Coleta oportunista de dados em redes de Internet das Coisas através do uso de otimização discreta

Edvar Afonso L. Filho[1] **e Miguel Elias M. Campista**[1] *

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

{edvar,miguel}@gta.ufrj.br

Abstract. *The Internet of Things (IoT) is based on data collection for future processing and decision-making. In multihop Low Power and Lossy Networks (LLN) scenarios, efficient data forwarding in terms of generated traffic and energy consumption is fundamental. This paper revisits the concept of mobile agents to collect data along the agents's itinerary. The idea is to avoid sending requests to the network when non-expired contents that were opportunistically collected are available in the* cache *of a central element. In the proposed mechanism, the itinerary is composed of devices of interest and intermediate devices in a closed loop at the origin. Knapsack optimization is used to add unsolicited data opportunistically. The reward is calculated according to the popularity of the data. The simulations show that it is possible to reduce network traffic and the energy consumed by the devices when compared to traditional mobile agent data gathering model.*

Resumo. *A Internet das Coisas (Internet of Things – IoT) se baseia na coleta de dados para futuro processamento e tomada de decisão. Em cenários de redes restritas (Low Power and Lossy Network - LLN) com múltiplos saltos, o encaminhamento eficiente de dados em termos de tráfego gerado e consumo de energia se torna fundamental. Este artigo revisita o conceito de agentes móveis para realizar coleta de dados ao longo dos itinerários dos agentes. A ideia é evitar o envio de requisições para rede quando conteúdos, não expirados e coletados de forma oportunista, estão armazenados no cache de um elemento central. No mecanismo proposto, o itinerário é composto por dispositivos de interesse e dispositivos intermediários em um ciclo fechado na origem. É utilizada a otimização Knapsack para acrescentar dados não solicitados de forma oportunista. A recompensa é calculada conforme a popularidade dos dados. As simulações mostram que é possível reduzir o tráfego na rede e a energia consumida pelos dispositivos quando comparado com a coleta de agentes móveis tradicional.*

1. Introdução

Na computação ubíqua, a ideia de redes de dispositivos conectados que interagem permanentemente com ambientes e usuários é cada vez mais presente. A Internet

*O presente trabalho foi realizado com apoio do CNPq; da FAPERJ; da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior Brasil (CAPES), Código de Financiamento 001; e da Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP), processos n^o 15/24494-8 e 15/24490-2.

das Coisas (*Internet of Things* - IoT) surge como parte dessa realidade onde os dispositivos coletam informações e interagem com aplicações, usuários ou outros dispositivos conectados à Internet. Com a evolução da tecnologia e miniaturização dos componentes eletrônicos, a aplicabilidade das redes passa a ser considerado em locais antes nunca imaginados. Para se ter uma ideia, pesquisas hoje, consideram a possibilidade de coleta de informações pelos sensores até mesmo no interior de materiais, tais como concreto de paredes e revestimento de produtos [Mekki et al. 2017]. Essa nova realidade exige uma transformação na arquitetura das redes e, naturalmente, cria demanda por novos protocolos de comunicação entre os dispositivos, como por exemplo, protocolos energeticamente mais eficientes.

Dentre as tarefas que mais consomem energia dos dispositivos estão a recepção e a transmissão de dados. Na arquitetura de comunicação em múltiplos saltos, a preocupação com energia é ainda maior, dado que o tráfego gerado por qualquer um dos nós pode exigir encaminhamento até o destino final remoto, tipicamente o *gateway* da rede. Como o encaminhamento, nesse caso, é inevitável para manutenção da conectividade da rede, alternativas como a introdução de inteligência no *gateway* ou alternativas de agregação de dados vêm sendo investigadas.

A inclusão de inteligência no *gateway* visa tornar a coleta de dados mais eficiente com o suporte da computação em névoa [Bonomi et al. 2012, Aazam and Huh 2014, Rahmani et al. 2015] ou uso de técnicas de *cache* otimizadas [Mišić and Mišić 2018]. Já as técnicas de agregação de dados [Xu et al. 2012] buscam aumentar a eficiência do protocolo de comunicação através da redução da quantidade de dados enviados em direção ao *gateway*. Nesse sentido, o uso de agentes móveis (AMs) surge como uma alternativa eficiente para a coleta de dados em redes LLNs. Agentes móveis são códigos de computador que são enviados na rede com tarefas previamente programadas a serem executadas em dispositivos LLNs determinados. Quando as tarefas correspondem a uma coleta de dados, os dispositivos LLN visitados são chamados nós fontes. As tarefas a serem executadas pelos AMs podem ser, por exemplo, o pré-processamento dos dados em cada dispositivo (técnicas de compressão de dados, fusão ou agregação) para reduzir a quantidade de dados coletados. Os AMs são despachados na rede de múltiplos saltos através de um itinerário que pode ou não ser pré-determinado pelo *gateway*. Na coleta tradicional com AMs, os agentes muitas vezes precisam percorrer muitos nós intermediários para percorrer o itinerário programado entre dois nós fontes consecutivos. As propostas tradicionais de AMs ignoram a coleta de dados nesses dispositivos intermediários, os quais são usados apenas para encaminhamento. Muitas vezes, estes dispositivos contêm dados atualizados e relevantes que podem ser de interesse dos usuários da rede. O emprego de AMs vem sendo rediscutido como uma forma de maximizar a quantidade de dados coletados da rede [Gavalas et al. 2017, Lu et al. 2019].

