas um ano, de setembro de 2012 até setembro de 2013, quase 3 milhões de automóveis particulares entraram em circulação no Brasil, ocasionando uma piora sensível no trânsito [DENATRAN, 2013]. A preferência pelo transporte particular é uma consequência da falta de confiança da população nos meios de transporte de massa, seja por questões de segurança, conforto ou comprometimento com os horários. Para reverter essa tendência e atrair passageiros, os sistemas de transporte públicos devem aumentar a qualidade dos seus serviços. Esse desafio é um dos principais objetivos dos recentes sistemas avançados de transporte público (*Advanced Public Transportation Systems* - APTS) que,

*Este trabalho foi parcialmente financiado pela CAPES, CNPq, FAPERJ e FINEP.

dentre outras tecnologias, usam redes de comunicação para prover mais informações aos usuários [Casey et al., 1991].

As redes de comunicação em ambientes veiculares vêm sendo investigadas com o uso de tecnologias que vão desde as de telefonia celular (3G/4G) até as de redes locais sem-fio (IEEE 802.11). Nessas redes, há dois tipos predominantes de comunicação, a V2I (*Vehicular-to-Infrastructure*) e a V2V (*Vehicular-to-Vehicular*). A primeira envolve nós veiculares e nós de rede estáticos instalados em infraestruturas ao longo das vias, já a segunda envolve somente nós móveis em veículos automotores. A arquitetura V2I pode oferecer maior conectividade de rede e por isso vem sendo utilizada com maior frequência, apesar do maior custo em infraestrutura.

O trabalho desenvolvido visa aumentar a confiabilidade dos transportes públicos fornecendo a estimativa do tempo de chegada dos ônibus aos seus pontos de parada. Entretanto, apesar de considerar a arquitetura V2I, o desafio em calcular as estimativas reside no fato do trânsito e dos passageiros influenciarem no tempo que um ônibus leva em sua trajetória (p.ex., quando congestionamentos se formam devido a acidentes). Além disso, situações como mudanças de trajetória podem acontecer de forma a perturbar as estimativas feitas pelo sistema. Um requisito, portanto, para calcular o tempo de chegada sem erros significativos é o conhecimento das posições dos ônibus de forma mais atual possível. Com esse propósito, muitos trabalhos utilizam sistemas de GPS, devido à sua grande acurácia. Contudo, o uso do GPS exige alguma outra tecnologia de comunicação para divulgação da posição do veículo, sendo os sistemas GPS normalmente apenas receptores. Alternativamente ao GPS, existem técnicas de localização que utilizam tecnologias de redes como as celulares e as sem-fio IEEE 802.11. Uma das vantagens do uso desses sistemas é que eles também podem ser utilizados para comunicações. Esse potencial é explorado mesmo em sistemas simples de localização que funcionam por proximidade. Dessa forma, a localização do veículo é dada pelo nó da rede ao qual ele está conectado. A acurácia desses sistemas de localização é inferior à dos sistemas GPS. Contudo, técnicas para aumentar a acurácia dos sistemas de redes vêm sendo propostas, chegando a obter erro médio quadrático da localização de apenas sete metros [Caceres et al., 2009].

Considerando os sistemas de localização em rede, alguns trabalhos da literatura calculam as estimativas usando métodos baseados em séries históricas coletadas em janelas de tempo que podem ser até da ordem de meses [Manolis e Kwstis, 2004]. Assim, a estimativa de localização usa a média pertencente ao mesmo período de um mesmo dia da semana, desconsiderando situações como acidentes e exigindo um volume de informações passadas muito grande. Em oposição à abordagem com séries históricas, existem também aquelas de tempo real, onde apenas as informações do próprio dia são usadas, modelando melhor situações imprevistas, como acidentes. As abordagens de tempo real assumem que o tempo que um dado veículo leva em um percurso é o mesmo gasto por um veículo anterior [Lin e Zeng, 1999] já que o histórico passado é curto. Existem ainda técnicas que utilizam modelos de regressão, que realizam a previsão através de um conjunto de variáveis independentes escolhidas para modelar o trânsito. O problema dessa premissa é que as variáveis dos sistemas de transporte são correlacionadas, o que limita a aplicabilidade dessa abordagem [Chien et al., 2002]. Por fim, existem ainda alguns métodos que usam filtros de Kalman que não possuem um bom desempenho quando os dados de localização são temporalmente esparsos [Karbassi e Barth, 2003].

Este trabalho propõe um sistema para a estimativa da chegada dos ônibus usando redes IEEE 802.11 chamado WiBus. O objetivo é manter ao mesmo tempo requisitos de baixo custo e baixa complexidade, reunindo vantagens das propostas anteriores. Para isso, o WiBus utiliza a localização por proximidade em relação a roteadores sem-fio IEEE 802.11, caracterizando um sistema baseado em redes veiculares com arquitetura V2I. Essa abordagem, ao contrário das que utilizam GPS, permite o uso de apenas um tipo de dispositivo na fase de localização do veículo e na fase de divulgação da informação, reduzindo a complexidade e o custo do sistema. Como neste trabalho os dados de localização são esparsos devido ao método de localização e como não há uma base de dados com informações de localização antigas, a técnica usada para previsão de tempo de chegada é a de tempo real. O WiBus estima o horário de chegada de ônibus e se adapta a mudanças de trajetória dinamicamente. Assim, mesmo que um ônibus fuja da sua trajetória conhecida, ele consegue se adaptar. O sistema é capaz de atender as demandas de uma cidade grande como a do Rio de Janeiro com erros da ordem de poucos minutos, como os alcançados a partir de experimentos com dados reais. Além disso, para facilitar o acesso do público ao sistema, foram desenvolvidos um aplicativo para smartphones Android, chamado BuZoom, e uma página Web que agem como interfaces para solicitação de estimativas. Com isso, pode-se dizer que o sistema atende as especificações para utilização em ambientes reais.

