Armazenamento Distribuído para Redes de Dispositivos Móveis

Ricardo Monteiro João A. Silva João M. Lourenço Hervé Paulino*
{rad.monteiro, jaa.silva}@campus.fct.unl.pt
{joao.lourenco, herve.paulino}@fct.unl.pt

NOVA Laboratory for Computer Science and Informatics
Departamento de Informática, Faculdade de Ciências e Tecnologia
Universidade NOVA de Lisboa, 2829-516 Caparica, Portugal

Resumo Os dispositivos móveis em proximidade geográfica representam um conjunto de recursos inexplorados, tanto em termos de capacidade de processamento como de armazenamento, o que abre caminho para novas aplicações com oportunidades e desafios únicos. Os sistemas atuais de partilha de dados (e. g., fotos, músicas, vídeos) para dispositivos móveis exigem que exista conectividade com a Internet para funcionarem. No entanto, em ambientes onde a conectividade com a Internet não é constante ou de boa qualidade (e. g., eventos desportivos e concertos), ou em locais remotos onde as infraestruturas de rede não existem, é difícil (ou mesmo impossível) partilhar dados entre vários dispositivos móveis. Para resolver este problema, os dispositivos móveis podem formar uma rede ad hoc para compartilhar os seus dados e recursos. Neste artigo propomos um sistema de armazenamento distribuído para partilha de dados entre dispositivos móveis de uso diário, e. g., *smartphones* e *tablets*, usando um mecanismo de *melhor esforço* para garantir persistência e disponibilidade de dados suportando *churn* (entrada e saída inesperada de dispositivos).

Palavras-Chave: Armazenamento Distribuído, Dispositivos Móveis, Redes ad hoc, Peer-to-Peer

1 Introdução

O poder de computação e de armazenamento dos *smartphones* atuais rivaliza com o dos *desktops* de há poucos anos atrás. Tal evolução permite executar num *smartphone* corrente tarefas que anteriormente tinham de ser delegadas em sistemas de maiores dimensões. Adicionalmente, hoje em dia, estes pequenos dispositivos oferecem múltiplas funcionalidades—como tirar fotografias, gravar

* Este trabalho foi parcialmente financiado pela Fundação para a Ciência e Tecnologia (FCT-MEC), no contexto do projeto de investigação CMUP-ERI/FIA/0048/2013, do plano estratégico PEst-UID/CEC/04516/2013 e bolsa de investigação SFRH/BD/99486/2014.

vídeos ou obter sinais GPS—que antes só eram possíveis com dispositivos especializados de difícil acesso para muitos dos consumidores. Dada a proliferação e o aumento das capacidades dos dispositivos móveis, é realista considerar as oportunidades que um conjunto de dispositivos móveis pode colaborativamente apresentar em termos de recursos computacionais e de armazenamento de dados.

Recentemente, tem havido interesse na utilização de uma nuvem de *smartphones* para formar uma rede de armazenamento para aplicações avançadas [7]—por exemplo, uma coleção de oito *smartphones*, cada um com 16 GB de armazenamento, pode representar coletivamente uma nuvem com uma generosa capacidade de 128 GB. Este conceito pode ser particularmente interessante em determinadas situações. Por exemplo, em locais onde não existem infraestruturas de rede disponíveis—localizações remotas, cenários de catástrofe natural, etc. Também em locais com um grande número de pessoas, as infraestruturas existentes podem não ter capacidade para suportar o elevado número de conexões, e. g., eventos desportivos ou concertos. Em eventos particulares, como festas de aniversário, reuniões ou pequenas conferências, também pode não compensar montar uma infraestrutura de rede espeficicamente para esses eventos, devido aos custos elevados ou a configurações demoradas.

Drolia et al. [9] comparam o processamento de dados numa nuvem de *smartphones* contra o envio desses mesmos dados para serem processados numa infraestrutura remota, no que se refere à duração em tempo, ao consumo de bateria e à quantidade de dados transmitidos. Os seus resultados mostram que para algumas classes de aplicações, o processamento na nuvem de *smartphones* é mais eficiente em termos de bateria consumida e latência na obtenção dos resultados.

Armazenamento distribuído em redes móveis ad hoc é um tema com alguma investigação [10,11], mas a maioria dos sistemas existentes não tem uma implementação concreta e funcional para dispositivos móveis ou, no caso específico do iTrust [11], não garante persistência dos dados (i. e., quando um dispositivo se desconecta, os conteúdos partilhados por esse dispositivo ficam inacessíveis).

Neste artigo apresentamos as características e arquitectura de um sistema de armazenamento e partilha de conteúdos distribuído para dispositivos móveis. O sistema permite a publicação de conteúdos privados e a obtenção de conteúdos partilhados por outros utilizadores conetados à mesma rede. Adicionalmente, este tenta garantir a persistência e disponibilidade dos dados na presença de *churn* (entrada e saída inesperada de dispositivos na rede). No entanto, dada a natureza volátil e dinâmica dos ambientes móveis não é possível fornecer uma garantia absoluta.

Desenvolvemos um protótipo para Android onde o utilizador pode partilhar as suas fotografias, visualizar os nomes e *thumbnails* das fotografias publicadas por outros utilizadores e obter essas mesmas fotografias. Este protótipo pode ser facilmente alterado para partilha de vários tipos de conteúdos diferentes (por exemplo, vídeos, músicas, etc.) e funciona em dispositivos móveis existentes, sem que se seja necessário realizar qualquer tipo de alteração ao dispositivo, como a obtenção de permissões de super-utilizador (*root*). A limitação atual passa

pela necessidade de existir uma rede sem fios disponível para os dispositivos se conectarem. Esta não precisa estar conectada à Internet e pode ser fornecida por um dispositivo móvel ativando a funcionalidade *hotspot*.

