Um Algoritmo de Posicionamento de Pontos de Coleta para uma Rede de Sensores Baseada em Ônibus Urbanos

Pedro Cruz[1], **Rodrigo S. Couto**[2] **e Luís Henrique M. K. Costa**[1] *

[1]Universidade Federal do Rio de Janeiro - PEE/COPPE/GTA

[2]Universidade do Estado do Rio de Janeiro - PEL/DETEL/FEN

{cruz,luish}@gta.ufrj.br, rodrigo.couto@uerj.br

Abstract. *Smart Cities can use data obtained from different sensors in order to offer better services. Such data is usually obtained by sensor networks. However, covering the whole city area with static sensors might be too expensive. This way, an alternative is to explore the strategy of embedding sensors into vehicles, enabling the mobility of the sensors and reducing the number of sensors needed. This paper analyses the performance of a network formed by buses from public transport system as a mobility platform for sensors capable of generating alert messages. This paper studies the trade-off between latency of the generated messages and number of collection points.*

Resumo. *Cidades Inteligentes podem utilizar dados obtidos por uma rede de sensores para oferecer melhores serviços. Entretanto, uma rede de sensores estáticos cobrindo toda a área de uma cidade pode apresentar um custo proibitivo. Assim, uma alternativa é explorar a estratégia de instalar sensores em veículos, que oferecem mobilidade ao sensoriamento, reduzindo o número de sensores necessários. Este trabalho analisa o desempenho de uma rede formada pelos ônibus de transporte público da cidade, como plataforma de mobilidade para sensores. O trabalho também propõe um algoritmo para a escolha de pontos de coleta e o avalia utilizando informações reais dos ônibus no Rio de Janeiro, analisando o compromisso entre a latência na entrega dos dados dos sensores e o número de pontos de coleta utilizados.*

1. Introdução

As grandes cidades acumulam problemas gerados pela alta densidade populacional. A oferta de serviços básicos à população, como transporte e segurança, é frequentemente um desafio. O conceito de Cidades Inteligentes é uma das formas de enfrentar essas dificuldades [Chourabi et al., 2012]. Dados sobre a cidade e seus serviços são coletados. A partir dessas informações, gera-se conhecimento e torna-se possível tomar decisões mais eficientes no gerenciamento da infraestrutura das cidades.

Nesse contexto, as redes de sensores servem para coletar e entregar dados que processados embasam decisões inteligentes sobre uma cidade. Tradicionalmente, os nós de uma rede de sensores sem fio coletam dados e os entregam a alguns nós especiais, denominados sorvedouros ou pontos de coleta. Os pontos de coleta são capazes de se

*Este trabalho foi realizado com recursos da FAPERJ, CNPq e CAPES.

conectar a uma rede externa, tipicamente a Internet. Através da Internet, os dados podem alcançar o nó gerenciador de operação, que é o destino final dos dados do ponto de vista da rede de sensores e de outro lado fornece serviços aos usuários finais a partir dos dados coletados [Akyildiz et al., 2002]. De acordo com a arquitetura de Internet das Coisas para Cidades Inteligentes proposta em [Zanella et al., 2014], o gerenciador de operação deve ser uma plataforma de computação em nuvem, capaz de fornecer elasticidade e resiliência aos serviços oferecidos.

A granularidade espacial de dados coletados com sensores estáticos é proporcional à quantidade de sensores envolvidos na coleta [Liu et al., 2005]. Para coletar dados por toda a área de uma Cidade Inteligente, podem ser necessários muitos sensores. Além disso, a rede utilizada para coletar os dados provenientes dos sensores também deve abranger toda a cidade. Por conta desses fatores, a solução pode não ser escalável, pois o custo de instalar e manter uma rede de sensores espalhada por toda uma cidade pode superar os benefícios proporcionados por tal projeto [Hancke et al., 2012]. Para aumentar a cobertura de sensores, [Liu et al., 2005] propõem mover os sensores pela área que se deseja sensoriar. Além de aumentar a área coberta, a movimentação dos nós também explora a entrega oportunística dos dados, como mostrado em [Ekici et al., 2006]. Dessa forma, não há a necessidade de uma rede cobrindo toda a cidade.

No entanto, a utilização de sensores móveis traz desafios, como estimar o atraso na entrega das mensagens. Como os sensores não estão em contato com pontos de coleta por todo o tempo, um sensor que possua dados a serem entregues deve aguardar até o próximo contato com algum ponto de coleta. Esse atraso pode restringir a utilização dos dados por parte de algumas aplicações. Tome-se como exemplo a emissão de um alerta de enchentes para a população. Nessa aplicação, sensores de chuva emitem alertas caso os índices pluviométricos instantâneos excedam um determinado limiar. Caso os dados sobre chuvas intensas demorem horas para alcançar a aplicação, os alertas serão inúteis para que se evite os transtornos causados por enchentes. Por isso, é importante que o atraso entre a coleta de um dado e sua entrega a um ponto de coleta seja estimado. Assim, é possível definir as aplicações que podem utilizar os dados coletados por essa rede.

Apesar da vasta literatura em sensoriamento móvel e redes oportunísticas, o estado da arte foca principalmente na mobilidade dos pontos de coleta, sem analisar os efeitos do número e do posicionamento dos pontos de coleta no atraso da entrega dos dados [Zhao et al., 2016]. Este trabalho visa a utilização dos ônibus do transporte público de uma cidade como plataforma de mobilidade para os nós de sensoriamento, com pontos de coleta posicionados nos pontos de ônibus. O trabalho analisa a utilização de sensores embarcados nos ônibus do sistema de transporte público para entregar dados em uma aplicação de emissão de alertas. Para tal, propõe-se um algoritmo de escolha de pontos de coleta. Assim, este trabalho apresenta três contribuições principais:

- a caracterização do atraso da entrega dos dados em uma rede de sensores embarcados nos ônibus de uma cidade, utilizando dados reais de tráfego de ônibus no Rio de Janeiro;
- a formulação de um problema de programação linear inteira (*Integer Linear Programming* - ILP) para escolha dos pontos de coleta, visando minimizar o atraso de entrega dos dados coletados;
- a proposta de um algoritmo guloso para a escolha dos pontos de coleta, que possui

resultados próximos à solução ótima do ILP proposto.