Este trabalho apresenta a proposta Agente-Knap que combina o uso de itinerários eficientes com coleta oportunista de dados para redução do consumo de energia dos nós em uma rede IoT. Para isso, a proposta revisita o conceito de AMs, que percorrem um itinerário, ou caminho, que inclui nós fontes de um determinado dado de interesse, como definido a priori pelo *gateway* da rede para a coleta. Diferente de propostas anteriores [Lu et al. 2019, Chou and Nakajima 2018, Liu et al. 2016], porém, há a coleta oportunista de dados, o que significa que dados podem ser coletados nos nós intermediários,

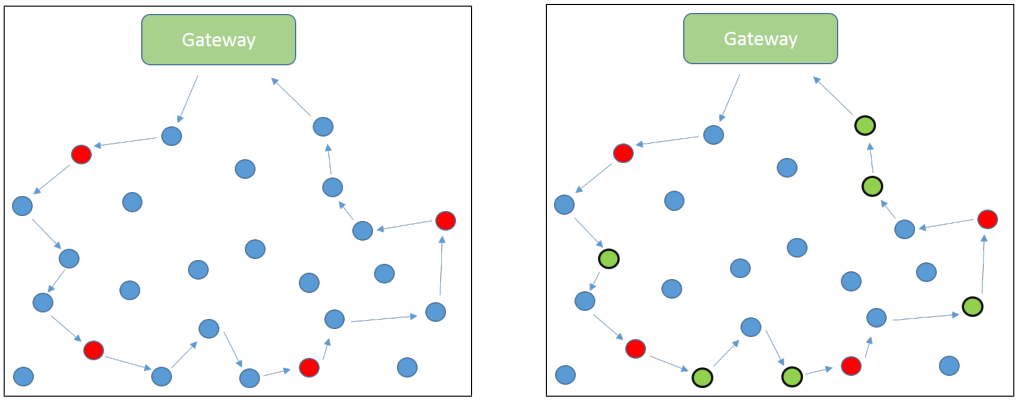
mesmo se estes não forem nós fontes dos dados de interesse, e agregados ao agente. Como o encaminhamento das mensagens é feito por múltiplos saltos, a coleta oportunista pode ser realizada de forma eficiente através do uso do método de otimização discreta *Knapsack*. O *Knapsack* se serve das diferentes prioridades que são dadas para os distintos tipos de dados existentes nos dispositivos da rede IoT. A associação feita na abordagem proposta considera que, as prioridades dos conteúdos funcionam como os "valores" dos itens do problema *Knapsack* e o espaço de carga para coletar os dados nas mensagens (*payload*) assume o papel da capacidade da "mochila". Outra contribuição do trabalho é aplicar técnicas de priorização de *cache* de dados no cálculo das prioridades dos conteúdos usadas no problema *Knapsack*. Assim como é aplicado em algumas técnicas de *cache*, o lista ordenada com as prioridades dos tipos dos dados é estabelecido conforme a popularidade de cada um. O cálculo da prioridade é feito através de uma combinação das técnicas *Least Frequently Used* (LFU) e *Least Recently Used* (LRU). Nas simulações feitas, é possível verificar que a proposta aqui apresentada permite redução de até 43% no consumo de energia da rede, redução no período de atualização dos conteúdos e redução do tráfego de mensagens na rede quando comparada à coleta de dados tradicional de AMs.

O restante deste artigo é organizado da seguinte forma: a Seção 2 introduz a arquitetura de rede considerada no trabalho e descreve o problema atacado. A Seção 3 apresenta o mecanismo de coleta proposto, o Agente-Knap. Na sequência, a Seção 4 descreve o ambiente de simulação e a Seção 5 apresenta os resultados alcançados. A Seção 6 lista os trabalhos relacionados e, finalmente, a Seção 7 conclui este trabalho e apresenta os trabalhos futuros.

2. Arquitetura da Rede e Descrição do Problema

A rede considerada neste trabalho é composta por um *gateway* e dispositivos fontes de dados. Cada dispositivo é capaz de coletar dados de um determinado conteúdo, sendo que mais de um dispositivo pode oferecer dados da mesma classe de conteúdo. Sendo assim, as consultas a serem realizadas a um determinado tipo de dado oferecido pela rede são inicialmente recebidas pelo *gateway* e, em seguida, coletadas da rede através de requisições originadas no próprio *gateway*. O *gateway* pode coletar dados de todos os nós fontes do tipo solicitado, sendo esses os mais atuais possíveis. Logo, a arquitetura proposta é centralizada no *gateway* e, dessa forma, toda inteligência do sistema existe nesse elemento central. Para participar da rede gerenciada pelo *gateway*, cada dispositivo deve registrar-se e informar qual tipo de conteúdo é capaz de prover, além do tamanho, em Bytes, desse conteúdo. Ainda como premissa, é assumido que o *gateway* possui informações sobre a posição geográfica de todos os dispositivos, assim como a potência de transmissão e recepção dos rádios, para realizar o cálculo estimado do roteamento na rede.

As propostas relacionadas à aplicação de Agentes Móveis (AMs) em redes LLN, podem ser classificadas em dois tipos: Planejamento Estático de Itinerários ou Planejamento Dinâmico de Itinerários. O primeiro tipo existe quando o planejamento do itinerário é todo feito pelo *gateway* e registrado no código do AM antes que este seja despachado na rede. O Planejamento Estático de Itinerário possui a vantagem de transferir o processamento mais complexo para o elemento que possui o maior poder de processamento na rede. Porém, exige que o *gateway* periodicamente requisite atualizações dos estado dos dispositivos na rede. No tipo Dinâmico, o AM tem autonomia para encontrar



(a) Coleta de dados na rede com o uso tradicional de agente móvel. Os dados são coletados nos nós fontes, indicados na cor vermelha. Os dados dos nós intermediários (cor azul) são ignorados pelo agente.

(b) Coleta de dados na rede com o uso do mecanismo Agente-Knap. Os dados são coletados nos nós fontes, indicados na cor vermelha, e nos nós intermediários selecionados, indicados na cor verde.

Figura 1. Diferenças entre os mecanismos de coleta de dados tradicional com o uso de agentes móveis e a coleta de dados Agente-Knap.

os melhores caminhos para percorrer a rede entre os nós fontes definidos. Essa abordagem exige um gasto maior de energia com processamento nos dispositivos da IoT, restritos em energia, mas elimina a necessidade de periódicas atualizações do *gateway*. Este artigo considera apenas o primeiro tipo, ou seja, o Planejamento Estático de Itinerários.

A Figura 1(a) ilustra uma aplicação típica de coleta com Planejamento Estático de Itinerários. A figura exemplifica uma configuração tradicional, com o uso de um agente móvel em uma rede LLN. Nesse exemplo, o agente é programado pelo *gateway* para realizar a coleta em quatro nós fontes, que aparecem em vermelho na figura.

Essa ideia é capturada pela proposta deste trabalho, que aproveita os itinerários programados para coleta de dados também dos nós intermediários. A Figura 1(b) apresenta um exemplo da coleta proposta pelo Agente-Knap. O agente proposto, além dos dados provenientes dos nós fontes em vermelho, considera também os conteúdos existentes em um subconjunto de nós intermediários, indicados pela cor verde. Os nós intermediários, provedores dos dados não solicitados, são selecionados baseado em um problema de otimização para o aumento da eficiência da coleta. O mecanismo proposto é descrito a seguir em maiores detalhes.