O presente trabalho está organizado da seguinte forma. A Seção 2 apresenta a arquitetura do WiBus. A Seção 3 fornece os detalhes da implementação do sistema e um exemplo de funcionamento para o algoritmo de criação e manutenção dinâmica de trajetórias. Na Seção 4, o cenário de testes é descrito, enquanto na Seção 5, os experimentos realizados e os resultados obtidos são apresentados. Finalmente, a Seção 6 conclui este trabalho e apresenta as direções futuras.

2. Arquitetura do WiBus

No sistema WiBus, ilustrado na Figura 1, existem quatro entidades: Central, Unidade de Acostamento (UA), Ônibus e Cliente. A Central é um computador que executa o programa principal do sistema. Ela possui todas as informações necessárias para o cálculo das estimativas, os mapas criados dinamicamente, a localização de cada veículo e os dados referentes aos tempos entre pontos consecutivos. Além disso, a Central é responsável por calcular as estimativas e responder as requisições dos Clientes. Já as Unidades de Acostamento (UAs) são os pontos de acesso instalados nos pontos de ônibus. Elas oferecem redes sem-fio com identificadores (*Service Set IDentifier* - SSID) pré-fixados iguais para a conexão dos roteadores nos ônibus. Para que um ônibus seja rastreado, ele deve possuir um roteador sem-fio IEEE 802.11 executando alguns programas desenvolvidos. Esses programas localizam o ônibus em relação às UAs e enviam a informação de localização à Central através das próprias UAs. O Cliente é qualquer pessoa que solicita estimativas ao sistema. A requisição do Cliente é através do aplicativo BuZoom para smartphones Android ou através da página web desenvolvidos.

Serviços do WiBus: O sistema desenvolvido oferece os serviços complementares de localização de veículos e a estimativa do tempo de chegada de um ônibus aos seus pontos de parada. Os dois serviços se baseiam na técnica de localização por proximidade em relação a pontos de acesso posicionados nos pontos de parada. Logo, ao se aproximar de

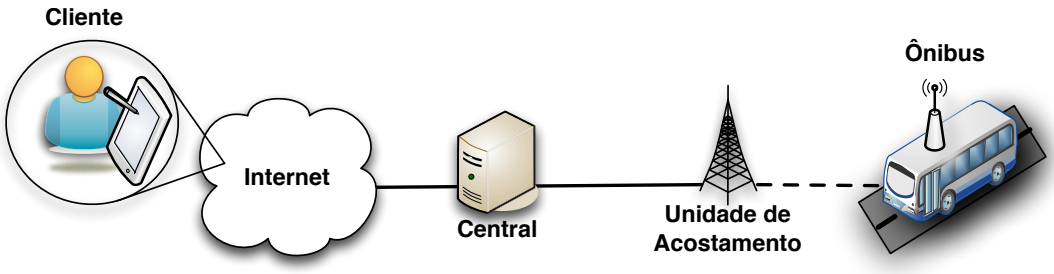


Figura 1. Entidades do sistema na arquitetura.

um ponto de parada, o ônibus se conecta ao ponto de acesso. Em seguida, ocorre uma troca de mensagens, onde o ônibus se identifica à UA e vice-versa. Após conectar-se à UA, o ônibus envia uma mensagem à Central informando a qual UA ele está conectado. Mesmo de posse da localização dos ônibus, ainda é necessário ter um mapa com as trajetórias das linhas de ônibus para oferecer o serviço de estimativa de horário de chegada. Esse mapa deve ser usado em conjunto com o histórico de períodos recentes que os ônibus levam entre dois pontos consecutivos da linha. A abordagem inicial para ter o mapa das linhas de ônibus é supor que as trajetórias não se alteram. Porém, essa suposição é falha já que mudanças temporárias podem ocorrer.

3. Implementação

O WiBus é implementado como um conjunto de seis programas, listados a seguir:

- `WiBusUAServer` – programa executado ininterruptamente na UA que a cada conexão de um ônibus envia uma mensagem com a identificação da própria UA ao ônibus conectado, recebendo, em seguida, uma mensagem com a identificação do ônibus utilizada apenas para manter um histórico de contatos na UA.
- `WiBusClientUA` – programa executado no Ônibus para a recepção das mensagens enviadas pelas UAs com a identificação delas e para envio de mensagens com a identificação do ônibus às UAs. O `WiBusClientUA` é executado toda vez que o Ônibus se conecta a uma UA.
- `WiBusClientCentral` – programa executado no Ônibus para recepção de mensagens enviadas pela Central e para envio de mensagens para a Central contendo informações de localização na rede. Esse programa é executado após o fim do programa `WiBusClientUA`.
- `WiBusCentralServer` – programa executado ininterruptamente na Central para envio de mensagens ao ônibus e recepção de mensagens enviadas pelo ônibus. A partir das mensagens recebidas, cria-se o mapa das linhas de ônibus, localizam-se os ônibus e calculam-se as estimativas de tempo de chegada dos ônibus. Além disso, este programa recebe e responde os pedidos de estimativa dos Clientes.
- `BuZoom` – programa da interface executada nos smartphones Android dos Clientes. Esse programa solicita estimativas de tempo de chegadas dos ônibus à Central.
- `BuZoom Web` – programa da interface web acessada pelos Clientes. Assim como o BuZoom, ele acessa a Central para obtenção das informações sobre os ônibus.