Este trabalho está estruturado da seguinte forma. A Seção 2 explica o cenário de sensoriamento explorado. A Seção 3 descreve os trabalhos relacionados, enquanto a Seção 4 apresenta o modelo de rede de sensores utilizado. A Seção 5 apresenta a formulação de uma solução ótima para o problema abordado. A Seção 6 propõe um algoritmo para obter uma solução sub-ótima em tempo viável e compara seus resultados com a solução ótima. A Seção 7 aplica o algoritmo proposto na rede formada pelos ônibus e pontos de parada da cidade do Rio de Janeiro, utilizando dados reais de seus posicionamentos. Por fim, a Seção 8 conclui o trabalho e apresenta direções futuras.

2. Sensoriamento Urbano Utilizando Ônibus de Transporte Público

Este trabalho aborda a utilização dos ônibus do transporte público para fornecer mobilidade aos sensores de uma cidade, com baixo atraso na entrega dos dados. A utilização dos ônibus como plataforma de mobilidade tem como vantagem o fato dos ônibus possuírem trajetórias previsíveis, serem parte da infraestrutura pública e não ser necessário gasto de energia com a mobilidade.

No cenário estudado, cada ônibus carrega um nó de sensoriamento, capaz de realizar medidas, armazená-las e transmiti-las através de uma interface sem fio para um ponto de coleta. Os pontos de coleta, localizados em pontos de ônibus, enviam pela Internet os dados recebidos para o nó gerenciador de operação. O gerenciador de operação, executado como um serviço de computação em nuvem, provê serviços para os usuários finais. O esquema está ilustrado na Figura 1.

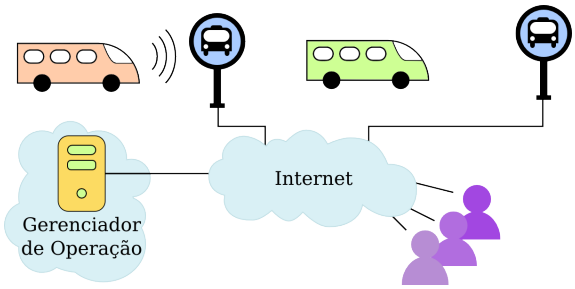


Figura 1. O esquema de distribuição através de ônibus.

A arquitetura de cada nó móvel de sensoriamento, proposta anteriormente em [Cruz et al., 2016], está ilustrada na Figura 2. Cada nó possui um Controlador, que gerencia todas as funções de coletar, armazenar temporariamente e transmitir os dados. O Receptor do Sistema Global de Navegação via Satélite (*Global Navigation Satellite System* - GNSS) calcula o posicionamento do nó e o Banco de Sensores coleta dados ambientais. A interface sem fio possibilita a conexão do nó com os pontos de coleta, para que os dados sensoriados possam ser transmitidos. A unidade Memória Persistente permite que os dados coletados sejam armazenados. Dessa forma, caso não haja uma conexão com um ponto de coleta, os dados coletados podem ser armazenados para envio posterior. Por último, a Fonte de Energia fornece alimentação ao nó, podendo ser alimentada pelo próprio ônibus. A uma determinada taxa de amostragem, o Controlador consulta os dados coletados pelo Banco de Sensores e os dados de posicionamento fornecidos pelo

Receptor GNSS. O Controlador armazena na unidade de Memória Persistente esses dados juntamente com o momento da coleta.

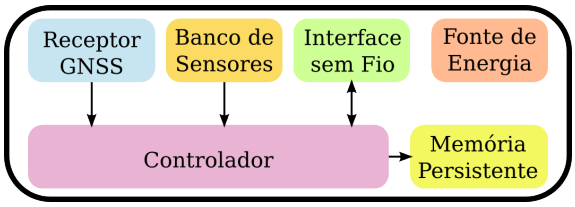


Figura 2. A arquitetura de um nó de sensoriamento embarcado em um ônibus.

Assume-se que os pontos de coleta são a única forma de contato de um nó com a nuvem. Em um trajeto, o intervalo de tempo d que o nó percorre entre dois pontos de coleta consiste no tempo durante o qual o nó não pode enviar dados à nuvem. Assim, é importante que esse intervalo d seja compatível com as aplicações que utilizam os dados sensoriados. Por exemplo, uma aplicação que coleta o histórico de medidas de temperatura da cidade, para posterior análise, pode tolerar atrasos da ordem de horas ou até dias. Por outro lado, as aplicações de alerta descritas anteriormente, como notificações de enchentes, podem tolerar atrasos de apenas alguns minutos.

A rede estudada é uma rede de sensores móveis sem fio. Essas redes possuem topologia dinâmica [Romer e Mattern, 2004], de forma que dois nós da rede podem estar conectados em alguns momentos e desconectados em outros. Por isso, nem sempre existe um caminho entre um sensor qualquer da rede e um ponto de coleta.

Neste trabalho, considera-se que não existe encaminhamento de dados *entre* nós de sensoriamento. O nó que coleta um determinado conjunto de dados é responsável por entregar esse conjunto de dados a um ponto de coleta. Assim, o atraso de envio após um dado ter sido sensoriado por um nó é igual ao tempo desde o sensoriamento do dado até o encontro do nó com um ponto de coleta, mais o tempo de transmissão do dado do nó até o ponto de coleta, mais o tempo de transmissão do ponto de coleta até a nuvem.

Tipicamente, os tempos de encaminhamento do nó até o ponto de coleta e de transmissão do ponto de coleta até o gerenciador de operação são da ordem de milissegundos. Além disso, o tempo que um ônibus demora desde o último encontro com um ponto de ônibus até um próximo encontro com um ponto de ônibus é da ordem de minutos. Por causa dessa discrepância na ordem de grandeza dos valores, este trabalho considera desprezível o atraso de encaminhamento do nó ao ponto de coleta e do ponto de coleta até a nuvem. Apenas o atraso gerado pela espera por conexão é considerado.

3. Trabalhos Relacionados

A literatura sobre posicionamento de pontos de coleta em redes de sensores sem fio estáticos é bem vasta e abrange a otimização de diversas métricas de qualidade de serviço. Wong *et al.* propõem um algoritmo que busca a menor latência para uma rede de sensores estáticos, a partir da configuração espacial de seus pontos de coleta [Wong et al., 2004]. O trabalho propõe um algoritmo computacionalmente custoso capaz de obter a solução ótima e um algoritmo menos custoso, capaz de obter uma aproximação para a configuração de seus pontos de coleta. Uma vez que o tempo de

armazenamento e encaminhamento de um conjunto de dados é muito maior do que o tempo de propagação dos dados em um enlace, a métrica utilizada por Wong *et al.* para minimizar o atraso nas mensagens é o número de saltos. Este trabalho diferencia-se de [Wong et al., 2004] pois são utilizadas redes dinâmicas, formada por ônibus, objetivando reduzir os custos de implantação, mas mantendo a área coberta pelos sensores.