O resto do artigo segue a seguinte estrutura. Na Secção 2 é apresentado algum contexto, os desafios que se colocam e como foram ultrapassados. De seguida, na Secção 3, apresentada-se a arquitetura do sistema com as suas várias camadas. Na Secção 4 reporta-se e analisa-se resultados experimentais. Na Secção 5 discute-se algum trabalho relacionado. Por último, na Secção 6 apresenta-se as conclusões finais e algum trabalho futuro.

2 Armazenamento de Dados para Dispositivos Móveis

Um sistema de armazenamento para redes ad hoc de dispositivos móveis tem de lidar com diversos desafios inerentes a este tipo de ambiente volátil e dinâmico. A descoberta/localização de recursos neste tipo de ambientes assemelha-se em muito às funcionalidades oferecidas pelos sistemas *peer-to-peer* (P2P) construídos para a Internet. Para além disso, existem os desafios inerentes a ambientes móveis, como a alta mobilidade dos nós (causando mudanças na topologia da rede), as entradas e saídas inesperadas de nós na rede (*churn*), e a comunicação volátil e de qualidade irregular. Tendo em conta os dispositivos móveis, também é necessário lidar com as suas propriedades e limitações características, como a bateria e largura de banda limitadas.

2.1 Localização de Recursos

A localização de recursos é um componente bastante importante em redes ad hoc para dispositivos móveis. Nestas redes a localização de recursos deve ser rápida, de maneira a que as aplicações respondam em tempo útil (e de maneira interativa) aos seus utilizadores, mas também deve ser eficiente, de maneira a haver a menor quantidade de comunicação possível, ajudando na poupança de bateria.

Um mecanismo muito simples para localização de recursos numa rede passa por inundar (*flooding*) a rede com mensagens, perguntando a todos os nós quem é que tem aquilo que se procura. São enviadas mensagens para diversos nós, escolhidos aleatoriamente, até que os recursos procurados sejam encontrados. Este tipo de mecanismo é utilizado em redes não estruturadas, onde os nós não têm conhecimento da localização dos recursos disponíveis. São, também, conhecidos por não serem muito eficientes em redes de grandes dimensões, por isso, é que normalmente são implementados outros mecanismos para prevenir o envio de mensagens em excesso, o que poderia causar a rutura da rede. Um exemplo de um sistema com uma rede não estruturada é o sistema *Gnutella* [16]. O sistema iTrust [11] também usa uma rede não estruturada e um mecanismo de *flooding* para localização de conteúdos.

Pelo contrário, nos sistemas com redes estruturadas, os nós têm uma visão parcial da rede, conhecendo alguns dos seus vizinhos. O tipo de rede estruturada

mais conhecida são as tabelas de dispersão distribuídas (TDD). Estas resolvem o problema do *flooding*, porque empregam um esquema de procura organizada, onde cada nó é responsável por parte dos recursos existentes no sistema. Desta maneira, os nós são capazes de direcionar os pedidos para os nós responsáveis muito mais rápida e facilmente. Um exemplo de uma TDD é o *Chord* [15].

Comparando os apectos positivos e negativos de cada abordagem, decidimos optar pela utilização de uma TDD. Apesar de ter um custo adicional associado à manutenção da rede (o envio de *pings*), acreditamos que num ambiente móvel, onde temos energia e recursos limitados, o custo da manutenção da rede será menor que o custo para garantir persistência de dados numa rede não estruturada. Para garantir persistência de dados numa rede não estruturada, seria necessário a constante verificação dos conteúdos existentes no sistema e da quantidade de réplicas existentes para cada conteúdo. O Phoenix [13] é um exemplo de um sistema com um protocolo de manutenção de réplicas para garantir persistência de dados numa rede não estruturada.

Para além da manutenção das réplicas, com uma TDD resolvemos diversos problemas existentes para atingir o objectivo que pretendemos. A TDD permite-nos determinar e distribuir a responsabilidade dos diversos conteúdos pelos nós existentes no sistema. Desta forma, cada nó é responsável por uma parte dos conteúdos existentes, garantindo que a rede contém, pelo menos, um número mínimo de réplicas para cada conteúdo distribuídas por diferentes nós. Para além disso, a localização e obtenção de conteúdos no sistema por parte de um nó não requer a inundação de pedidos na rede, reduzindo também o consumo de recursos por cada pedido.

2.2 Comunicação

Existem alguns projectos que tentam resolver o problema da criação e operacionalidade de redes ad hoc WiFi compostas por dispositivos móveis (por exemplo: *Serval Project* [2], e *SmartPhone Ad-hoc Networking (SPAN)* [18]). No entanto, ambos os projetos, requerem alterações e configurações nos dispositivos. Para realizar estas modificações é necessário ter permissões de super-utilizador (*root*). Além disso, a gama de modelos suportados por estes projetos é bastante pequena.

Para contornar este problema e criar uma rede ad hoc de múltiplos dispositivos, podemos fazer uso do WiFi Direct [3]. Casetti et al. [5], Lombera et al. [11], e Teófilo et al. [17] usam esta estratégia para os dispositivos comunicarem em modo ad hoc. Outro modo para formar redes ad hoc entre dispositivos móveis é através do Bluetooth [1] (utilizado, por exemplo, em [20]).

Atualmente, o nosso protótipo requer um ponto de acesso (por exemplo: *router*, ou um dispositivo com *hotspot*) e ainda não abordámos a comunicação ad hoc. Como trabalho futuro, o nosso objetivo é estudar e implementar um mecanismo de comunicação ad hoc através de WiFi Direct semelhante ao utilizado por Teófilo et al. [17].