3. Agente-Knap Proposto

O objetivo principal do Agente-Knap consiste em aumentar a eficiência da coleta de dados utilizando AMs em uma rede de múltiplos saltos. Os dados coletados são armazenados no cache do *gateway* com um tempo de validade informado pelos dispositivos servidores dos dados. A ideia central é que os conteúdos armazenados no cache, enquanto ainda válidos, evitem que novas requisições sejam encaminhadas à rede. Dessa forma, é possível economizar recursos dos dispositivos LLNs a partir da redução da quantidade de requisições disparadas na rede.

O AM da proposta Agente-Knap é definido como um código de computador com estrutura similar ao agente descrito no trabalho [Chen et al. 2006], onde o AM possui

campos reservados para o código de processamento, para a lista de nós fontes que devem ser visitados, para a indicação do próximo salto, que pode ser um nó fonte ou um nó intermediário, e uma parte dedicada ao armazenamento dos dados coletados. O código de processamento é a principal parte do AM e é responsável pela execução de tarefas previamente programada para serem executadas nos dispositivos. Por exemplo, o pré-processamento dos dados coletados, permitindo a redução do tamanho da mensagem transmitida na rede.

3.1. Modelo de Coleta Utilizado

Antes de enviar o AM na rede, o *gateway* deve calcular o itinerário, que é a sequência de nós fontes que devem ser visitados e os respectivos nós intermediários. Desta forma, o Agente-Knap proposto realiza um ciclo de coleta de dados nos nós intermediários de forma oportunista, enquanto o encaminhamento é feito pelos múltiplos saltos.

A ideia central da proposta é a baseada no gerenciamento do espaço de carga útil (*payload*) a ser preenchido no código do AM. Esse espaço carrega os dados dos dispositivos a medida que avança pelo itinerário. Antes da transmissão do AM pelo *gateway*, um tamanho máximo (P) do *payload*, em Bytes, é conhecido pelo *gateway*. Esse *payload*, deve ser divido em duas partes: um espaço fixo reservado para a coleta garantida dos conteúdos nos nós fontes (G), que são os alvos prioritários do agente de coleta; e um espaço (C) utilizado para a concatenação dos conteúdos da coleta oportunista realizada nos nós intermediários.

O problema *Knapsack* que é solucionado pelo *gateway* deve considerar o tamanho C como a capacidade da "mochila", para selecionar quais conteúdos oportunistas devem preencher o *payload*. Os conteúdos oportunistas são adicionados ao *payload* do AM de forma concatenada, sem violar o tamanho máximo C. A Figura 2 apresenta o conceito através de um exemplo, no qual o *payload* foi preenchido com quatro conteúdos de mesmo tipo coletados de nós fontes no espaço reservado G e quatro conteúdos diferentes concatenados de maneira oportunista no espaço C. A combinação entre coleta oportunista e cálculo do itinerário são os dois principais desafios abordados pelo Agente-Knap, como descritos a seguir.

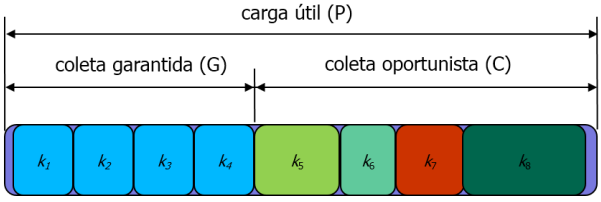


Figura 2. Carga útil P (*payload*) do AM com os espaços para coleta garantida (G) e para coleta oportunista (C).

Para modelagem do sistema, deve-se considerar a existência de um número m de tipos de dados ou conteúdos existentes na rede, representados pelo conjunto $\mathcal{K} = \{k_1, ..., k_m\}$. A Figura 3 apresenta o diagrama em blocos do *gateway* e a relação entre os módulos. Os blocos Topologia da rede, Tamanho dos conteúdos e Tamanho para coleta oportunista representam parâmetros de entrada do sistema que devem ser informados na

inicialização. Os conteúdos estão distribuídos pelos n nós da rede, representados pelo conjunto $\mathcal{N} = \{n_1, ..., n_n\}$. Para o escopo deste trabalho e simplificação da análise, considera-se que cada nó da rede fornece informações para apenas um conteúdo, apesar do modelo proposto ser facilmente adaptável para cenários onde os dispositivos são servidores de mais de um tipo de conteúdo. Inicialmente, o *gateway* recebe requisições da Internet, R_j ($j = \{1, 2.., p\}$ e $p \to \infty$), enviadas por clientes solicitantes do sistema, para os diferentes conteúdos, no instante de tempo t_j. É assumido que as requisições chegam ao *gateway* de forma sequencial. Após receber requisições, o *gateway* considera diferentes conjuntos de nós fontes, $\mathcal{N}_f \subset \mathcal{N}$, e as informações de topologia da rede para cálculo do itinerário. Essas informações, conhecidas pelo *gateway* como condição para participação na rede, são entradas do módulo Cálculo de Itinerário. Nesta etapa, são considerados os algoritmos *Dijkstra* e *Christofides* para cálculo do itinerário otimizado para a coleta dos dados. Com o recebimento das requisições, o *gateway* também armazena cada instante de tempo t_j em um banco de dados local para registro do histórico e cálculo da pontuação *Knapsack* para cada conteúdo. O algoritmo *Christofides* é uma das heurísticas utilizadas para a solução do Problema do Caixeiro-Viajante (PCV), que é o tipo de problema a ser resolvido para cálculo do itinerário do AM. A saída do algoritmo é a sequência de nós que devem ser visitados e a solução é determinada após os cálculos dos caminhos entre pares de nós e os respectivos custos determinados pelo algoritmo *Dijkstra*. *Christofides* é uma solução para o PCV que, no pior caso, garante uma razão de menos de 3/2 entre a solução encontrada e a solução ótima [Christofides 1976].