Existem outros trabalhos que estudam o sensoriamento de Cidades Inteligentes através de sensores embarcados em veículos urbanos. O projeto Opensense utiliza sensores embarcados em veículos do transporte público para monitorar partículas de poluição dispersas no ar [Marjovi et al., 2015]. Esse projeto utiliza a rede GSM para a entrega de dados, sem explorar a comunicação oportunística possibilitada pela mobilidade dos nós que compõem a rede. O projeto Mosaic segue uma abordagem semelhante, utilizando sensores embarcados em veículos para medir partículas de poluição [Dong et al., 2015]. Para tal, utiliza WiFi, GPRS ou Bluetooth como tecnologias para a entrega de mensagens aos pontos de acesso. O foco desses trabalhos é o tratamento de dados para sensores de poluição na presença de mobilidade, sem a análise da entrega das medidas obtidas pelos sensores, foco do presente trabalho.

O projeto BusNet [Zoysa et al., 2007] utiliza os ônibus do transporte público para obter dados sobre poluição e também para monitorar as condições da pavimentação das ruas. Porém, a abordagem não procura estabelecer a latência no tempo de entrega das medidas aos usuários e às aplicações. Dessa forma, o trabalho é limitado a aplicações tolerantes a atrasos indefinidos na entrega dos dados. Em contraste, este trabalho quantifica o atraso que os dados devem sofrer desde sua coleta até sua entrega.

[Dias e Costa, 2016] analisam a vazão de uma rede na qual os ônibus de transporte público podem se comunicar e mostram que existe potencial de transferência de dados suficientes para diversos tipos de aplicações, inclusive sensoriamento. O presente trabalho procura caracterizar o atraso na entrega dos dados, a fim de definir melhor as aplicações em sensoriamento que podem utilizar os ônibus enquanto plataforma de mobilidade.

4. Modelo de Rede de Sensores com Pontos de Coleta Limitados

Neste trabalho, assume-se que, dado o custo de implantação, o número de pontos de coleta na rede de sensores móveis formada pelos ônibus deve ser limitado. Assim, o problema é decidir quais pontos de ônibus devem ser escolhidos como pontos de coleta da rede analisada. Esta seção descreve o modelo de redes proposto. A Tabela 1 fornece as notações utilizadas.

Seja $b \in \mathcal{B}$ um ônibus que coleta dados, $p \in \mathcal{P}$ um ponto (parada) de ônibus candidato a ponto de coleta, P_b uma lista ordenada na qual cada elemento de $P_b(i)$ é o i-ésimo ($1 \leq i \leq m$) ponto p com o qual o ônibus b teve contato, e $T_b(i)$ o instante no qual o ônibus b teve contato com o i-ésimo ponto de P_b. É possível interpretar a sequência P_b como o trajeto de b entre os pontos com os quais b tem contato.

Quando um ônibus $b \in \mathcal{B}$ coleta dados ao longo de P_b, esses dados sofrem atrasos de, no máximo, $(T_b(2) - T_b(1), T_b(3) - T_b(2), ..., T_b(m) - T_b(m-1))$. O atraso $T_b(2) - T_b(1)$ é experimentado apenas pelos dados coletados imediatamente após b perder o contato com $P_b(1)$. Todos os outros dados coletados pelo ônibus no caminho entre $P_b(1)$ e $P_b(2)$ sofrem atrasos menores, até que b encontre $P_b(2)$. O mesmo raciocínio é

Tabela 1. Notações utilizadas no trabalho.

Notação	Descrição	Tipo
$\mathcal{B}$	Ônibus que circulam pela cidade	Conjunto
$\mathcal{P}$	Candidatos a ponto de coleta da cidade	Conjunto
$\mathcal{P}_{bp}^{s}$	Candidatos a ponto de coleta que fazem contato com o ônibus b anteriormente ao ponto p	Conjunto
$\mathcal{P}_{bp}^{d}$	Candidatos a ponto de coleta que fazem contato com o ônibus b posteriormente ao ponto p	Conjunto
$\mathcal{I}$	Pontos de ônibus que são os primeiros ou os últimos pontos de qualquer um dos ônibus	Conjunto
P_b	A sequência de pontos de ônibus que fazem contato com o ônibus b, por ordem de contato	Parâmetro
$P_b(i)$	Função que retorna o i-ésimo elemento da sequência P_b	Parâmetro
$T_b(i)$	Instante no qual o ônibus b encontra o i-ésimo elemento da sequência P_b	Parâmetro
D_b	A sequência de intervalos entre contatos do ônibus b com pontos de coleta para o ônibus b	Parâmetro
$D_b(i)$	O i-ésimo elemento de D_b e o intervalo de tempo entre o contato com $P_b(i)$ e o contato com $P_b(i+1)$	Parâmetro
d_{bpq}	Atraso total, para um ônibus b, entre os pontos p e q	Parâmetro
$N_{desejado}$	Quantidade desejada de pontos de coleta	Parâmetro
$A_{máx}$	Maior atraso possível na rede entre dois pontos de um ônibus	Variável
x_i	Valor binário que indica se o ponto i é escolhido como ponto de coleta	Variável
y_{bpq}	Valor binário indicando, para um ônibus b, se q é o próximo ponto de coleta a partir do ponto p	Variável

válido para qualquer par $(P_b(i), P_b(i+1))$ do trajeto P_b. Por motivos de simplicidade, neste trabalho $(T_b(i+1) - T_b(i))$ é denominado o atraso entre $P_b(i)$ e $P_b(i+1)$.

É possível, então, definir D_b como a sequência de atrasos para cada ônibus b e obter os valores de cada elemento em D_b a partir da subtração:

$$D_b = \{T_b(2) - T_b(1), T_b(3) - T_b(2), ..., T_b(m) - T_b(m-1)\}. \quad (1)$$

Pode-se, então, definir o atraso máximo da rede como sendo o maior atraso nas sequências de atrasos de todos os ônibus:

$$A_{máx} = \max_{b \in \mathcal{B}}(\max(D_b)). \quad (2)$$

4.1. Remoção de um candidato a ponto de coleta

No problema estudado, todos os pontos de ônibus $p \in \mathcal{P}$ são candidatos a pontos de coleta. Se o ponto de ônibus $p \in \mathcal{P}$ não for escolhido para ser ponto de coleta, diz-se que o ponto p foi removido. Quando um ponto p é removido, os nós que poderiam descarregar dados neste ponto de coleta agora devem descarregar no próximo ponto de seu trajeto que seja um ponto de coleta. Os nós que podem descarregar em p são todos os nós b que possuem p em uma ou mais posições de sua sequência P_b.