WiFi Direct. O WiFi Direct é um protocolo padronizado pela WiFi Alliance, com o objectivo de permitir comunicações de dispositivo para dispositivo, sem a

necessidade de um ponto de acesso intermédio. A comunicação entre dispositivos usando o WiFi Direct ocorre dentro de um único *grupo*. Um dispositivo do grupo actua como líder (*Group Owner (GO)*) e os restastes dispositivos, designados de *clientes*, são emparelhados ao GO. Os clientes podem ser de dois tipos: *clientes P2P*, têm suporte ao protocolo WiFi Direct; ou, *clientes WiFi*, não têm suporte ao protocolo WiFi Direct e vêem o GO como se fosse um ponto de acesso sem fios convencional.

Os papeis de cada dispositivo dentro do grupo mantêm-se durante toda a sessão. Se o GO se desconectar, os restantes dispositivos têm de criar um novo grupo e eleger um novo GO. Para criar um novo grupo e ser o GO do mesmo, os dispositivos têm que suportar a tecnologia WiFi Direct. Caso contrário, apenas é possível conectarem-se a grupos já formados, vendo o GO como um ponto de acesso. Também é possível a um dispositivo pertencer a mais que um grupo, formando uma ponte entre os dois grupos. Para isso, o dispositivo tem de suportar dois endereços MAC diferentes e existem as seguintes limitações nos dispositivos Android actualmente:

1. Um dispositivo ser GO de mais que um grupo.
2. Um dispositivo ser GO de um grupo e cliente P2P de outro,
3. Um dispositivo ser cliente de mais que um grupo.

Resumindo, o GO de um grupo tem de se conectar como cliente WiFi a outro grupo para formar uma ponte entre os dois grupos. Desta forma, é possível conectar múltiplos grupos e comunicar entre dispositivos de diferentes grupos.

Casetti et al. [5] descrevem esta técnica para dispositivos Android de origem, isto é, sem modificar o hardware ou software original, nem desbloquear o dispositivo para ter permissões de super-utilizador (*root*). Teófilo et al. [17] estenderam esse trabalho permitindo transferências de dados com maiores velocidades entre grupos, necessidade de menor número de reenvios (hops) para transferir mensagens entre grupos, menor número de dispositivos necessários por área geográfica, e uso de recursos mais equilibrado.

Bluetooth. O Bluetooth é uma tecnologia sem fios padronizada pelo *Bluetooth Special Interest Group (SIG)* para troca de dados entre dispositivos a curta distancia. Uma rede formada por dois dispositivos conectados por Bluetooth é designada de *piconet*.

As *piconets* são compostas por um líder (*master*), um máximo de sete dispositivos activos (*slaves*), e até 255 dispositivos inactivos. O master coordena as comunicações com os slaves. Os slaves não podem comunicar entre eles, apenas comunicam com o master. Os inactivos não podem comunicar até que mudem para um estado activo (master ou slave). O papel de cada dispositivo na piconet pode mudar, no entanto nunca pode existir mais que um master e sete slaves.

Usando Bluetooth é possível formar uma rede com múltiplos grupos, designando-se de *scatternet* (e.g., [20]), onde um master ou slave pode estar conectado a uma segunda piconet. Neste caso, o nó alterna a comunicação entre as duas piconets por intervalos de tempo.

2.3 Persistência dos Dados

Os ambientes móveis, como os que estamos a considerar, são altamente dinâmicos devido à mobilidade dos nós. Tanto a mobilidade dos nós, como as entradas e saídas de nós na rede, causam diversas alterações na topologia da rede.

Um sistema de armazenamento distribuído para redes móveis ad hoc tem de estar ciente da alta probabilidade dos nós abandonarem os sistema inesperadamente, causando perda de conteúdos. A solução para diminuir a perda de conteúdos devido às saídas de nós é criar múltiplas réplicas de cada conteúdo. Esta estratégia não é infalível, pois no limite, todas as localizações das réplicas de determinado conteúdo podem desconectar-se da rede ao mesmo tempo, levando à perda permanente desse conteúdo.

Para além de aumentar a garantia da persistência de dados, a replicação também pode aumentar o desempenho em sistemas distribuídos especialmente em redes móveis ad hoc. Existindo múltiplas réplicas de cada conteúdo, o nó que faz o pedido pode aceder à replica mais próxima da sua localização, reduzindo a latência para obtenção dos dados.

Métodos de Replicação. Métodos e estratégias de replicação para redes móveis ad hoc são um tema com uma grande quantidade de investigação. Existem diversas publicações com estratégias de replicação bastante diferentes umas das outras que têm como objectivo resolver problemas diferentes. Dado esta vasta gama de métodos de replicação, Derhab et al. [8] propõem uma classificação para os diferentes tipos de métodos de replicação. De acordo com os autores, existem dois tipos de métodos de replicação: protocolos adaptados a ambientes móveis, e protocolos não adaptados.

Os protocolos adaptados a ambientes móveis tentam resolver pelo menos um dos seguintes três desafios: partições da rede, escalabilidade, ou consumo de energia.

1. Protocolos conscientes da energia (e.g., [14]): Estes métodos detectam quando o nó está a ficar com pouca bateria e replica os dados antes que a bateria termine, tendo como objectivo reduzir o consumo de energia.
   A distribuição e o balanceamento da carga de trabalho dos nós é uma forma de reduzir o consumo de energia aos nós que contêm os dados mais populares.
2. Protocolos conscientes das partições da rede (e.g., [6]): Tentam prever se algum nó irá se desconectar da rede e replicar o seu conteúdo previamente, ou ou se poderá ocorrer alguma partição na rede e replicar os dados que estão unicamente disponíveis numa das possíveis futuras partições para a outra partição.
3. Protocolos escaláveis (e.g., [19]): Um protocolo escalável não tem um grande *overhead* conforme aumenta o número de nós na rede. Estes protocolos tendem a diminuir a quantidade de comunicações existentes entre os nós.