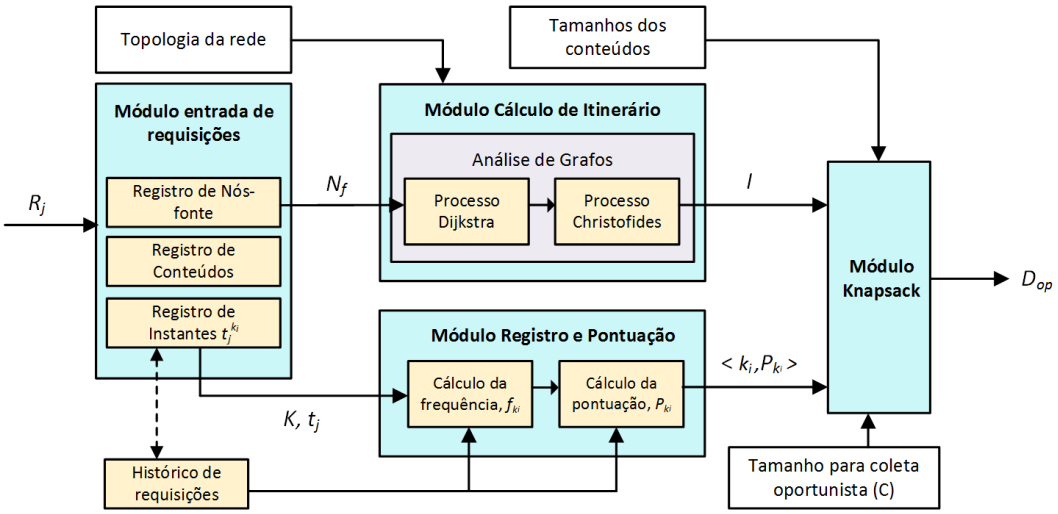


Figura 3. Estrutura em blocos do sistema no *gateway*.

Ao receber cada requisição, o *gateway* inicialmente verifica se existem dados atualizados no *cache*, na quantidade de nós fontes desejados. Se a requisição for integralmente atendida pelos dados em *cache*, o *gateway* envia a resposta para o cliente solicitante sem a necessidade de enviar requisições para a rede de dispositivos IoT. Caso não haja dados suficientes atualizados no *cache*, o *gateway* envia um agente móvel para a rede com o objetivo de coletar informações apenas para a quantidade de nós fontes necessária para completar a informação desejada. Por exemplo, caso seja recebida uma requisição para coleta de informação de temperatura de sete dispositivos estratégicos na rede e o *gateway* só tenha informação atualizada em *cache* de quatro dispositivos, o *gateway* calcula o iti-

nerário e despacha um agente móvel para realizar a coleta em apenas três nós fontes.

A saída do Módulo de Cálculo de Itinerário é o itinerário, I, contendo todos os nós que serão percorridos pelo AM até seu retorno para o *gateway*. I, portanto, é uma das entradas do Módulo *Knapsack*. As outras entradas do Módulo *Knapsack* são: um conjunto de tuplas $\langle k_i, P_{k_i} \rangle$, que relaciona os conteúdos k_i servidos pelos nós do itinerário com suas respectivas pontuações P_{k_i}, o tamanho em Bytes de cada conteúdo e o espaço do *payload* a ser considerado, C, em Bytes. O cálculo da pontuação para cada conteúdo é feita através da Equação 1:

$$P_{k_i} = \alpha \left(\frac{1}{t_j - t_{j-1}} \right) + (1 - \alpha) f_{k_i}, \tag{1}$$

onde f_{k_i} é a frequência das requisições recebidas para o conteúdo k_i. A Equação 1 é dividida em duas parcelas. A primeira é inversamente proporcional ao intervalo de tempo entre duas requisições consecutivas para um dado conteúdo k_i, ou seja, quanto mais próximos os instantes de recebimento de duas requisições consecutivas para k_i, maior a pontuação P_{k_i} correspondente. A segunda parcela é diretamente proporcional à frequência de requisições recebidas para o conteúdo k_i. Quanto mais frequentes são as requisições para um determinado conteúdo k_i, maior a pontuação P_{k_i}.

O parâmetro α existe para permitir um balanceamento entre os pesos de cada uma das parcelas. Valores de α próximos de um, por exemplo, permitem que uma rajada de requisições para um determinado conteúdo gere uma alta pontuação, enquanto valores de α próximos de zero resultam em maior pontuação para os conteúdos mais populares. Com os valores, P_{k_i}, os tamanhos de cada conteúdo e o tamanho máximo para a coleta oportunista, C, o problema *Knapsack* é solucionado pelo *gateway*, onde são determinados quais conteúdos e em quais nós podem ser coletados oportunisticamente; esta saída é D_{op}. Dessa forma, é garantido o preenchimento ótimo do *payload* no agente de coleta pelos conteúdos mais valiosos disponibilizados pelos nós do itinerário. Isso ocorre sem que seja violado o tamanho máximo desejado para preenchimento do *payload*.

4. Ambiente de Simulação

Um ambiente de simulação utilizando a linguagem *python* foi desenvolvido para avaliação da proposta deste trabalho. Uma topologia em grade 6×10, com um nó representando o *gateway*, foi utilizada nos experimentos. Os sessenta nós da grade possuem oito vizinhos cada, com exceção dos nós nas linhas e colunas das bordas da topologia. O *gateway* é fixo e posicionado no centro da rede. O *gateway* possui conexão apenas com os quatro nós mais próximos e se comunica através de múltiplos saltos com o restante da rede. Os custos dos enlaces foram distribuídos aleatoriamente com valores entre 1 e 5. Os pesos dos enlaces são sempre inteiros.

A distribuição pela rede de quatro tipos de conteúdo, k_1, k_2, k_3 e k_4, foi considerada em todas as simulações. A distribuição dos conteúdos pelos diferentes dispositivos foi feita de forma aleatória e uniforme na inicialização do sistema. Todos os dispositivos são inicializados com um conteúdo de tamanho 12 Bytes, que é aproximadamente o tamanho de conteúdos considerados em outros experimentos de pesquisas relacionadas a IoT [Baccelli et al. 2014].