A remoção de p possui efeitos na sequência D_b, ilustrados na Figura 3. Se o ponto p é o elemento $P_b(i)$ e p é removido, então o os dados coletados a partir de $P_b(i-1)$ somente são entregues quando o nó tem contato com o ponto de coleta subsequente, $P_b(i+1)$. Assim, o atraso do $(i-1)$-ésimo elemento em P_b, denotado por $D_b(i-1)$, é acrescido de $D_b(i)$. Além disso, $D_b(i)$ é removido. Em virtude da remoção de $P_b(i)$ e $D_b(i)$ das sequências P_b e D_b, todos os pontos posteriores a $P_b(i)$ e $D_b(i)$ são deslocados uma posição para trás.

A remoção de um candidato a ponto de coleta não possui os mesmos efeitos para ônibus diferentes, pois os ônibus podem seguir trajetos diferentes. A Figura 4 ilustra um exemplo do efeito da remoção do candidato q para dois ônibus, $b, c \in \mathcal{B}$ que se locomovem pelos pontos $p, q, r, s, t \in \mathcal{P}$. A sequência de pontos de b é $P_b = (p, q, r)$ e a sequência de

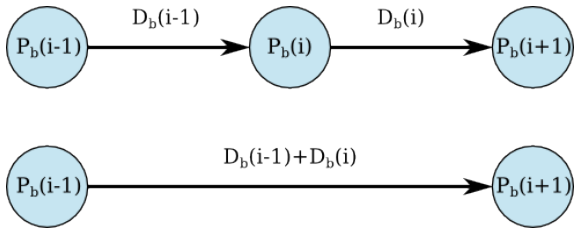


Figura 3. O efeito no tempo entre contatos causado pela remoção de um candidato a ponto de coleta.

atrasos é $D_b = (1, 2)$. A sequência de pontos de c é $P_c = (s, q, t)$ e a sequência de atrasos é $D_c = (3, 1)$. Quando ocorre a remoção do candidato a ponto de coleta q, P_b se torna (p, r), D_b se torna (3), P_c se torna (s, t) e D_c se torna (4).

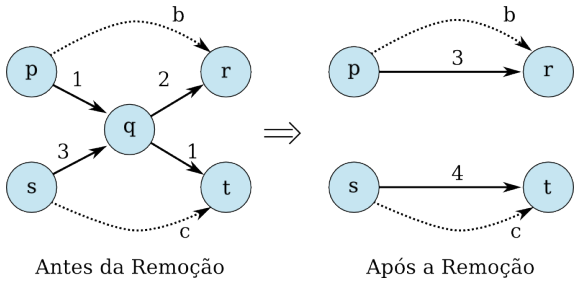


Figura 4. Remoção de um ponto de coleta do ponto de vista de ônibus diferentes.

Define-se, então, a operação de remoção de um candidato a ponto de coleta $p \in \mathcal{P}$:

1. Exclusão de p do conjunto $\mathcal{P}$
2. Para cada $b \in \mathcal{B}$:
   (a) Se houver algum $P_b(i) = p$, para qualquer i:
      i. Remoção $P_b(i)$ de P_b
      ii. Atribuição de $D_b(i-1) := D_b(i-1) + D_b(i)$
      iii. Remoção $D_b(i)$ de D_b

A remoção do ponto p altera um ou mais atrasos em D_b, quando p é parte de P_b. O *atraso de remoção de* p ($A_{rem}(p)$) é definido como o maior atraso que pode ser causado pela remoção de p:

$$A_{rem}(p) = \max_{b \in \mathcal{B}}(\max_{i \in \{2,\ldots,(|P_b|-1)\}} (\{D_b(i-1) + D_b(i) | P_b(i) = p\})). \quad (3)$$

De acordo com o modelo adotado, os pontos que estão no início ou no final de um trajeto P_b não podem ser removidos. O ponto inicial não pode ser removido pois a operação de remoção de um ponto de coleta não é definida para pontos que não possuam um ponto anterior. O último ponto de P_b não pode ser removido porque o último atraso em D_b é o intervalo de tempo entre o penúltimo e o último pontos de P_b. A remoção do último ponto de P_b causaria a indefinição desse atraso. Além disso, o último ponto de P_b não pode ser removido para garantir a entrega de mensagens, já que dados de sensores podem ser gerados até o final do trajeto. O conjunto dos pontos iniciais e finais de todas as sequências P_b para todos os ônibus em $\mathcal{B}$ é denotado conjunto $\mathcal{I}$.

Escolhe-se como métrica objetivo o atraso máximo da rede, pois esse será maior atraso experimentado por uma mensagem de alerta emitida por um nó embarcado em um ônibus. Existem outras medidas possíveis para escolher pontos de coleta para uma rede de sensores embarcados em ônibus do transporte público. Por exemplo, a menor soma dos atrasos esperados, definida por $A_{SUM} = \sum_{b \in B} \sum_{d \in D_b} d$. Essa medida garante que, na média, o atraso experimentado pelas mensagens será o menor possível. Porém, um algoritmo que busque o menor atraso possível no caso médio, não necessariamente minimiza os atrasos considerados grandes. É possível que algumas mensagens tenham atrasos esperados inaceitáveis. Assume-se uma aplicação de emissão de alertas onde todas as mensagens são consideraras críticas. Nessa hipótese, é importante garantir que todas as mensagens sejam entregues em tempo hábil. Baseado no modelo descrito, este trabalho propõe uma heurística para minimizar o aumento do atraso causado pela remoção de um candidato a ponto de coleta. A Seção 5 formula uma solução ótima para o problema, porém computacionalmente intensa, enquanto a Seção 6 apresenta um algoritmo guloso baseado na heurística de remoção do candidato a ponto de coleta com menor atraso de remoção, dado pela Equação 3.