Os métodos de replicação podem tentar resolver mais que um destes três problemas, e o perfeito seria resolver os três problemas. Mas para o nosso

conhecimento e de acordo com Derhab et al., não existe ainda um protocolo que faça isso nem que seja simultaneamente escalável e consciente das partições.

Para além destas três características, um protocolo de replicação pode ser proactivo ou reactivo, dependendo se os dados são replicados periodicamente ou se o envio de réplicas é desencadeado devido a mudanças na topologia da rede.

Atualmente, ainda não abordámos protocolos adaptados a ambientes móveis. Temos um protocolo adaptativo à popularidade dos conteúdos, onde os conteúdos mais populares (pedidos mais vezes) são replicados por um maior número de nós de maneira a distribuir melhor a carga e tentar poupar a bateria dos dispositivos que replicam os dados mais populares.

2.4 Recursos Limitados

Apesar das múltiplas funcionalidades que os dispositivos móveis apresentam hoje em dia, outras continuam a ser limitativas. Os *smartphones* de hoje em dia têm processadores, memória principal e espaço em disco que rivalizam com os computadores pessoais de poucos anos atrás, mas a bateria continua a ser um grande entrave ao que se consegue fazer num dispositivo deste tamanho. Assim, o uso dessas capacidades deve ser cuidadosamente gerido de maneira a não esgotar a bateria rapidamente.

Pela simples razão de que a bateria é o recurso mais limitativo dos dispositivos móveis de hoje em dia, optámos por implementar um mecanismo com o objetivo de poupar energia quando o dispositivo se encontra num estado de bateria fraca. Este mecanismo será descrito em detalhe mais abaixo.

3 Uma Abordagem Prática

Nesta secção, apresentamos o desenho da nossa solução para um sistema de armazenamento e partilha de dados distribuído e totalmente descentralizado para redes de dispositivos móveis.

O nosso sistema foi desenvolvido como uma biblioteca e segue o modelo clássico de armazenamento com pares chave/valor, onde cada chave serve como identificador e é mapeada para um valor imutável. Atualmente, o sistema suporta as seguintes operações:

inserir$\langle chave, valor \rangle$ Armazena o par chave/valor dado no sistema, disponibilizando-o aos outros nós;

obter$\langle chave \rangle$ Procura no sistema pela chave especificada, retornando o valor associado caso exista;

remover$\langle chave \rangle$ Remove o conteúdo mapeado pela chave especificada, caso o conteúdo tenha sido colocado por este nó; e

pesquisar$\langle filtro \rangle$ Retorna todas as chaves que estão em conformidade com o filtro fornecido.

Uma vez que estamos a considerar um ambiente com dispositivos móveis, o sistema foi desenvolvido tendo em conta algumas propriedades desses ambientes, nomeadamente (i) a persistência dos dados; e (ii) a consciência de energia.

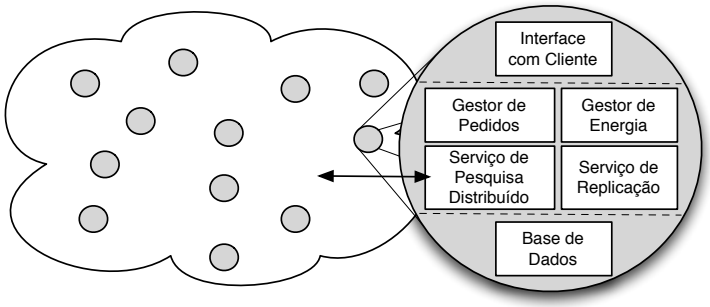

Figura 1. Arquitectura do sistema.

Para garantir a persistência dos dados no sistema é utilizada uma abordagem de *melhor esforço*. Dada a natureza volátil do ambiente que estamos a considerar, os efeitos do *churn* dificultam que o sistema garanta persistência "absoluta" dos dados, ou seja, estes permanecerão no sistema desde que todos os nós que armazenam um item não se desconectem do sistema no mesmo exato momento.

Uma vez que estamos a considerar dispositivos móveis, o sistema tem de empregar mecanismos de poupança de energia, caso contrário o seu uso torna-se pouco convidativo (ou até impeditivo). O nosso sistema tem consciência da energia do dispositivo, proporcionando um mecanismo para economizar energia (i. e., um modo de economia de bateria) quando o nível de energia de um dispositivo está abaixo de um limiar pré-definido. Ao entrar no modo de economia de bateria, o cliente ainda pode publicar e obter conteúdos, mas deixa de receber pedidos de outros nós. Mais informação sobre esta propriedade será dada mais abaixo.

3.1 Arquitetura do Sistema

Com o objetivo de implementar um sistema justo (em termos de recursos consumidos), optámos por uma arquitetura completamente descentralizada e simétrica, ou seja, todos os nós têm o mesmo papel no sistema e executam os mesmos serviços. Um nó é composto por vários componentes, apresentados na Figura 1, seguindo um modelo de três camadas e separando de forma clara as camadas de apresentação, lógica, e dados.

Interface com o Cliente. Fornece a interface para as aplicações cliente, fornecendo as operações disponíveis (`inserir`, `obter`, `remover`, e `pesquisar`) como uma biblioteca.

Gestor de Pedidos. O Gestor de Pedidos (GP) coordena o processamento dos pedidos enviados pelas operações realizadas na interface. As operações `obter` são primeiramente direcionadas para a camada de dados local (Base de Dados), para verificar se podem ser servidas localmente. Se necessário, só de seguida, é que são

transmitidas para o Serviço de Pesquisa Distribuído (SPD) para ser realizada uma pesquisa em todo o sistema. As operações `inserir` são tratadas pelo Serviço de Replicação (SR) que trata de gerir as réplicas necessárias nos nós remotos. As operações `pesquisar` são retransmitidas para a camada de dados local e para o SPD, para uma cobertura completa do espaço de pesquisa.