Na estratégia de coleta de dados simulada, cada requisição recebida dos solicitantes pelo *gateway* identifica um tipo de conteúdo do qual deseja-se extrair informações. Cada requisição também determina a quantidade de nós fontes necessária para a coleta de forma a garantir precisão mínima da informação sensoreada desejada. Em cada requisição não são informadas as identificações de quais dispositivos os conteúdos devem ser coletados mas sim a quantidade de dispositivos. Essa estratégia foi adotada para as simulações da coleta tradicional e para a coleta proposta. Nas simulações, a chamada coleta tradicional é aquela realizada pelo agente móvel apenas nos nós fontes solicitados. Já a coleta Agente-Knap considera dados de outros tipos de conteúdos, sendo esses os tipos disponíveis nos nós intermediários do itinerário.

Os experimentos desenvolvidos nas simulações são de três tipos: Sensibilidade à popularidade dos conteúdos, Consumo de energia médio nos dispositivos e Quantidade de *cache-hits*.

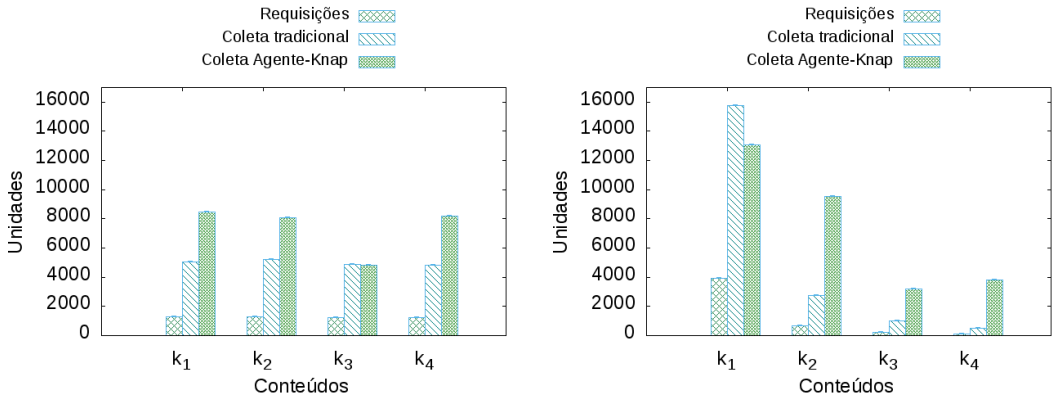
Todos os ciclos de coleta, quando necessários, são disparados a partir da chegada de requisições vindas da Internet e recebidas pelo *gateway*. Tais requisições são recebidas com uma taxa de chegada que segue uma distribuição de Poisson com $\lambda = 5$. As distribuições de probabilidades dos conteúdos entre as requisições foram a Uniforme ou Zipf. Todos os experimentos são realizados com a chegada de mil requisições no *gateway*, parâmetro s_{zipf} igual a 2,0 e α igual a 0,001. A exceção foram os testes de Sensibilidade à popularidade dos conteúdos, onde o parâmetro α foi variado e foram enviadas cinco mil requisições.

5. Resultados

5.1. Sensibilidade à popularidade dos conteúdos

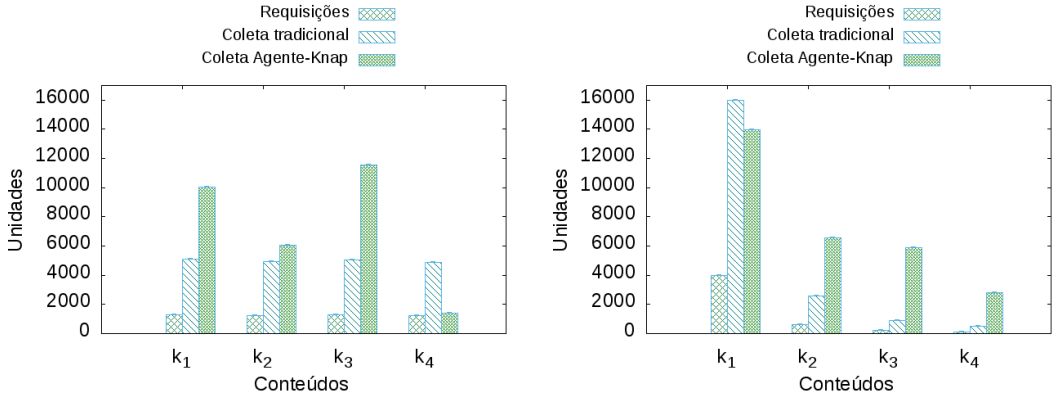
O objetivo deste experimento é verificar o desempenho do sistema proposto em um cenário onde os conteúdos distribuídos na rede IoT possuem diferentes popularidades. Aqui foi considerado que a popularidade de um conteúdo está diretamente relacionada ao histórico de requisições para o conteúdo, quanto mais requisições no histórico, mais popular é o conteúdo. As simulações foram executadas considerando duas distribuições de popularidade entre os conteúdos: distribuição uniforme e distribuição Zipf. Assim como no trabalho de Zhang et al [Zhang et al. 2015], a distribuição Zipf foi escolhida porque não foi encontrado nos trabalhos relacionados um modelo de distribuição de tráfego para acessar redes LLN/WSN. O indicador utilizado para verificar o desempenho foi a quantidade de unidades de informação coletadas na rede. Cada unidade de informação considerada corresponde a um dado de 12 Bytes enviado por cada dispositivo em resposta a uma requisição. Para pequenos valores de α, é esperado que haja maior correlação entre a popularidade dos conteúdos e a quantidade de unidades de informação coletadas da rede referente ao mesmo conteúdo. Para todos os testes com a distribuição Zipf, foi utilizado o parâmetro s_{zipf} = 2,0. Como o objetivo é testar a sensibilidade da estratégia de coleta, nesta simulação não foi considerada a influência do *cache*. Ou seja, nenhum dado foi armazenado em *cache*.

No teste, foi comparado o desempenho da coleta Agente-Knap para dois valores do parâmetro α, 1 e 0,001. Em todos os testes foram considerados quatro nós fontes nas coletas. Para a coleta proposta, o C foi simulado com o tamanho de 60 Bytes. Para cada requisição recebida, na coleta tradicional, os conteúdos foram coletados pelo agente



(a) $\alpha = 1$, Distribuição Uniforme. (b) $\alpha = 1$, Distribuição Zipf.

Figura 4. Sensibilidade do sistema de coleta Agente-Knap à popularidade dos conteúdos com α = 1.



(a) $\alpha = 0.001$, Distribuição Uniforme. (b) $\alpha = 0.001$, Distribuição Zipf.