5. Formulação da Solução Ótima

O problema deste trabalho escolhe, dentre todos os pontos existentes, uma quantidade $N_{desejado}$ de pontos de coleta. Essa escolha busca minimizar o maior atraso possível na rede entre dois pontos de coleta, definido na Equação 2. A solução ótima para esse problema é obtida através da formulação de um problema de programação linear inteira (ILP), como se segue:

$$\text{minimizar } A_{máx} \tag{4}$$

$$\text{sujeito a } \sum_{p \in \mathcal{P}} x_p = N_{desejado}; \tag{5}$$

$$\sum_{q \in \mathcal{P}^d_{bp}} y_{bpq} = x_p \quad \forall b \in B, \quad \forall p \in \mathcal{P}^s_b; \tag{6}$$

$$\sum_{p \in \mathcal{P}^s_{bq}} y_{bpq} = x_q \quad \forall b \in B, \quad \forall q \in \mathcal{P}^d_b; \tag{7}$$

$$A_{máx} - \sum_{q \in \mathcal{P}^d_{bp}} d_{bpq} y_{bpq} \geq 0 \quad \forall b \in B, \quad \forall p \in \mathcal{P}^s_b; \tag{8}$$

$$x_p = 1 \quad \forall p \in \mathcal{I} = \left(\left(\bigcup_{b \in B} \mathcal{P}^s_b \setminus \mathcal{P}^d_b \right) \bigcup \left(\bigcup_{b \in B} \mathcal{P}^d_b \setminus \mathcal{P}^s_b \right) \right); \tag{9}$$

$$A_{máx} \in \mathbb{Z}; \quad y_{bij} \in \{0,1\} \; \forall \, b \in \mathcal{B}, \; \forall \, i,j \in \mathcal{P}; \quad x_i \in \{0,1\} \; \forall \, i \in \mathcal{P}. \tag{10}$$

O objetivo do problema, dado pela Equação 4, é minimizar $A_{máx}$. A Equação 5 especifica que a quantidade de pontos de coleta escolhidos deve ser igual a $N_{desejado}$. A Equação 6 indica que, se o ponto i for escolhido e se o ônibus b faz contato com algum outro ponto após i, então i deve ser antecessor de algum ponto j. Analogamente, a

Equação 7 indica que, se o ponto j for escolhido e se o ônibus b faz contato com algum outro ponto antes de j, então j deve ser sucessor de algum ponto i. Assim, as Equações 6 e 7 juntas definem as variáveis y_{bij} para o cálculo do atraso mostrado na Figura 3. Isto é, se em um trajeto do ônibus b os pontos i e j forem utilizados e os outros pontos entre eles não, o atraso entre esses dois pontos será utilizado para o cálculo de $A_{máx}$ na Equação 8. A Equação 9 especifica que, se um ponto i é o primeiro ou o último ponto que qualquer um dos ônibus faz contato, então i é obrigatoriamente escolhido como ponto de coleta, como visto na Seção 4.1. Finalmente, a Equação 10 define o domínio de cada variável.

6. Algoritmo de Escolha dos Pontos de Coleta

A solução do problema ótimo é NP-difícil, por se tratar de um ILP, e assim não escala. Dessa forma, propõe-se um algoritmo guloso que inicializa considerando que todos os pontos de ônibus candidatos a ponto de coleta são pontos de coleta. A cada iteração, o algoritmo remove do conjunto dos candidatos o ponto com menor atraso de remoção. A operação de remoção de um candidato a ponto de coleta é definida na Seção 4.1. O algoritmo termina quando o conjunto solução possui $N_{desejado}$ elementos ou quando todos os candidatos devem ser pontos de coleta, pois $|\mathcal{P}| - |\mathcal{I}| <= N_{desejado}$.

O algoritmo recebe como parâmetros: o conjunto $\mathcal{B}$; o conjunto $\mathcal{P}$; o conjunto $\mathcal{I}$; o vetor $\boldsymbol{trajetos}$, que contém uma lista $\boldsymbol{P_b}$ para cada elemento em $b \in \mathcal{B}$ ($\boldsymbol{P_{bi}}$ é a sequência de candidatos a ponto de coleta pelos quais b_i passa); o vetor $\boldsymbol{d_{prox}}$, que contém, em cada índice i a sequência de atrasos $\boldsymbol{D_b}$ para b; e $N_{desejado}$, que é o número de pontos que se deseja manter.

Algoritmo 1 Algoritmo de Escolha dos Pontos de Coleta

Precondições: $\mathcal{B} = \{b_1, \ldots, b_K\}, \mathcal{P} = \{p_1, \ldots, p_N\}, \mathcal{I} = \{p_{py}, \ldots, p_{pz}\}, \boldsymbol{trajetos} = (\boldsymbol{P_{b1}}, \ldots, \boldsymbol{P_{bK}}), \boldsymbol{d\_prox} = (\boldsymbol{D_{b1}}, \ldots, \boldsymbol{D_{bK}}), N_{desejado}$

se $|\mathcal{P}| - |\mathcal{I}| \leq N_{desejado}$ **então retorne** $\mathcal{I}$ ▷ Não é possível obter $N_{desejado}$ pontos
para $b \in \mathcal{B}$ **faça** ▷ Inicializar a lista de atrasos de remoção
  para $i \leftarrow 1, |\boldsymbol{P_b}|$ **faça**
    $\boldsymbol{max\_ts}[\boldsymbol{P_b}[i]] \leftarrow \max(\boldsymbol{D_b}[i-1] + \boldsymbol{D_b}[i], \boldsymbol{max\_ts}[\boldsymbol{P_b}[i]])$
$\mathcal{R} \leftarrow \emptyset$
enquanto $|\mathcal{R}| + |\mathcal{I}| + N_{desejado} < |\mathcal{P}|$ **faça** ▷ Remover os pontos até o número desejado
  para $p \in \mathcal{P}$ **faça** ▷ Selecionar um ponto com menor atraso de remoção, não removido nem ilícito
    se ($d\_min == NULO$ **ou** $d\_min < \boldsymbol{max\_ts}[p]$) **e** $p \notin I$ **e** $p \notin R$ **então**
      $d\_min \leftarrow \boldsymbol{max\_ts}[p]$
      $p\_a\_remover \leftarrow p$
  para $\boldsymbol{P_b} \in \boldsymbol{trajetos}$ **faça**
    para $i \leftarrow 1, |\boldsymbol{P_b}|$ **faça**
      se $\boldsymbol{P_b}[i] = p\_a\_remover$ **então**
        $\boldsymbol{D_b}[i-1] \leftarrow \boldsymbol{D_b}[i-1] + \boldsymbol{D_b}[i]$ ▷ Aumentar a espera do ponto anterior
        se $\boldsymbol{max\_ts}[\boldsymbol{P_b}[i+1]] < \boldsymbol{D_b}[i-1] + \boldsymbol{D_b}[i])$ **então**
          $\boldsymbol{max\_ts}[\boldsymbol{P_b}[i+1]] \leftarrow \boldsymbol{D_b}[i-1] + \boldsymbol{D_b}[i])$ ▷ Atualizar atrasos de remoção
        se $i > 1$ **então**
          se $(\boldsymbol{max\_ts}[\boldsymbol{P_b}[i-1]] < \boldsymbol{D_b}[i-2] + \boldsymbol{D_b}[i-1])$ **então**
            $\boldsymbol{max\_ts}[\boldsymbol{P_b}[i-1] \leftarrow \boldsymbol{D_b}[i-2] + \boldsymbol{D_b}[i-1]$
        Remover($\boldsymbol{P_b}[i]$) ▷ Remover o elemento i de $\boldsymbol{P_{bi}}$
  Inserir($\mathcal{R}, p$) ▷ Inserir o ponto escolhido no conjunto de pontos removidos
retorne $\{\mathcal{S} \setminus \mathcal{R}\}$

No Algoritmo 1, o vetor $\boldsymbol{max\_ts}$ mantém, para cada ponto p, seu atraso de remoção. A função Remover($\boldsymbol{P_{bi}}, p$), na linha 20, remove o ponto p da lista $\boldsymbol{P_{bi}}$.