Serviço de Pesquisa Distribuído. A função do SPD é determinar a localização dos itens publicados por nós remotos, e executar todas as tarefas de manutenção relacionadas com a rede, tais como a gestão dos nós vizinhos. A implementação do nosso protótipo atual recorre ao TomP2P [4], uma Tabela de Dispersão Distribuída implementada em Java e que funciona em Android. A fim de tornar o TomP2P adequado a dispositivos móveis e redes móveis, foram realizadas várias otimizações, das quais destacamos a redução da troca de mensagens em vários casos: (i) o mecanismo de replicação original transmitia o conteúdo a ser replicado de forma pro-ativa infinitamente para todos os nós encarregues da replicação. Atualmente, é aplicada uma estratégia reativa, propagando unicamente os conteúdos a replicar quando são detetadas alterações no grupos de nós que replicam os dados; e (ii) as operações de pesquisa e de obtenção de dados, que antes faziam *flooding* na rede para determinar os nós que guardam os itens, utilizam agora uma estratégia de potência de duas escolhas [12], que só requer dois nós selecionados aleatoriamente. Caso estes dois nós não tenham o conteúdo desejado, serão selecionados outros dois nós e assim sucessivamente até o conteúdo ser encontrado. Esta abordagem é susceptível de aumentar a latência das operações, mas reduz o número de mensagens trocadas e, por conseguinte, a energia consumida.

Serviço de Replicação. A replicação de dados é uma técnica conhecida para tentar garantir a persistência de dados em sistemas distribuídos. Apropriadamente, o SR é responsável pela manutenção do número de réplicas para cada item publicado de acordo com uma estratégia de replicação predefinida. O sistema cresce a partir do TomP2P e é completamente descentralizado, distribuindo a responsabilidade de garantir o nível de replicação de dados de cada item por vários nós. Assim, para cada item, há um nó *responsável* por gerir o número designado de réplicas, detetando eventuais desconexões de nós que contenham réplicas. Para cada desconexão, o nó responsável seleciona um novo nó e envia uma réplica para esse mesmo nó. A desconexão de um nó responsável desencadeia a eleição de um novo nó responsável.

Atualmente, o sistema suporta três estratégias de replicação diferentes: (i) *replicação estática*; e (ii) *replicação com reconhecimento da rede* estão incluídas no TomP2P; (iii) *replicação por popularidade* regista o número de pedidos de cada item e aumenta o número de réplicas para os itens com mais acessos. Esperamos que este mecanismo reduza a latência na obtenção de um item popular e ao mesmo tempo reduza o consumo de energia nos nós que armazenam estas réplicas, simplesmente porque os dados populares serão difundidos entre um maior número

de nós, conseguindo assim um melhor balanceamento de carga ao solicitar esses itens.

Gestor de Energia. Quando o nível da bateria fica abaixo de um limiar pré-definido o nó é parcialmente desativado. O Gestor de Energia (GE) desliga as tarefas de manutenção relacionadas com a rede no SPD, fazendo com que este nó fique *invisível* para os restantes nós do sistema. Além disso, o GE desliga o SR para terminar as tarefas de replicação. Esta abordagem reduz o consumo de energia do nó, e o sistema encontra novas localizações para guardar réplicas existentes no nó parcialmente desativado. Se o nó for recarregado e o nível da bateria ultrapassar o limite, o nó volta a juntar-se totalmente ao sistema e continua a funcionar como antes.

Base de Dados. Uma base de dados do tipo chave/valor para mapear as chaves para os conteúdos partilhados e obtidos.

4 Resultados Experimentais

Nesta secção mostramos e analizamos alguns resultados experimentais do nosso protótipo. O nosso sistema foi desenhado como uma biblioteca, e com essa biblioteca desenvolvemos uma aplicação Android para partilha de fotografias. Assim, temos dois níveis de avaliação: (i) avaliação do protótipo Android, que permite aferir acerca da usabilidade do nosso sistema ao nível do utilizador; e (ii) avaliação da biblioteca num computador, que permite uma avaliação com um número maior de nós.

4.1 Protótipo em Android

Desenvolvemos uma aplicação móvel em Android para a gestão de uma galeria de fotografias. A aplicação oferece várias *vistas*, nomeadamente para a visualização de conteúdos privados (locais), de conteúdos partilhados pelo próprio utilizador, de conteúdos que foram obtidos de dispositivos remotos e uma lista de nomes e *thumbnails* dos conteúdos disponíveis no sistema. Para os conteúdos privados, o utilizador tem a possibilidade de escolher e partilhar o que desejar. A partilha de uma fotografia faz o sistema gerar um identificador exclusivo que é armazenado na BD e publicado no sistema. Os conteúdos partilhados podem ser removidos do sistema apenas pelo utilizador que partilhou o conteúdo. No entanto, caso o conteúdo tenha sido obtido por outro dispositivo não é possível removê-lo do dispositivo remoto, ficando apenas indisponível para futuras obtenções.

Para avaliar o impacto da nossa implementação no consumo de bateria de um *smartphone*, realizámos dois testes preliminares numa rede composta por dois tipos de dispositivos: *tablets* Nexus 7 de 2013 com Android 5.1 e *smartphones* Motorola Moto G de 2ª geração com Android 5.0.2. O primeiro teste teve como objetivo determinar o consumo de bateria quando nenhuma operação é realizada

pelo sistema, para além da manutenção da rede pelo SPD. Primeiro, quisemos determinar o consumo de bateria para a criação de uma rede através do modo de *ponto de acesso* de um dispositivo. Para isso, criámos uma rede onde o Nexus 7 tinha o papel de *ponto de acesso* e os Motos G juntaram-se à rede. Numa execução de 8 horas, o consumo de bateria foi de 24% e 6% para o dispositivo que criou a rede (ou seja, o ponto de acesso) e os dispositivos que se juntaram, respetivamente.