Figura 5. Sensibilidade do sistema de coleta Agente-Knap à popularidade dos conteúdos com α = 0,001.

móvel apenas nos nós fontes, totalizando 48 Bytes. Para a coleta Agente-Knap, cada requisição gerou respostas com, além dos 48 Bytes dos nós fontes, aproximadamente mais 60 Bytes, ou cinco unidades de informação.

As Figuras 4(a) e 4(b) apresentam os resultados para os testes com α = 1. Para cada conteúdo, as barras representam a quantidade de requisições recebidas e as unidades de informação recebidas na coleta tradicional e na coleta Agente-Knap. É possível verificar que com um valor de α mais próximo de 1, há baixa correlação entre a popularidade de um conteúdo e a quantidade de unidades de informação coletadas para o mesmo conteúdo. Esse comportamento muda para menores valores de α. Nas Figuras 5(a) e 5(b) é possível verificar uma maior correlação, principalmente quando é utilizada a distribuição Zipf, onde as diferenças entre as popularidades dos conteúdos é acentuada. Os resultados também mostram que a quantidade de informação coletada foi maior para a coleta tradicional quando os conteúdos são mais populares. Essa característica se explica pela distribuição dos conteúdos nos itinerários. Por exemplo, se um itinerário calculado possui dez conteúdos, sendo dois conteúdos k_1, dois conteúdos k_2 e seis conteúdos k_4, mesmo que o conteúdo k_1 seja o mais popular, a otimização *Knapsack* irá preencher o

espaço C com muitas unidades de informação do conteúdo k_4. Ou seja, o fato de k_1 não ser o conteúdo mais densamente distribuído na rede, apesar de ser o mais requisitado, resulta em uma coleta mais equilibrada entre os conteúdos na proposta Agente-Knap. Dependendo dos requisitos das aplicações e da estratégia de *cache* desejada, essa característica pode ser positiva pois torna o processo de coleta mais diversificado, sem violar a ordenação de prioridades calculada para os conteúdos.

5.2. Consumo de energia nos dispositivos

Neste experimento foi realizada a comparação do consumo de energia nos dispositivos da rede entre a estratégia de coleta tradicional com AM e a coleta Agente-Knap. Para cada dispositivo IoT, o consumo de energia foi registrado, em Joules, para a execução de duas tarefas: A transmissão de mensagens, onde o consumo considerado foi de 9,72μJ para cada Byte transmitido; e a recepção de mensagens, onde o consumo de energia foi de 8,22μJ para cada Byte recebido. Foi considerado que o agente móvel é despachado pelo *gateway* para realizar a coleta com tamanho inicial de 200 Bytes. Também foi considerado o uso do *cache* no *gateway* para armazenamento dos dados coletados. Os valores de consumo de energia utilizados foram os mesmos adotados em [Jin et al. 2016] para testes com os sensores *Tmote Sky*.

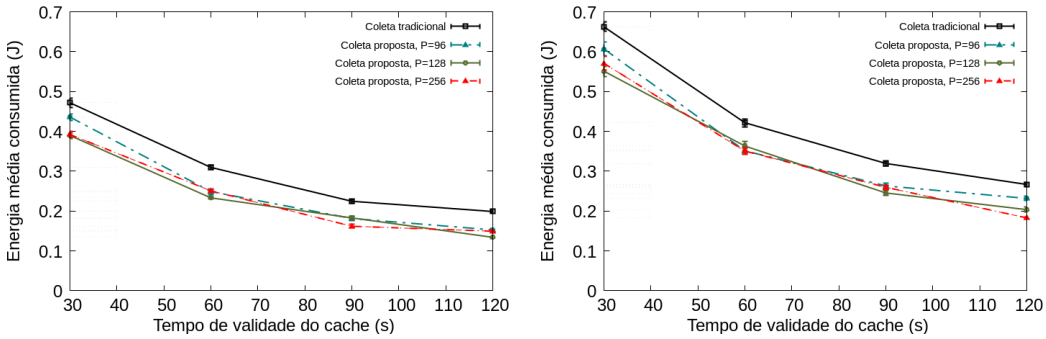
Os testes foram realizados requisições de atualização para cinco e sete nós fontes. Em ambos casos, os valores de P simulados variaram entre 96, 128 e 256 Bytes. Os resultados são apresentados nas Figuras 6(a) e 6(b) mostram as relações entre o consumo de energia médio na rede e o prazo de validade dos conteúdos armazenados no *cache* do *gateway*. Neste teste, foi considerado que todos os conteúdos possuem o mesmo prazo de validade. Foram testados prazos de 30, 60, 90 e 120 segundos. Para cada ponto dos gráficos, foram realizadas 10 rodadas de simulação, com envio de 1000 requisições cada. Foi utilizado nível de confiança de 95%.

Para ambos experimentos, é possível verificar que a coleta Agente-Knap tem menor consumo médio de energia para todos os valores de tempo de validade em *cache*. No teste com cinco nós fontes, o maior ganho da proposta Agente-Knap foi alcançado com valor de *payload* P do agente móvel de tamanho 256 Bytes e com tempo de validade do conteúdo no *cache* de 90 segundos. Neste ponto, o consumo médio da coleta tradicional foi de 224 mJ e o da coleta Agente-Knap foi de 161 mJ, uma redução de 28,12% no consumo médio de energia. No teste com sete nós fontes, o maior ganho da proposta Agente-Knap foi alcançado com valor de *payload* P do agente móvel de 256 Bytes e com tempo de validade do conteúdo no *cache* de 120 segundos. Neste ponto, o consumo médio da coleta tradicional foi de 266 mJ e o da coleta Agente-Knap foi de 182 mJ, uma redução de 31,57% no consumo médio de energia.

Apesar da escolha de P com 256 Bytes ter sido o maior ganho da coleta Agente-Knap em ambos os casos, foi observado que esse tamanho de *payload* não foi o que apresentou o maior ganho em todos os diferentes tempos de validade do *cache*. Para cada tempo diferente de validade dos conteúdos em *cache*, um diferente tamanho de *payload* foi mais eficiente com o uso da coleta proposta.