No WiBus, o ônibus descobre sua posição na rede com a execução do programa `WiBusClientUA`. Em seguida, o programa `WiBusClientCentral` é executado, enviando uma mensagem à Central. Essa mensagem contém as seguintes

informações: UA onde o ônibus está conectado, UA anterior na qual o ônibus esteve conectado, o identificador do ônibus e o identificador da linha de ônibus. O programa `WiBusCentralServer`, ao receber essa mensagem, realiza seu tratamento.

3.1. Central

No WiBus, uma linha de ônibus é caracterizada pelo seu número e seu sentido (p.ex. 913 Del Castilho), de maneira que todas as linhas de ônibus não são consideradas circulares para o sistema. Além disso, uma linha de ônibus é composta de vários trechos, onde um trecho é definido como um par de UAs, em que a primeira UA do par vem imediatamente antes da segunda na trajetória da linha de ônibus.

Ao receber uma mensagem, a Central atualiza a posição dos ônibus com as UAs especificadas na mensagem. Em seguida, a Central atualiza um mapa associativo que relaciona trechos das linhas de ônibus aos tempos gastos pelos últimos ônibus nesses trechos. Esse mapa associativo é utilizado na estimativa do tempo necessário para o ônibus mais próximo alcançar cada UA das linhas de ônibus.

Como uma linha de ônibus é caracterizada como uma sequência de trechos, para estimar o tempo necessário para que um ônibus alcance uma determinada UA, basta somar as estimativas de tempo necessárias para o ônibus percorrer cada um dos trechos que liga a UA onde o ônibus se encontra à UA alvo da estimativa. Assim, o problema se reduz a estimar o tempo necessário para que um ônibus percorra um dado trecho. No WiBus, a estimativa para um trecho é calculada como a média dos tempos gastos pelos K últimos ônibus que percorreram o mesmo trecho. Em outras palavras, é uma média móvel simples de tamanho K dos tempos do trecho mantidos no mapa associativo da Central. O uso da média móvel simples provê complexidade reduzida ao cálculo das estimativas.

A Central, utilizando os conceitos acima descritos, estima o tempo necessário para que o ônibus mais próximo alcance cada UA de uma dada linha de ônibus. Isso é realizado calculando para todos os ônibus da linha, as estimativas até cada UA dessa linha, armazenando apenas a menor estimativa para cada UA. Esse processo é descrito formalmente no Algoritmo 1, onde $estimativa_b$ armazena a estimativa para um dado ônibus b alcançar o trecho atual a partir de sua posição inicial, e a função `estimativaTrecho` retorna a estimativa do ônibus b percorrer apenas o trecho atual. Já, o algoritmo de criação e atualização dos mapas das linhas de ônibus é descrito na Seção 3.2.

Algoritmo 1: Cálculo da estimativa para próximo ônibus alcançar UAs de uma linha de ônibus.

Entrada: linha de ônibus l.
Saída: estimativa mínima para cada UA.

```
1 para todo ônibus b servindo a linha l faça
2     enquanto trecho existe faça
3         estimativa_b += estimativaTrecho( trecho);
4         se estimativa_b < estimativa da UA atual então
5             estimativa da UA atual = estimativa_b
6         fim
7         trecho = próximo trecho;
8     fim
9 fim
```

3.2. Algoritmo de Criação e Manutenção Dinâmica de Trajetórias

A necessidade de representar trajetórias dinamicamente emerge de situações onde imprevistos alteram as trajetórias, como quando vias são fechadas. Contudo, peculiaridades do mundo real, como um motorista de ônibus mudando a trajetória inadvertidamente, impedem que mudanças sinalizadas pelos veículos sejam assumidas como verdade instantaneamente. Uma maneira de verificar se a trajetória da linha de ônibus realmente mudou é através da observação de outros ônibus da mesma linha. A mudança é confirmada, caso outros ônibus da mesma linha repitam a mesma trajetória. Assim, o algoritmo de criação e manutenção dinâmica de trajetórias proposto utiliza um esquema de pesos para evitar mudanças prematuras na trajetória das linhas de ônibus.

Esse esquema possui dois tipos de peso: um que é usado nos arcos, que são as estruturas utilizadas para indicar as UAs próximas e anteriores de uma UA na trajetória do ônibus, e outro que é usado na própria UA. O primeiro tipo de peso rege a dinamicidade do mapa, capturando o quão rápido uma modificação na trajetória é considerada como permanente. Esse peso pode ser ajustado desde que respeite a restrição de ser no máximo duas vezes o número de ônibus servindo a linha. Essa restrição é necessária para que, em uma mudança de trajetória, os pesos dos arcos de uma UA em remoção alcancem o valor zero antes do peso da própria UA.

O segundo tipo de peso é usado para remoção de UAs que não pertençam mais a uma dada trajetória. O valor desse tipo de peso deve ser grande o suficiente para que nenhuma UA seja excluída erroneamente. Por isso, o valor inicial atribuído a uma UA é igual ao dobro do produto da quantidade de UAs da linha pela quantidade de ônibus realizando-a. Como o peso máximo depende do número de UAs cadastradas, ao acrescentar uma UA em uma linha, todas as UAs da linha têm seu peso igualado ao novo máximo, por questões de equivalência. Vale ressaltar, portanto, que o sistema trata também situações onde uma nova UA é inserida na trajetória ou uma anterior reaparece. Essa informação é obtida das mensagens recebidas pela Central.

No WiBus, o valor do peso inicial atribuído a uma UA também é o peso máximo permitido às UAs. Isso significa que é necessário que todos os ônibus de uma dada linha percorram todas as UAs da linha ao menos duas vezes, para que uma UA não utilizada seja excluída. Similarmente, o incremento dado quando uma UA é usada deve garantir que essa UA não seja excluída enquanto o ônibus percorre as outras UAs da linha. Para isso, o incremento é igual ao dobro da quantidade de UAs na linha de ônibus.