Numa execução com a mesma duração (8 horas), mas agora com a nossa aplicação a correr, observámos um aumento no consumo de bateria de 11% e 2%, em relação à execução anterior, para o dispositivo que criou a rede e os dispositivos que se juntaram, respetivamente.

Estes valores foram obtidos através do cálculo da média de duas observações, onde se verificaram variações de apenas 2% e 4%, para o Moto G e Nexus 7, respectivamente. Não foram realizados testes em que o ponto de acesso é oferecido pelo Moto G por falta de suporte de tal funcionalidade nesses dispositivos. No entanto, não é necessário que seja um dispositivo a fornecer a rede, podendo esta ser formada a partir de um *router* disponível na área. Dessa forma, evita-se gastos de bateria na criação do *ponto de acesso* por parte de um dispositivo.

Num segundo teste, pretendemos determinar se o envio de pedidos e obtenção de conteúdos consumia muita bateria. Neste teste, o Nexus 7 estava continuamente a fazer pedidos e a obter (e apagar, devido a limitações de armazenamento no dispositivo) uma imagem de 2 MB durante uma hora e o Moto G a responder aos pedidos, enviando a imagem. No final, o Nexus 7 obteve e armazenou em disco a imagem 4962 vezes usando 10% de bateria. De notar que também estava a servir de ponto de acesso. Por sua vez, o nó que serviu os pedidos (Moto G) teve um consumo de bateria de 14%.

4.2 Avaliação

Em seguida apresentamos alguns testes que demonstram os resultados das otimizações que realizámos ao TomP2P. Os testes foram executados numa máquina, onde foram criadas várias instâncias de nós (um processo para cada nó). A máquina onde os testes foram executados tem um processador Intel Core i5-2430M 2.4GHz, 4 GB de memória RAM e corre o sistema operativo Linux, distribuição Ubuntu 14.04.1.

Se nada for dito em contrário, todos os testes foram executados com as seguintes configurações fixas: (i) o tempo de execução dos testes foi de 20 minutos; (ii) o período de intervalo de execução das tarefas de manutenção da replicação foi de 60 segundos; e (iii) o número mínimo de réplicas que o sistema manteve para cada item foi de 5 réplicas.

Otimização do mecanismo de replicação. Um bom mecanismo de replicação que não cause um *overhead* exagerado, mas que consiga manter os dados no sistema é importante para qualquer tipo de sistema distribuído que necessite de manter os dados disponíveis. Dada as características dos ambientes móveis, este componente é ainda mais importante.

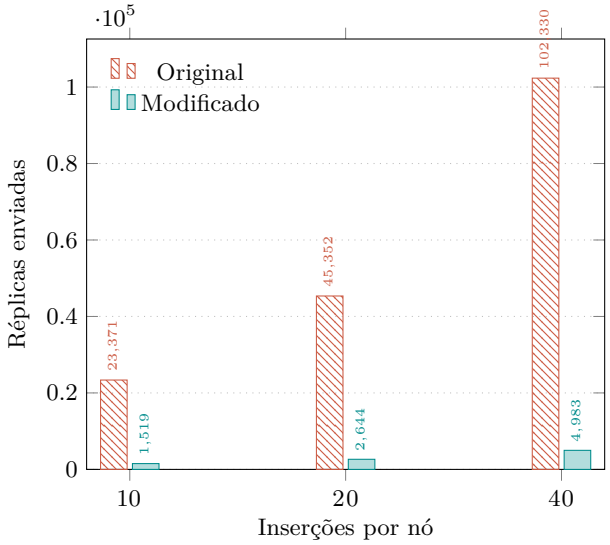

Figura 2. Número de réplicas enviadas por inserções.

Para adaptar o mecanismo de replicação existente no TomP2P, fizemos algumas modificações de forma a reduzir o número de réplicas transmitidas. O protocolo original tem um mecanismo proativo, ou seja, periodicamente envia as réplicas para os seus destinos de replicação mesmo que o destinatário já tenha os dados. Esta característica poderá não ser um grande problema em redes convencionais (como a Internet), dependendo da quantidade e tamanho dos conteúdos, velocidade da rede, número de nós, etc. No entanto o mesmo não se verifica em rede móveis, devido à largura de banda e bateria limitadas.

Dado este problema, alterámos o protocolo para um mecanismo reativo, onde as réplicas são enviadas apenas quando (e se necessário) há alterações na rede.

Nos gráficos das Figuras 2 e 3 podemos observar a quantidade de réplicas enviadas pelo protocolo original (*Original*) e pela nossa modificação (*Modificado*). Para os resultados do gráfico da Figura 2 foram utilizados 20 nós. Para os resultados do gráfico da Figura 3 foram feitas 10 inserções de conteúdos por cada nó.

Nestes gráficos podemos verificar que conseguimos reduzir bastante as réplicas transmitidas. Neste caso, o sistema esteve em execução apenas durante 20 minutos, numa execução com um período de tempo mais alargado estas diferenças seriam ainda mais acentuadas.

Número de réplicas transmitidas com *churn*. O *churn* é um problema com um impacto maior em ambientes móveis. Dado este problema, queremos avaliar o sistema quando está exposto a níveis de *churn* diferentes.