O laço de repetição da linha 6 é o laço principal do algoritmo. A cada iteração, um ponto é escolhido para ser removido. A escolha do ponto a ser removido, entre as linhas 7 e 10, escolhe um ponto $p\_a\_remover$ que tenha o menor atraso de remoção que não esteja nem no conjunto $\mathcal{I}$ dos pontos ilícitos e nem no conjunto $\mathcal{R}$, dos pontos já removidos. A atualização dos atrasos máximos do ponto anterior e do ponto posterior a p acontece entre as linhas 15 e 19. Na linha 20, o ponto p é removido de P_b.

O algoritmo retorna o conjunto $\{\mathcal{S} \setminus \mathcal{R}\}$ de pontos de coleta escolhidos quando o conjunto possui tantos elementos quanto $|\mathcal{P}| - N_{desejado}$, ou retorna um conjunto vazio quando não é possível remover nenhum ponto, pois o número de elementos no conjunto $\mathcal{I}$ de pontos ilícitos é maior do que $|\mathcal{S}| - N_{desejado}$.

6.1. Análise da complexidade de tempo do algoritmo

As entradas do Algoritmo 1 são o conjunto $\mathcal{B}$, de tamanho K; o conjunto $\mathcal{P}$, de tamanho N; o vetor de listas $\boldsymbol{trajetos}$, de tamanho K; o vetor de listas $\boldsymbol{d\_prox}$, de tamanho K; e o inteiro $N_{desejado}$. Seja M o tamanho da maior lista em $\boldsymbol{trajetos}$. Consequentemente, a maior lista em $\boldsymbol{d\_prox}$ possui tamanho $M - 1$.

A complexidade de tempo do algoritmo é a complexidade do teste inicial, na linha 1, somada à complexidade da inicialização da lista de atrasos por remoção, na linha 2, somada à complexidade da remoção dos pontos, na linha 6.

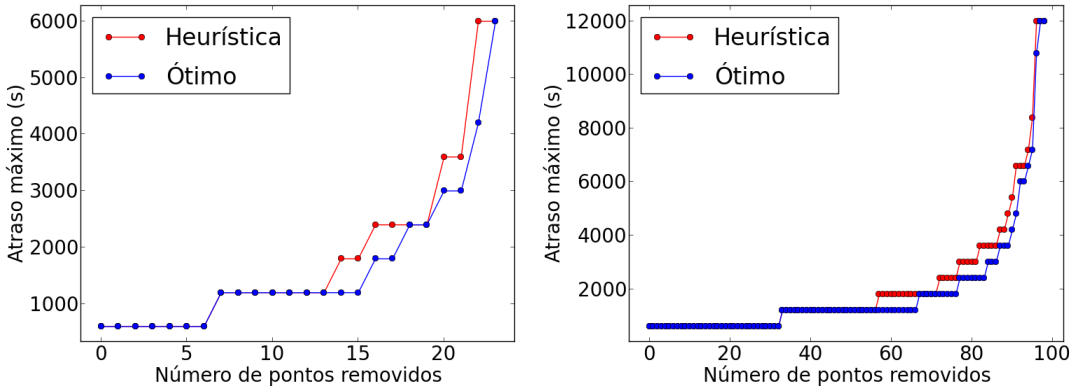
A verificação inicial envolve comparar a cardinalidade dos conjuntos $\mathcal{P}$ e $\mathcal{I}$ com $N_{desejado}$. Portanto, utilizando a notação O para análise de pior caso, a complexidade desse trecho é $O(1)$.

A operação de inicialização das listas de espera máxima de remoção envolve uma iteração sobre todos os M elementos de $\boldsymbol{P_b}$ aninhada em uma iteração sobre todos os K elementos de $\mathcal{B}$. Dessa forma, a complexidade desse trecho é $O(KM)$.

A remoção dos pontos possui um laço externo, na linha 6, no qual $|\mathcal{R}|$ cresce de uma unidade a cada iteração até, no máximo, atingir o valor de $N_{desejado}$. Assim, a complexidade desse laço é a complexidade dos laços internos multiplicada por N, que é a cardinalidade de pior caso para $\mathcal{R}$.

Aninhados ao laço da linha 6 existem dois outros laços, em sequência. Na linha 7, um laço percorre todos os N elementos de $\mathcal{P}$, verificando se $p \in \mathcal{I}$ e se $p \in \mathcal{R}$. Sabendo que os conjuntos $\mathcal{I}$ e $\mathcal{R}$ são sempre subconjuntos de $\mathcal{P}$, é possível utilizar estruturas de dados nas quais a complexidade das verificações $p \in \mathcal{R}$ e $p \in \mathcal{I}$ seja $O(1)$. Logo, a complexidade do laço da linha 7 é $O(N)$.

O laço da linha 20 percorre todos os K elementos P_b da lista $trajetos$. Aninhado ao laço da linha 20, um outro laço itera sobre todos os, no máximo, M elementos de cada P_b. Durante cada iteração, o elemento D_b do vetor $\boldsymbol{max\_ts}$ é acessado e os elementos $D_b(i-2)$, $D_b(i-1)$ e $D_b(i)$ são acessados. Para evitar percorrer toda a lista D_b até o elemento i, é possível utilizar um ponteiro em $P_b(i)$ que aponte para $D_b(i)$. Dessa forma, o acesso a $D_b(i-2)$, $D_b(i-1)$ e $D_b(i)$ dentro dessa iteração tem complexidade de pior caso $O(1)$. Assim, o laço da linha 20 possui complexidade $O(KM)$.



(a) Grade de 25 pontos, com 5 ônibus.

(b) Grade de 100 pontos, com 10 ônibus.

Figura 5. Maior atraso da rede para número de pontos removidos em uma grade 25 pontos e em uma grade 100 pontos.