Durante o transiente de um processo de mudança de uma linha de ônibus, uma UA pode possuir mais de uma possibilidade de próxima UA ou UA anterior. Nesses casos, uma UA possui uma lista de arcos indicando as UAs próximas ou anteriores a ela, ao invés de apenas um arco. Assim, a atualização dos arcos para as UAs anteriores ou próximas é realizada da seguinte forma: subtrai-se uma unidade do peso de todos os arcos da lista, de UAs anteriores ou próximas, e somam-se duas unidades ao peso do arco, anterior ou próximo, usado. Caso o peso de algum arco se torne zero na atualização, ele é removido, indicando que uma UA não é mais usada como anterior ou próxima da atual.

Para concluir o esquema de atualização dos pesos dos arcos, resta explicar como é realizada a atualização do outro tipo de peso. No processo de atualização do peso das UAs de uma linha de ônibus, o peso de todas as UAs da linha é decrementado de uma

unidade. Após a subtração, incrementa-se o peso da UA atual com o valor anteriormente especificado, isto é, duas vezes a quantidade de UAs na linha de ônibus.

Por fim, outra operação necessária é a exclusão das UAs não utilizadas da linha de ônibus. Nesse processo as UAs cadastradas que tem peso igual a zero são eliminadas. Com isso, o pseudocódigo simplificado responsável pela criação e atualização dos mapas das linhas de ônibus pode ser visto em Algoritmo 2.

Algoritmo 2: Criação e atualização do mapa de linha de ônibus.

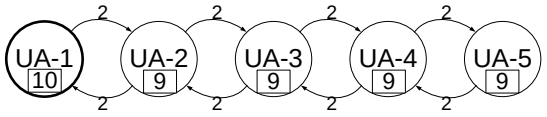
Entrada: mensagem m recebida do ônibus b servindo a linha l.
Saída: mapa da linha de ônibus atualizado.

```
1  se recebeu mensagem anterior então
2      se b mudou de linha então
3          Altera quantidade de ônibus da linha atual l e da linha antiga;
4      fim
5      se b visto pela primeira vez então
6          Soma 1 à quantidade de ônibus da linha atual l;
7      fim
8      busca l descrita em m;
9      se achou l então
10         busca UA_atual de m em l;
11         se achou UA_atual então
12             Atualiza arcos para UAs anteriores da UA_atual;
13             Atualiza peso das UAs de l;
14             Atualiza arcos para próximas UAs da UA_anterior;
15             Exclui UAs não utilizadas de l;
16         senão
17             Cria UA_atual de m;
18             Insere UA_atual em l;
19             Peso das UAs de l = peso_maximo;
20             Cria arco para a UA_anterior na UA_atual;
21         fim
22     senão
23         Cria UA_atual de m;
24         Cria l de m com UA_atual;
25     fim
26 fim
```

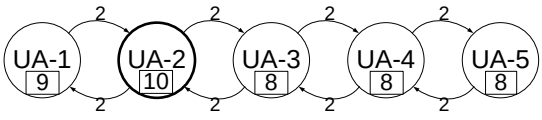
3.3. Exemplo de Funcionamento

O exemplo de funcionamento demonstra como o sistema lida com mudanças na trajetória de uma linha de ônibus. No exemplo, um ônibus percorre a trajetória de uma linha de ônibus fictícia que sofre modificações. Para tal, os mapas da linha de ônibus são representados na Figura 2. Nessas figuras, as circunferências representam as UAs e o número no interior do retângulo em cada UA é o peso da UA usado no critério de exclusão. Já os números nas setas representam os pesos dos arcos associados às informações de próxima UA e UA anterior da UA de origem da seta. Toda vez que um ônibus se move, o algoritmo de atualização dos mapas é executado pela Central.

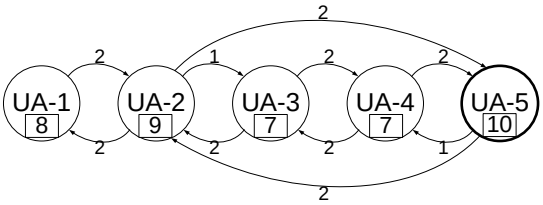
O teste se inicia com o ônibus se movendo até a UA-1. Com isso, o algoritmo de



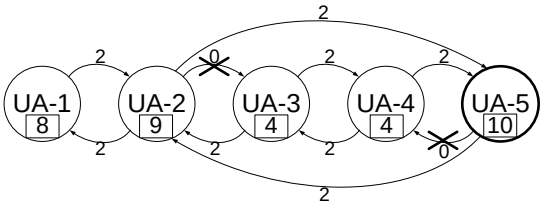
(a) Trajetória inicial da linha de ônibus.



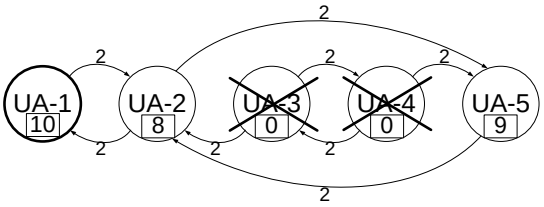
(b) Ônibus se move até a UA-2.



(c) Ônibus alcança a UA-5, sem passar pela UAs 3 e 4.



(d) A mudança na trajetória é concretizada.



(e) Exclusão das UAs 3 e 4.

Figura 2. Exemplo de funcionamento do Algoritmo de Criação e Manutenção Dinâmica de Trajetórias.

atualização de trajetórias atualiza o peso da própria UA para o máximo, e reduz em uma unidade o peso de todas as outras UAs da linha. Essa situação se encontra na Figura 2(a).