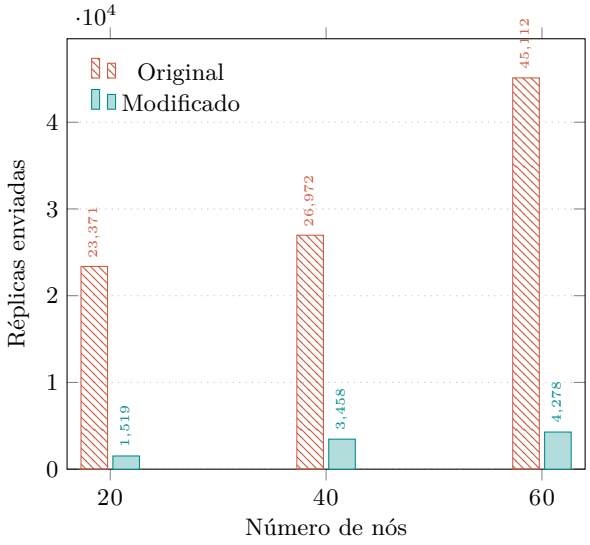

Figura 3. Número de réplicas enviadas por número de peers.

No gráfico da Figura 4 podemos visualizar os resultados de uma execução com 20 nós, onde cada um fez 10 inserções no sistema. Os valores de *churn* no gráfico denotam o número de nós que se desconectaram do sistema em intervalos de 60 segundos. Após a desconexão dos nós, novos nós (em igual número) foram conectados ao sistema.

Como era esperado, os resultados do protocolo de replicação original são praticamente iguais aos resultados sem *churn*, pois este protocolo funciona de forma proativa, enviando réplicas periodicamente independentemente se há ou não alterações na rede.

No protocolo de replicação modificado, podemos verificar que o número de réplicas aumenta bastante. No entanto, como estamos a desconectar 25% e 50% dos nós existentes (5 e 10 nós, respetivamente) é necessário repor uma grande quantidade de réplicas nos novos nós. Tendo em conta que são 200 conteúdos e 15 nós (depois da desconexão de 5 nós) para enviar réplicas aos novos 5 nós, por minuto há uma média de 230 réplicas enviadas, o que dá cerca de 15 réplicas enviadas por nó por minuto. Nos resultados obtidos na avaliação do protótipo em Android, onde um dispositivo fez quase 5000 pedidos, guardou e apagou o ficheiro do disco, obtendo uma média de 83 ficheiros pedidos por minuto. Cruzando os dois resultados, podemos concluir que o envio de 15 réplicas por minuto, por nó, não é muito significativo.

5 Trabalho Relacionado

Nesta secção apresentamos projectos relacionados com armazenamento distribuído em redes ad hoc, e com comunicações ad hoc para dispositivos móveis.

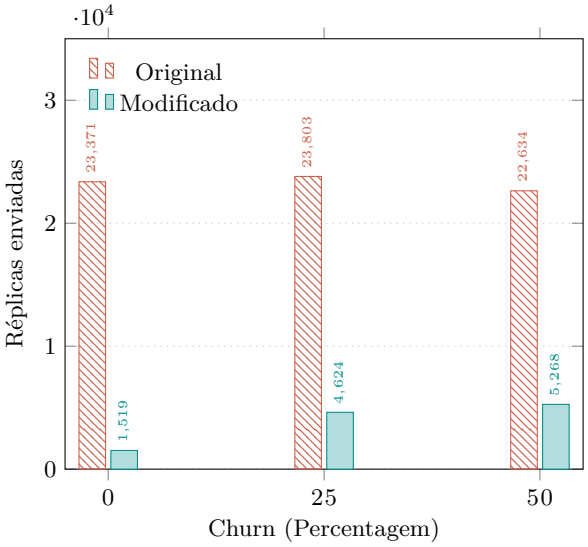

Figura 4. Número de réplicas enviadas consoante o nível de *churn*.

Phoenix [13] é o sistema que, para nosso conhecimento, tem os objectivos mais semelhantes ao que pretendemos: garantir persistência e disponibilidade de dados. Os autores propõem um protocolo baseado em rondas para a manutenção das réplicas no sistema através de transmissões *broadcast* e *single-hop*. Em cada ronda, é elegido um bloco (os conteúdos são partidos em diversos blocos) para verificação do número de réplicas existentes e, se necessário, enviar o bloco a nós que não o tenham para repor o número mínimo de réplicas existentes. No entanto, não é proposto nenhum mecanismo de pesquisa e obtenção de dados existentes no sistema.

O projecto iTrust [11] permite publicar, pesquisar e obter conteúdos através de uma rede móvel ad hoc constituída por dispositivos Android com WiFi Direct. Neste sistema o dispositivo que quer partilhar conteúdos envia várias cópias de meta-dados para múltiplos dispositivos remotos aleatoriamente. Estes meta-dados contêm a localização do conteúdo. Para obter conteúdos, o dispositivo envia pedidos para dispositivos remotos aleatoriamente até encontrar algum que contenha os meta-dados. De seguida, faz o pedido ao dispositivo que contem o conteúdo. Uma limitação da abordagem utilizada neste sistema é a impossibilidade de garantir persistência de dados. Caso o dispositivo que partilha determinado conteúdo se desconecte da rede, o conteúdo perde-se.

Luo et al. [10] propõem um sistema de armazenamento baseado em quóruns probabilísticos, permitindo acessos de leitura e escrita concorrentes. Os pedidos de leitura e escrita são difundidos até atingirem o número de nós necessários para ter o quórum de leitura ou escrita, respectivamente. Esta característica causa a necessidade de configuração para diferentes ambientes (número de nós, área, taxa de leituras e escritas, etc.). Este sistema não é totalmente descentralizado

e simétrico, ou seja, a tarefa de armazenamento de dados não é partilhada por todos os nós, havendo a eleição de alguns servidores (onde serão guardados os dados) na inicialização do sistema. A avaliação e testes foram realizados em um ambiente de simulação, não havendo uma implementação funcional para dispositivos.