5.3. Quantidade de *cache-hits*

O objetivo da simulação Quantidade de *cache-hits* foi verificar o desempenho da estratégia pró ativa de atualização dos conteúdos no *cache* atuando em conjunto com a

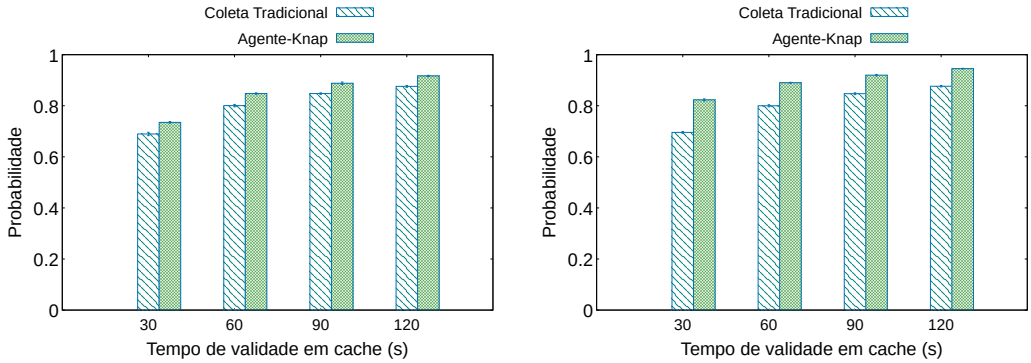


(a) Cinco nós fontes, P = 96, 128 e 256 Bytes. (b) Sete nós fontes, P = 96, 128 e 256 Bytes.

Figura 6. Consumo médio de energia na rede: Comparação entre os mecanismos de coleta tradicional e Agente-Knap, considerando diferentes tempos de validade dos conteúdos em *cache*.

estratégia de coleta oportunista proposta. Nessa análise, foi considerado que um *cache-hit* significa que um dado solicitado foi respondido pelo *gateway* com a informação disponível no *cache*. Por exemplo, caso uma solicitação seja feita para sete nós fontes do conteúdo temperatura e o *gateway* possua em *cache* a informação atualizada de quatro dispositivos de temperatura, são contabilizados quatro *cache-hits* apesar do *gateway* ainda precisar despachar o agente móvel para coleta em três nós fontes da rede. Os testes foram feitos com P do agente móvel para dois tamanhos distintos: 96 e 256 Bytes. Foram feitas simulações com requisições de coleta para sete nós fontes. Foram feitas dez rodadas para cada tempo de validade do *cache* com envio de 1000 requisições cada. Foi utilizado intervalo de confiança de 95%.

As Figuras 7(a) e 7(b) apresentam os resultados respectivamente para os testes com 96 e 256 Bytes. É possível identificar que o desempenho da coleta Agente-Knap aumenta com o crescimento do espaço do *payload* (P) para a coleta. Com P de 256 Bytes, o maior ganho com a coleta Agente-Knap foi de 12,74% com tempo de validade em *cache* de 30 segundos. Com P de 96 Bytes, o maior ganho com a coleta Agente-Knap foi de 4,72% com tempo de validade em *cache* de 60 segundos.



(a) Sete nós fontes, P = 96 Bytes, tempos de validade de 30, 60, 90 e 120 segundos. (b) Sete nós fontes, P = 256 Bytes, tempos de validade de 30, 60, 90 e 120 segundos.

Figura 7. *Cache-hits*: Comparação entre os mecanismos de coleta tradicional e Agente-Knap para diferentes tempos de validade dos conteúdos em *cache*.

6. Trabalhos Relacionados

A pesquisa de otimização da coleta de dados já foi tema de alguns trabalhos no contexto de redes LLN e IoT. O uso de agentes móveis para coleta dos dados nas redes LLN surge em trabalhos direcionados para redes de sensores, onde as pesquisas consideraram novas abordagens para reduzir o consumo de energia e tráfego de dados em redes LLN. Dong et al. consideram o uso de agentes móveis com cálculo dinâmico do itinerário de múltiplos agentes móveis controlados por um *gateway* também móvel [Dong et al. 2014]. O encaminhamento do agente móvel é feito por múltiplos saltos e, como o *gateway* é móvel, o cálculo do itinerário para retorno ao *gateway* é feito pelo agente móvel de forma autônoma e dinâmica. Para isso é proposto o uso de roteamento geográfico, onde cada dispositivo da rede tem informações sobre sua posição geográfica que são processadas para o roteamento até o destino. Os focos da proposta são uma distribuição mais uniforme do consumo de energia nos dispositivos e um roteamento autônomo do agente móvel para realizar a coleta e garantir o atendimento aos requisitos de serviço dos usuários. Gavalas et al. propõem um mecanismo de itinerários estáticos para múltiplos agentes móveis e considera principalmente a eficiência no consumo de energia influenciada pelo aumento do tamanho do agente móvel a medida que o mesmo avança no itinerário [Gavalas et al. 2017] e agrega informações dos nós do itinerário. Diferentemente da proposta apresentada neste artigo, ambos trabalhos não consideram que dados relevantes também podem ser coletados oportunisticamente nos dispositivos intermediários, enquanto o agente móvel é encaminhado, através do itinerário, de um nó fonte de dados para outro. Pela pesquisa realizada para o desenvolvimento desta proposta não foram encontrados outros trabalhos na literatura que considere o mecanismo de coleta oportunista de dados ou a relevância dos conteúdos, como o grau de popularidade, para aumentar a eficiência na coleta de dados com uso de agentes móveis.

A coleta oportunista de dados em redes de sensores e em redes IoT tem sido tema relevante em pesquisas recentes [Aguilar et al. 2017, Yang et al. 2017, Zhan et al. 2018, Xu et al. 2018]. Porém, os trabalhos consideram a coleta oportunista nos dispositivos baseada em curtos intervalos de tempo de comunicação, estabelecidos com os nós vizinhos, na maior parte dos casos a um salto de distância. Diferente dessas propostas, a contribuição apresentada neste trabalho considera a coleta oportunista feita na rede em nós que podem estar a muitos saltos de distância, enquanto as mensagens são encaminhadas em direção ao destino.

A seleção de quais conteúdos devem ser priorizados na coleta oportunista é outra contribuição deste trabalho, onde uma pontuação é atribuída aos diferentes conteúdos baseada no histórico de solicitações dos conteúdos pelos clientes. O mecanismo é similar às estratégias de *cache* em pesquisas de redes tradicionais [Li et al. 2016] ou redes IoT [Zhang et al. 2015].