À medida que o ônibus se movimenta sem alterações na trajetória, os pesos dos arcos entre as UAs continuam no valor máximo permitido. Entretanto, a cada movimento do ônibus, o peso de todas as UAs da linha é decrementado de uma unidade, enquanto o peso da UA onde o ônibus se encontra é incrementado. Por exemplo, o mapa atualizado para o ônibus na UA-2 pode ser visto na Figura 2(b).

Nesse exemplo de funcionamento, supõe-se que as UAs 3 e 4 estão defeituosas. Com isso, a Central não recebe as mensagens referentes a esses dois pontos de acesso, fazendo com que a próxima UA funcionando na trajetória seja a UA-5. Ao chegar à UA-5, o ônibus envia uma mensagem à Central, informando que ele está lá e que antes esteve na UA-2. Com isso, a UA-5 é cadastrada na lista de próximas UAs da UA-2. Da mesma maneira, a UA-2 é cadastrada na lista de UAs anteriores da UA-5. Ambos os arcos são inicializados com peso máximo e as outras informações da lista de próximas UAs da UA-2 e da lista de UAs anteriores da UA-5 são decrementadas de uma unidade. O resultado desse processo pode ser visto na Figura 2(c).

Continuando o exemplo, o ônibus percorre a trajetória alcançando a UA-1 e depois a UA-2. Quando o ônibus se aproxima da UA-5 novamente, o sistema verifica que, novamente, a UA anterior a UA-5 é a UA-2. Por conseguinte, a UA seguinte a UA-2 é a UA-5. Com isso, o peso desses arcos se mantém, enquanto o peso dos outros arcos decresce de uma unidade. Assim, os arcos antigos atingem o valor zero e por isso são excluídos, o que representa a concretização da modificação da trajetória. Isso pode ser afirmado, pois apesar das UAs 3 e 4 existirem na lista de UAs da linha, elas já não são alcançáveis a

partir do ponto inicial da linha de ônibus. As modificações realizadas podem ser vistas na Figura 2(d).

Com a trajetória do ônibus atualizada, não ocorrem mais alterações nos arcos entre as UAs da trajetória. Todavia, o ônibus continua realizando a trajetória, e os pesos das UAs são atualizados de acordo. Esse processo se repete até os pesos das UAs 3 e 4 chegarem a zero. Isso faz com que o mecanismo de atualização da linha de ônibus exclua essas UAs da lista de UAs da linha, como pode ser visto na Figura 2(e).

Com a exclusão das UAs 3 e 4 da lista de UAs da linha de ônibus, o valor máximo dos pesos das UAs diminui. Porém, o peso de cada UA só é atualizado, obedecendo ao novo limite, quando o ônibus alcançar cada UA. Com esse exemplo, mostra-se que os mapas das linhas de ônibus podem ser alterados dinamicamente. Vale ressaltar que a velocidade de concretização de alterações na trajetória pode ser configurada escolhendo o valor máximo dos pesos dos arcos das UAs.

3.4. Interface Gráfica

A interface gráfica desempenha um papel fundamental no projeto. É através dela que o usuário faz sua requisição de forma amigável. A primeira interface gráfica criada para o sistema foi uma página web, o BuZoom Web. No entanto, tal estrutura é mais indicada quando o acesso à informação é efetuado em computadores. Acreditando-se que a maioria dos usuários deva realizar sua pesquisa pelos seus smartphones, foi desenvolvido o aplicativo BuZoom, que pode ser visto na Figura 3. Essa interface está disponível para dispositivos com sistema operacional Android 2.2 (Froyo) ou versões mais recentes.

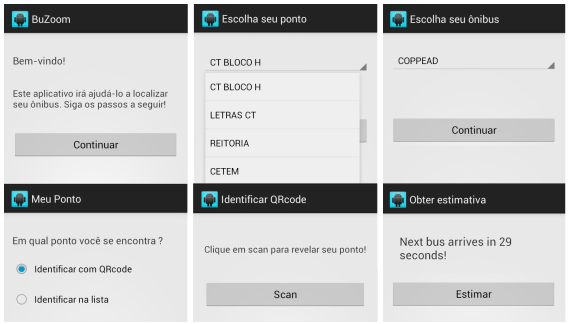


Figura 3. Telas do BuZoom para Android.

A consulta do usuário pode ser feita de duas formas distintas, cada uma mais apropriada para um cenário específico. Na primeira, o usuário usa um leitor de `QRcode`, que está disponível no ponto, para identificar o ponto de ônibus. Na segunda, o usuário escolhe seu ponto em uma lista suspensa. A primeira forma é mais apropriada para usuários móveis que, apesar de já estarem no ponto de ônibus, não sabem onde se localiza tal ponto. Já a segunda forma é mais apropriada para os usuários remotos que ainda não se encontram no ponto ou que desejam ir até o ponto apenas quando o ônibus estiver mais próximo. Após a escolha do ponto de ônibus, é possível escolher a linha desejada através de uma listagem. Para enviar o par (ponto escolhido, ônibus escolhido) para a Central, é preciso que o usuário tenha conexão com a Internet. Após o envio da requisição, a resposta pode ser visualizada na tela do dispositivo.

As ferramentas utilizadas no desenvolvimento do BuZoom foram o Android SDK (*Software Development Kit*) e o JDK (*Java Development Kit*). Já o reconhecimento do QRcode foi feito importando o pacote `Zxing.integration.android` e fazendo uma requisição a um aplicativo auxiliar chamado `BarcodeScanner`. A produção dos `QRcodes`, referentes aos pontos de ônibus do Campus Ilha do Fundão da UFRJ, foi feita com o aplicativo `QRdroid`.