Thomas e Robble desenvolveram o projecto Smart Phone Ad-Hoc Networks (SPAN) [18] e têm o objectivo de criar uma rede ad hoc de dispositivos Android. Para comunicar através de WiFi em modo peer-to-peer os dispositivos têm de ser reconfigurados, o que requer fazer desbloquear o dispositivo (fazer *root*) para obter permissões de super-utilizador. Esta necessidade causa dificuldades na sua utilização por parte de utilizadores comuns. Para além disso, também funciona apenas numa pequena gama de modelos existentes.

6 Conclusão

Neste artigo apresentamos um sistema de armazenamento descentralizado para redes de dispositivos móveis. O sistema foi desenvolvido como uma biblioteca e desenvolvemos uma aplicação móvel para Android usando essa biblioteca. Os resultados experimentais retirados com os dispositivos móveis mostram que o nosso sistema pode ser usado continuamente durante um evento, como uma festa, um concerto ou um evento desportivo, para partilhar conteúdos sem esgotar a bateria do dispositivo. Os testes de maior escala mostram indícios de que as otimizações realizadas também apresentam potencial para melhorar a eficiência e a escalabilidade do sistema.

Como trabalho futuro temos ainda a realização de uma avaliação mais extensa e com um maior número de máquinas, de maneira a perceber melhor qual o impacto das otimimizações realizadas: os prós e contras do protocolo de replicação por popularidade para os dados mais populares e menos populares, e se diferentes níveis de *churn* também afetam o *overhead* e capacidade de persistência de dados deste protocolo; número de mensagens de pedidos necessárias, e tempo que demora, para obtenção de conteúdos com os diferentes protocolos de replicação; tempo para obtenção de dados para conteúdos de diferentes tamanhos; e, capacidade de manter os conteúdos nos sistema para os diferentes protocolos de replicação;

Atualmente, o protótipo em Android necessita de um ponto de acesso para a comunicação entre os vários dispositivos, como um *router* ou um *ponto de acesso* móvel. O nosso próximo passo é permitir comunicações ad hoc (por exemplo, utilizando comunicações entre múltiplos grupos WiFi Direct) e, com isso, extender a aplicabilidade do nosso sistema para cenários onde não existem pontos de acesso sem fios ou até mesmo quando a rede está sobrecarregada.

Referências

1. Bluetooh, Bluetooth Special Interest Group (SIG). `http://www.bluetooth.com`
2. The Serval Project. `http://www.servalproject.org`, acedido a: 2015-07-14
3. WiFi Alliance, WiFi Direct. `http://www.wi-fi.org/discover-wi-fi/wi-fi-direct`

4. Bocek, T.: TomP2P, A P2P-based high performance key-value pair storage library. http://www.tomp2p.net/
5. Casetti, C., Chiasserini, C., Pelle, L.C., Valle, C.D., Duan, Y., Giaccone, P.: Content-centric routing in wi-fi direct multi-group networks. In: WoWMoM (2015)
6. Chen, K., Nahrstedt, K.: Integrated data lookup and replication scheme in mobile ad hoc networks. In: ITCom. pp. 1–8 (2001)
7. DARPA Strategic Technology Office: Content-based mobile edge networking (CBMEN). http://www.darpa.mil/program/content-based-mobile-edge-networking, acedido a: 2015-07-14
8. Derhab, A., Badache, N.: Data replication protocols for mobile ad-hoc networks: a survey and taxonomy. IEEE Communications Surveys and Tutorials 11(2), 33–51 (2009)
9. Drolia, U., Martins, R., Tan, J., Chheda, A., Sanghavi, M., Gandhi, R., Narasimhan, P.: The case for mobile edge-clouds. In: UIC/ATC '13. pp. 209–215. IEEE Computer Society (2013)
10. Luo, J., Hubaux, J.P., Eugster, P.T.: Pan: Providing reliable storage in mobile ad hoc networks with probabilistic quorum systems. In: MobiHoc. pp. 1–12. ACM (2003)
11. Michel Lombera, I., Moser, L., Melliar-Smith, P., Chuang, Y.T.: Mobile ad-hoc search and retrieval in the itrust over wi-fi direct network. In: ICWMC. pp. 251–258 (2013)
12. Mitzenmacher, M.: The power of two choices in randomized load balancing. In: IEEE TPDS (2001)
13. Panta, R.K., Jana, R., Cheng, F.T., Chen, Y.F.R., Vaishampayan, V., et al.: Phoenix: Storage using an autonomous mobile infrastructure. IEEE TPDS 24(9), 1863–1873 (2013)
14. Shinohara, M., Hayashi, H., Hara, T., Nishio, S.: Replica allocation considering power consumption in mobile ad hoc networks. In: IEEE PerCom Workshops). pp. 463–467 (2006)
15. Stoica, I., Morris, R., Karger, D.R., Kaashoek, M.F., Balakrishnan, H.: Chord: A scalable peer-to-peer lookup service for internet applications. In: SIGCOMM. pp. 149–160 (2001)
16. T. Klinberg and R. Manfredi: Gnutella protocol specification v0.6. http://rfc-gnutella.sourceforge.net/src/rfc-0_6-draft.html (June 2002), acedido a: 2015-07-14
17. Teófilo, A., Remédios, D.: Group-to-group bidirectional Wi-Fi Direct communication with two relay nodes. In: MobiQuitous (2015), to appear
18. Thomas, J., Robble, J., Modly, N.: Off grid communications with android meshing the mobile world. In: IEEE-HST. pp. 401–405 (Nov 2012)
19. Yu, H., Martin, P., Hassanein, H.: Cluster-based replication for large-scale mobile ad-hoc networks. In: WIRELESSCOM. vol. 1, pp. 552–557. IEEE (2005)
20. Zaruba, G.V., Basagni, S., Chlamtac, I.: Bluetrees-scatternet formation to enable bluetooth-based ad hoc networks. In: IEEE ICC. vol. 1, pp. 273–277. IEEE (2001)