7. Conclusão e Trabalhos Futuros

Este trabalho propôs um novo mecanismo para coleta de dados em redes LLN utilizando agentes móveis, com planejamento de itinerário estático e simples. O mecanismo proposto melhora a eficiência da comunicação na rede através da redução do tráfego de requisições enviadas para rede com o uso da coleta oportunista de dados para atualização pró ativa do *cache*. Nos resultados, foi possível verificar o potencial do mecanismo de coleta através da redução do consumo de energia na rede. Até onde se sabe, este é o primeiro

trabalho a propor o uso de otimização discreta *Knapsack* para realizar a coleta oportunista de conteúdos em redes com o uso de agentes móveis. A associação das pontuações dos conteúdos às popularidades dos mesmos permitiu a otimização do processo de coleta inteligente, priorizando os itens de maior interesse.

Como trabalhos futuros, pretende-se aprimorar a eficiência do mecanismo de coleta oportunista através da variação dinâmica do espaço reservado para coleta dos dados oportunistas, C, baseado na pontuação dos conteúdos disponíveis no itinerário; aplicar técnicas de agregação dos dados, visando uma redução dos conteúdos inseridos no *payload* e incluir um processo para seleção dos nós fontes relacionado com requisitos de Qualidade de Serviço das aplicações dos usuários. Por fim, pretende-se implementar o mecanismo de coleta proposto em uma aplicação real para redes IoT, incluindo a análise do desempenho do mecanismo em um cenário com perdas na comunicação.

Referências

Aazam, M. and Huh, E.-N. (2014). Fog computing and smart gateway based communication for cloud of things. In *Future Internet of Things and Cloud (FiCloud), 2014 International Conference on*, pages 464–470. IEEE.

Aguilar, S., Vidal, R., and Gomez, C. (2017). Opportunistic sensor data collection with bluetooth low energy. *Sensors*, 17(1):159.

Baccelli, E., Mehlis, C., Hahm, O., Schmidt, T. C., and Wählisch, M. (2014). Information centric networking in the IoT: experiments with ndn in the wild. In *Proceedings of the 1st ACM Conference on Information-Centric Networking*, pages 77–86. ACM.

Bonomi, F., Milito, R., Zhu, J., and Addepalli, S. (2012). Fog computing and its role in the internet of things. In *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*, pages 13–16. ACM.

Chen, M., Kwon, T., Yuan, Y., and Leung, V. C. (2006). Mobile agent based wireless sensor networks. *Journal of computers*, 1(1):14–21.

Chou, Y.-C. and Nakajima, M. (2018). A clonal selection algorithm for energy-efficient mobile agent itinerary planning in wireless sensor networks. *Mobile Networks and Applications*, 23(5):1233–1246.

Christofides, N. (1976). Worst-case analysis of a new heuristic for the travelling salesman problem. Technical report, Carnegie-Mellon University, Pittsburgh.

Dong, M., Ota, K., Yang, L. T., Chang, S., Zhu, H., and Zhou, Z. (2014). Mobile agent-based energy-aware and user-centric data collection in wireless sensor networks. *Computer networks*, 74:58–70.

Gavalas, D., Venetis, I. E., Konstantopoulos, C., and Pantziou, G. (2017). Mobile agent itinerary planning for WSN data fusion: considering multiple sinks and heterogeneous networks. *International Journal of Communication Systems*, 30(8):e3184.

Jin, Y., Gormus, S., Kulkarni, P., and Sooriyabandara, M. (2016). Content centric routing in IoT networks and its integration in RPL. *Computer Communications*, 89:87–104.

Li, S., Xu, J., Van Der Schaar, M., and Li, W. (2016). Popularity-driven content caching. In *Computer Communications, IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on*, pages 1–9. IEEE.

Liu, B., Cao, J., Yin, J., Yu, W., Liu, B., and Fu, X. (2016). Disjoint multi mobile agent itinerary planning for big data analytics. *EURASIP Journal on Wireless Communications and Networking*, 2016(1):99.

Lu, J., Feng, L., Yang, J., Hassan, M. M., Alelaiwi, A., and Humar, I. (2019). Artificial agent: The fusion of artificial intelligence and a mobile agent for energy-efficient traffic control in wireless sensor networks. *Future Generation Computer Systems*, 95:45–51.

Mekki, K., Derigent, W., Zouinkhi, A., Rondeau, E., Thomas, A., and Abdelkrim, M. N. (2017). Rawpg: A data retrieval protocol in micro-sensor networks based on random walk and pull gossip for communicating materials. *IEEE Internet of Things Journal*, 4(2):414–426.

Mišić, J. and Mišić, V. B. (2018). Proxy cache maintenance using multicasting in coap iot domains. *IEEE Internet of Things Journal*, 5(3):1967–1976.

Rahmani, A.-M., Thanigaivelan, N. K., Gia, T. N., Granados, J., Negash, B., Liljeberg, P., and Tenhunen, H. (2015). Smart e-health gateway: Bringing intelligence to internet-of-things based ubiquitous healthcare systems. In *Consumer Communications and Networking Conference (CCNC), 2015 12th Annual IEEE*, pages 826–834. IEEE.

Xu, G., Ngai, E. C.-H., and Liu, J. (2018). Ubiquitous transmission of multimedia sensor data in internet of things. *IEEE Internet of Things Journal*, 5(1):403–414.

Xu, X., Li, X.-Y., Wan, P.-J., and Tang, S. (2012). Efficient scheduling for periodic aggregation queries in multihop sensor networks. *IEEE/ACM Transactions on Networking (TON)*, 20(3):690–698.

Yang, S., Adeel, U., Tahir, Y., and McCann, J. A. (2017). Practical opportunistic data collection in wireless sensor networks with mobile sinks. *IEEE Transactions on Mobile Computing*, 16(5):1420–1433.

Zhan, Y., Xia, Y., Zhang, J., and Wang, Y. (2018). Incentive mechanism design in mobile opportunistic data collection with time sensitivity. *IEEE Internet of Things Journal*, 5(1):246–256.

Zhang, Z., Ma, H., and Liu, L. (2015). Cache-aware named-data forwarding in internet of things. In *Global Communications Conference (GLOBECOM), 2015 IEEE*, pages 1–6. IEEE.