4. Rede de Testes

Para testar o WiBus, foi montado um pequeno protótipo no laboratório, onde os diferentes elementos da arquitetura são emulados. A Central é um PC com sistema operacional Debian instalado, 8 GB de memória RAM e processador Intel Core i7 860. Por outro lado, as UAs e os ônibus são representados por roteadores sem-fio D-Link, modelo DIR-320. A Central e as UAs estão conectadas através de um roteador, também D-Link e modelo DIR-320, que foi configurado para funcionar como comutador. Já a conexão das UAs e da Central com esse comutador é realizada por cabo Ethernet. O modelo DIR-320 possui processador ARM de 240 MHz e 32 MB de memória RAM. Além disso, possui uma entrada USB que pode ser usada para aumentar a capacidade de armazenamento do roteador, que originalmente é de apenas 4 MB, ou para instalar uma segunda interface de rede sem-fio. O sistema operacional usado nos roteadores é o OpenWRT versão Backfire 10.03.1, uma distribuição Linux para dispositivos embarcados.

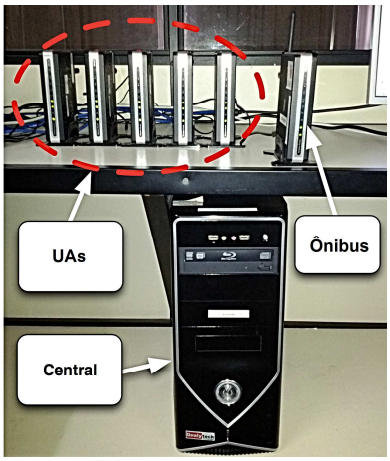


Figura 4. Ambiente de testes.

5. Avaliação

Nos experimentos deste trabalho o peso usado para os arcos das UAs é igual ao máximo possível definido (Seção 3.2). O primeiro experimento tem o objetivo de avaliar a capacidade de atendimento de ônibus do WiBus, ou seja, avaliar a quantidade máxima de ônibus sob a qual o sistema ainda é capaz de rastrear os veículos e calcular as estimativas. Já o segundo experimento, que é realizado com dados reais de um cenário universitário, tem o objetivo de avaliar os erros das estimativas calculadas pelo WiBus.

5.1. Capacidade de Atendimento de Ônibus

Este teste simula as condições da cidade do Rio de Janeiro e tem por objetivo avaliar a capacidade de atendimento de ônibus do WiBus. No teste, são cadastrados 8.000

ônibus distribuídos uniformemente em 800 linhas fictícias de ônibus, de forma a reproduzir um cenário similar ao do Rio de Janeiro [de Urbanismo Pereira Passos, 2011]. Esses cadastros são feitos através de um *script* que emula a passagem do ônibus pelas UAs, enviando mensagens do roteador que faz papel de ônibus à Central. Como o número de UAs inseridas em uma linha de ônibus pode variar, por exemplo, devido às diferentes quantidades de pontos de ônibus nas linhas, o número de UAs das linhas de ônibus é variado no experimento de 10 a 100 com incrementos de 10. Após a criação das trajetórias das linhas fictícias, um dos ônibus cadastrados percorre uma das trajetórias 10 vezes, enviando mensagens para a Central ao trocar de UA. A Central trata as mensagens e mede o tempo gasto no tratamento. Todo o processo é repetido três vezes para cada número de UAs diferente nas linhas de ônibus. Os resultados do teste podem ser vistos na Figura 5.

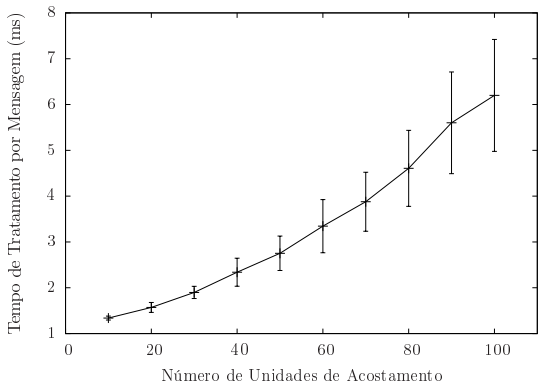


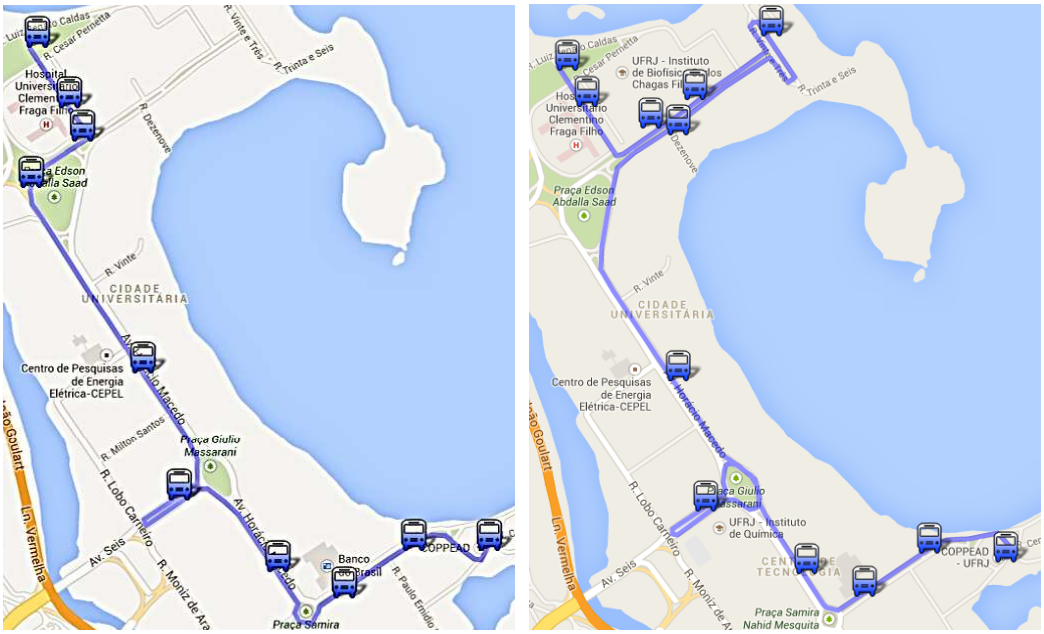
Figura 5. Tempo de tratamento de mensagens.