A função $Inserir(\mathcal{R}, p)$ insere p no conjunto $\mathcal{R}$ e, portanto, tem complexidade $O(1)$. Dadas as complexidades dos laços internos, a complexidade do laço da linha 6 é $O(N^2 + KMN + N)$.

A complexidade de tempo do algoritmo é dada por $O(1+KM+N^2+KMN+N)$. Pela notação O, a complexidade de tempo é dada pelo maior entre $O(N^2)$ e $O(KMN)$.

6.2. Comparação com o problema ótimo

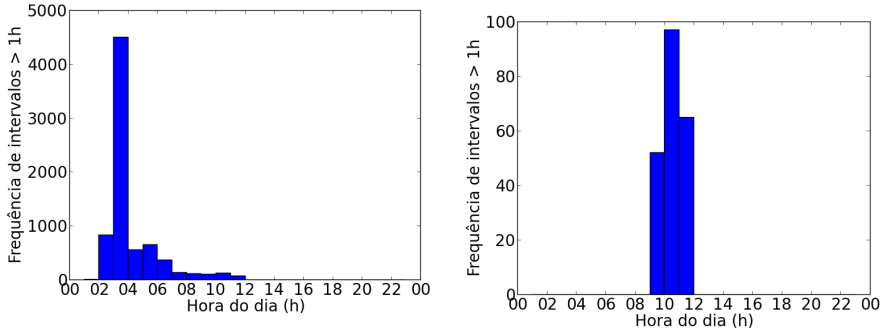
Para estimar a aproximação fornecida pelo algoritmo proposto, é realizada uma comparação entre as soluções encontradas pelo algoritmo e as soluções do problema ótimo, descrito na Seção 5. São utilizados dados sintéticos para simular um cenário de mobilidade de ônibus entre pontos de ônibus. Os pontos de ônibus são organizados em grade e é simulada a movimentação dos ônibus entre os pontos. Todos os ônibus iniciam e terminam seus trajetos no mesmo ponto. Para solução do ótimo, utiliza-se a ferramenta IBM ILOG CPLEX 12.5.1.

A Figura 5(a) mostra os resultados obtidos, para 5 ônibus percorrendo uma grade de 25 pontos, com 5 linhas e 5 colunas. Na Figura 5(b), é possível observar os resultados para 10 ônibus percorrendo uma grade de 100 pontos, com 10 linhas e 10 colunas. Pela observação dos gráficos, é possível notar que os atrasos máximos crescem em saltos, indicando que há pontos de coleta que são uma espécie de gargalo que, quando removidos, aumentam significativamente o atraso da rede. Na maior parte dos resultados, o algoritmo proposto está a menos de um gargalo de distância.

7. Utilização do Algoritmo em um Cenário Real

A Federação das Empresas de Transporte de Passageiros do Estado do Rio de Janeiro (FETRANSPOR) e a Prefeitura da Cidade do Rio de Janeiro disponibilizam dados das posições dos ônibus [IPLANRIO, 2016a] e das posições dos pontos de ônibus [IPLANRIO, 2016b] da cidade do Rio de Janeiro. A partir desses dados, gera-se as entradas do algoritmo proposto neste trabalho.

Neste trabalho, os dados são coletados por um período de 24h, desde a 0:00 h do dia 29 de novembro de 2016 até a 0:00 h do dia 30 de novembro de 2016 e inseridos em



(a) Ocorrência de atrasos maiores do que 1 h ao longo de 24 h sem filtros.

(b) Ocorrência de atrasos maiores do que 1 h ao longo de 24 h com seleção apenas de contatos entre 8:00 h e 22:00 h.

Figura 6. Distribuição de atrasos maiores que 1 h antes e após filtragem entre 8:00 h e 22:00 h.

Tabela 2. Características dos conjuntos de dados entre 8h e 22h.

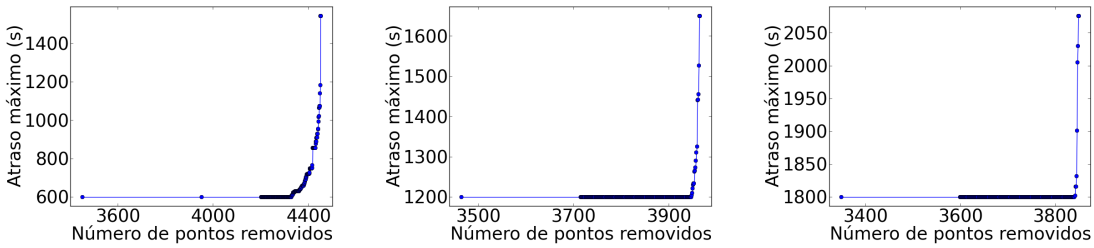
Limite de maior atraso (s)	Número de ônibus	Número de pontos	Número de pontos em $\mathcal{I}$
Sem limite	6.484	6.244	2.851
1.800	5.863	6.244	2.778
1.200	5.320	6.234	2.662
600	3.304	6.181	2.175

um banco de dados. Define-se que, caso um ônibus esteja a menos de 300 m de um ponto de ônibus, então ambos estão em contato. É selecionada a tupla (instante, ônibus, ponto de ônibus) para todos os instantes em que algum ônibus possui contato com algum ponto de ônibus. Com esses dados, são conhecidos os conjuntos $\mathcal{B}$ e $\mathcal{P}$. Para cada ônibus, são criados P_b e D_b. São obtidos dados de 6.309 pontos de ônibus e 6.627 ônibus.

Os conjuntos $\mathcal{B}$, $\mathcal{P}$ são obtidos e as listas P_b e D_b são calculadas para cada $b \in \mathcal{B}$. O gráfico da Figura 6(a) mostra a distribuição dos atrasos maiores do que 1 h ao longo das horas do dia. É possível notar um elevado número de intervalos maiores do que 1 h no período anterior às 8:00 h. Ao selecionar apenas pontos entre 8:00 h e 22:00 h é obtido o gráfico de intervalos maiores do que 1 h ilustrado na Figura 6(b). As escalas dos gráfico são diferentes, pois, do contrário, não seria possível observar os valores de ambas as figuras. Supõe-se que esses intervalos mais longos ocorram durante a madrugada pois muitos ônibus estão parados na garagem durante o período noturno.

Para eliminar o problema exposto na Figura 6(a), utiliza-se neste trabalho apenas os dados coletados entre 8:00 h e 22:00 h. Além disso, realiza-se uma nova filtragem, eliminando os ônibus que possuem algum atraso maior do que 600 s, 1.200 s ou 1.800 s. Assim, remove-se, para cada filtragem, todos os ônibus $b \in \mathcal{P}$ que não fazem contato com nenhum ponto de ônibus em um intervalo maior que o atraso máximo tolerado. Por fim, o conjunto $\mathcal{I}$ é construído. A Tabela 2 mostra características dos dados após a filtragem.