A partir do gráfico, infere-se que o sistema WiBus é capaz de monitorar e estimar toda a cidade do Rio de Janeiro com apenas um computador similar ao usado. Essa inferência é realizada observando que o tempo de tratamento por mensagem é inferior a 7 ms e supondo que os ônibus da cidade levam em média mais de um minuto para alcançar à próxima UA da trajetória. Com isso, o sistema tem pelo menos 60 segundos para tratar as 8.000 mensagens. Dividindo-se os 60 segundos pelo tempo de tratamento do pior caso, 7 ms, obtém-se uma taxa de aproximadamente 8.500 mensagens tratadas por minuto. Portanto, todas as mensagens enviadas à Central são tratadas, mesmo considerando o pior caso para todas as linhas de ônibus da cidade. Como o sistema é capaz de atender grandes demandas, no experimento seguinte é avaliada a qualidade das estimativas realizadas.

5.2. Experimentos em um Cenário Universitário

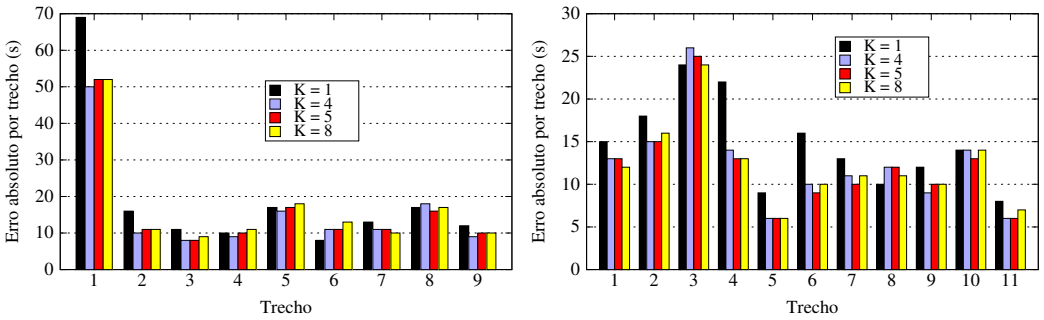
Para este teste, mediu-se nove vezes o tempo gasto em todos os trechos das linhas internas de ônibus universitários "COPPEAD" e "Estação UFRJ", mostradas na Figura 6. Esses dados foram usados para avaliar a qualidade das estimativas do sistema WiBus. As estimativas das linhas de ônibus são calculadas através de médias móveis simples de tamanho K. Portanto, também é necessário definir um valor para o parâmetro K. Para ambos os fins, as estimativas e os erros foram calculados para os dados reais medidos com K variando de 1 a 8. Por questões de clareza, são apresentados os erros absolutos médios por trecho dos casos onde K vale 1, 4, 5 e 8 para ambas as linhas de ônibus na Figura 7.

O valor de K é escolhido de forma a minimizar o erro médio da estimativa do ônibus desde o ponto inicial até o ponto final da linha. Os valores de K são os que



(a) Linha “COPPEAD”. (b) Linha “Estação UFRJ”.

Figura 6. Trajetórias das linhas de ônibus universitárias.

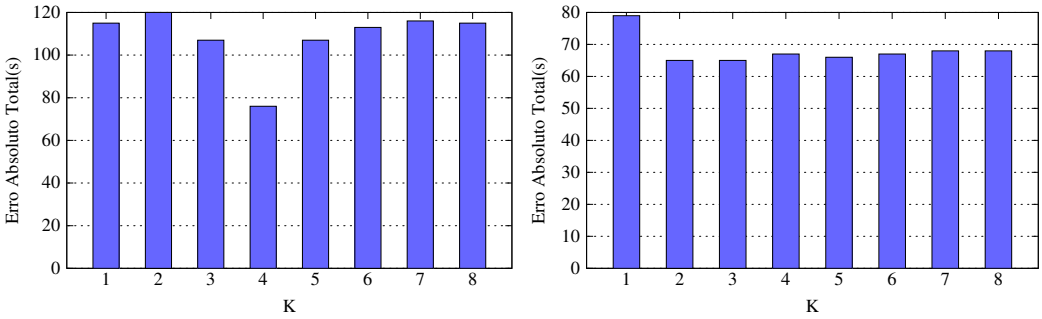


(a) Linha “COPPEAD”. (b) Linha “Estação UFRJ”.

Figura 7. Erro absoluto médio para diferentes valores de K.

produzem o menor erro absoluto segundo os gráficos da Figura 8, $K = 4$ para a linha “COPPEAD” e $K = 3$ para a linha “Estação UFRJ”.

A escolha de tamanhos diferentes para as médias móveis é justificada pelo fato de linhas de ônibus diferentes apresentarem comportamentos diferentes. As características dos trechos das linhas de ônibus estudadas são resumidas nas Tabelas 1 e 2. Nessas tabelas, observa-se que alguns trechos apresentam elevado coeficiente de variação, definido como a divisão do desvio padrão pela média. O elevado coeficiente de variação desses trechos é justificado pela presença de semáforos e travessias de pedestres. Além disso, a partir das informações da tabela, pode-se calcular o coeficiente de variação médio das duas linhas. A linha “COPPEAD” possui coeficiente de variação médio igual a 0,21, enquanto a linha “Estação UFRJ” apresenta o mesmo coeficiente igual a 0,18. Esses valores ajudam a compreender os gráficos da Figura 8, já que linhas com maiores variações tendem a apresentar um maior compromisso entre informações recentes e informações de longo



(a) Linha “COPPEAD”. (b) Linha “Estação UFRJ”.