Para reduzir a influência de ruídos nos dados, utilizam-se valores médios de atraso. Assim, o atraso médio entre dois pontos $p, q \in \mathcal{P}$ é calculado a partir dos intervalos de todos os ônibus que passam pelos dois pontos em sequência. Após isso, o atraso médio é substituído em todas as sequências D_b de todos os ônibus $b \in \mathcal{B}$.



(a) Apenas ônibus sem nenhum atraso inicial maior do que 600 s.

(b) Apenas ônibus sem nenhum atraso inicial maior do que 1200 s.

(c) Apenas ônibus sem nenhum atraso inicial maior do que 1800 s.

Figura 7. O atraso máximo da rede ($A_{máx}$) em função do número de pontos de coleta removidos.

O resultado da execução do algoritmo para o conjunto de dados coletados pode ser observado na Figura 7. O eixo X de cada gráfico representa o número de pontos removidos pelo algoritmo e o eixo Y, o atraso máximo da rede. É possível notar que o atraso máximo da rede também cresce em saltos, como visto na Seção 5.

Nos três casos avaliados, são removidos mais de 55% dos pontos de ônibus sem nenhum prejuízo ao atraso máximo inicial da rede. Dessa forma, é possível atender a uma aplicação cuja tolerância a atrasos seja de até 1.800 s, 1.200 s ou 600 s instalando pontos de coleta em apenas 45% dos pontos de ônibus da cidade.

8. Conclusões e Trabalhos Futuros

As redes de sensores sem fio são uma importante ferramenta para as Cidades Inteligentes, mas o custo de implementar esse tipo de rede por uma grande área urbana é um desafio. Para tal, utilizam-se redes de sensores sem fio móveis, que podem reduzir os custos de sensoriamento e de entrega de mensagens, através da entrega oportunística de mensagens. Este trabalho explorou a mobilidade dos ônibus do transporte público de uma cidade para conferir mobilidade a nós de sensoriamento, que entregam os dados obtidos e mensagens de alerta a pontos de coleta instalados nos pontos de ônibus.

Para análise do desempenho da rede abordada, este trabalho propôs um algoritmo de distribuição dos pontos de coleta pelos pontos de ônibus. Esse algoritmo tem como objetivo minimizar o atraso máximo da rede estudada, para um dado número de pontos utilizados. O algoritmo foi comparado com a solução ótima, disponibilizada por uma formulação de programação linear inteira, obtendo resultados próximos ao ótimo. Além disso, o algoritmo mostrou que é possível obter uma rede de coleta utilizando 49% dos ônibus da cidade do Rio de Janeiro com atraso máximo esperado de 600 s, utilizando 1.917 pontos de coleta, cerca de 30% do total de pontos de ônibus da cidade.

Como trabalhos futuros, pretende-se combinar uma métrica de minimização de atraso médio com a métrica utilizada, de forma a reduzir o atraso médio na entrega das mensagens, sem prejuízos ao atraso máximo. Além disso, serão utilizadas medidas coletadas por um período maior e novos filtros para eliminação de ruídos.

Referências

Akyildiz, I. F., Su, W., Sankarasubramaniam, Y. e Cayirci, E. (2002). A survey on sensor networks. *IEEE communications magazine*, 40(8):102–114.

Chourabi, H., Nam, T., Walker, S., Gil-Garcia, J. R., Mellouli, S., Nahon, K., Pardo, T. A. e Scholl, H. J. (2012). Understanding smart cities: An integrative framework. Em *System Science (HICSS), 2012 45th Hawaii International Conference on*, p. 2289–2297. IEEE.

Cruz, P., Neto, J. B. P., Campista, M. E. M. e Costa, L. H. M. K. (2016). On the accuracy of data sensing in the presence of mobility. Em *7th International Conference Network of the Future*. NoF.

Dias, D. e Costa, L. H. M. K. (2016). Análise da capacidade de dados de uma rede de Ônibus urbanos. Em *XXXIV Simpósio Brasileiro de Telecomunicações e Processamento de Sinais*. SBrT.

Dong, W., Guan, G., Chen, Y., Guo, K. e Gao, Y. (2015). Mosaic: Towards city scale sensing with mobile sensor networks. Em *Parallel and Distributed Systems (ICPADS), 2015 IEEE 21st International Conference on*, p. 29–36. IEEE.

Ekici, E., Gu, Y. e Bozdag, D. (2006). Mobility-based communication in wireless sensor networks. *IEEE Communications Magazine*, 44(7):56 – 62.

Hancke, G. P., Hancke Jr, G. P. et al. (2012). The role of advanced sensing in smart cities. *Sensors*, 13(1):393–425.

IPLANRIO (2016a). Descrição do dataset conjunto gps ônibus. Disponível em `http://data.rio/dataset/gps-de-onibus`.

IPLANRIO (2016b). Documentação de paradas das linhas de ônibus. Disponível em `http://data.rio/dataset/pontos-de-parada-de-onibus`.

Liu, B., Brass, P., Dousse, O., Nain, P. e Towsley, D. (2005). Mobility improves coverage of sensor networks. Em *Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing*. MobiHoc.

Marjovi, A., Arfire, A. e Martinoli, A. (2015). High resolution air pollution maps in urban environments using mobile sensor networks. Em *2015 International Conference on Distributed Computing in Sensor Systems*, p. 11 – 20. IEEE.

Romer, K. e Mattern, F. (2004). The design space of wireless sensor networks. *IEEE wireless communications*, 11(6):54–61.

Wong, J. L., Jafari, R. e Potkonjak, M. (2004). Gateway placement for latency and energy efficient data aggregation [wireless sensor networks]. Em *Local Computer Networks, 2004. 29th Annual IEEE International Conference on*, p. 490–497. IEEE.

Zanella, A., Bui, N., Castellani, A., Vangelista, L. e Zorzi, M. (2014). Internet of things for smart cities. *IEEE Internet of Things Journal*, 1(1):22–32.

Zhao, D., Ma, H., Li, Q. e Tang, S. (2016). A unified delay analysis framework for opportunistic data collection. *Wireless Networks*, p. 1–13.

Zoysa, K. D., Keppitiyagama, C., Seneviratne, G. P. e Shihan, W. W. A. T. (2007). A public transport system based sensor network for road surface condition monitoring. Em *NSDR '07 Proceedings of the 2007 workshop on Networked systems for developing regions*. NSDR.