Figura 8. Erro médio de estimativa do ponto inicial até o final para diferentes valores de K.

prazo, como é o caso da linha “COPPEAD”, enquanto linhas com menores variações não apresentam esse comportamento de forma tão nítida, como a linha “Estação UFRJ”.

Trecho	Média (s)	Desvio Padrão	Coeficiente de Variação
1	116,44	52,04	0,45
2	42,67	12,43	0,29
3	44,33	9,90	0,22
4	107,56	11,75	0,11
5	79,67	17,26	0,22
6	140	14,73	0,11
7	85,33	10,83	0,13
8	63,67	16,59	0,26
9	61,22	9,00	0,15

Tabela 1. Linha “COPPEAD”.

Trecho	Média (s)	Desvio Padrão	Coeficiente de Variação
1	58,56	13,62	0,23
2	42,11	15,31	0,36
3	85,44	24,69	0,29
4	122,11	15,14	0,12
5	173,89	5,92	0,03
6	128	9,75	0,08
7	127,56	10,88	0,09
8	80,44	11,64	0,14
9	27,33	8,94	0,33
10	97,11	13,56	0,14
11	61,56	7,63	0,12

Tabela 2. Linha “Estação UFRJ”.

Finalmente, a partir da Figura 8 verifica-se que mesmo para o maior trajeto das linhas, do ponto inicial até o ponto final, o erro médio absoluto das estimativas se mantém abaixo de 80 segundos. Com isso, conclui-se que as estimativas fornecidas pelo WiBus têm grande potencial para os usuários dos transportes públicos, já que o erro da estimativa do sistema representa o tempo que o usuário deverá esperar em média no ponto de ônibus até que um ônibus da linha desejada chegue. Nesse contexto, o erro obtido pode ser considerado desprezível frente ao tempo que um usuário sem informação aguardaria.

6. Conclusões e Trabalhos Futuros

Este trabalho propôs um sistema para estimativa dos horários de chegada dos ônibus nos seus respectivos pontos. O WiBus utiliza informações obtidas a partir de uma rede sem-fio IEEE 802.11, cujos dispositivos estão instalados nos ônibus e na infraestrutura ao longo das vias, para manter atualizadas as informações de localização dos ônibus. Com o intuito de evitar falhas do sistema no caso de mudanças de trajetória, um algoritmo de criação e manutenção dinâmica de mapas digitais foi proposto e avaliado. O serviço

de estimativa de chegada de ônibus nos respectivos pontos foi baseado em médias móveis simples de tamanho K, tendo K sido calculado para duas linhas de ônibus universitárias com base em dados reais. Os resultados obtidos mostram que as estimativas realizadas pelo WiBus apresentam erro da ordem de minuto, mesmo do ponto inicial até o ponto final das linhas de ônibus avaliadas. Os resultados ainda mostram que o WiBus é capaz de atender as exigências de uma cidade grande como o Rio de Janeiro, sendo capaz de tratar mais de 8.500 mensagens por minuto no pior caso. O sistema WiBus possui adicionalmente duas possibilidades de interface com o usuário: uma via página web e a outra via aplicativo para Android. Em ambos os casos, a interface é chamada de BuZoom.

Como trabalhos futuros, pretende-se estender as medidas com mais dados práticos dos tempos gastos pelos ônibus para percorrer trechos de suas linhas. De posse de mais dados, seria possível avaliar mais precisamente a escolha do parâmetro K, sendo possível inclusive a criação de um algoritmo dinâmico para ajuste de K em função do erro.

Referências

Caceres, M., Sottile, F. e Spirito, M. (2009). WLAN-based real time vehicle locating system. Em *Vehicular Technology Conference. VTC Spring 2009. IEEE 69th*, p. 1–5.

Casey, R. F., Labell, L. N., Prensky, S. P. e Schweiger, C. L. (1991). Advanced public transportation systems: the state of the art. Relatório técnico, U.S. Department of Transportation.

Chien, S., Ding, Y. e Wei, C. (2002). Dynamic bus arrival time prediction with artificial neural networks. *Journal of Transportation Engineering*, 128(5):429–438.

de Urbanismo Pereira Passos, I. M. (2011). Total de linhas, frota operante, passageiros transportados, viagens realizadas, quilometragem coberta, combustível utilizado e pessoal ocupado pelo sistema de Ônibus - município do Rio de Janeiro - 1994 - 2011 (tabela nº 1736). Armazém de Dados - Informações sobre a cidade do Rio - `http://www.armazemdedados.rio.rj.gov.br/`.

DENATRAN (2013). Frota - DENATRAN - Departamento Nacional de Trânsito. `www.denatran.gov.br/frota.htm`.

Karbassi, A. e Barth, M. (2003). Vehicle route prediction and time of arrival estimation techniques for improved transportation system management. Em *Intelligent Vehicles Symposium, 2003. Proceedings. IEEE*, p. 511–516.

Lin, W.-H. e Zeng, J. (1999). Experimental study of real-time bus arrival time prediction with GPS data. *Transportation Research Record: Journal of the Transportation Research Board*, 1666(1):101–109.

Manolis, K. e Kwstis, D. (2004). Intelligent transportation systems - travelers' information systems the case of a medium size city. Em *Mechatronics. ICM '04. Proceedings of the IEEE International Conference on*, p. 200–204.