Um Modelo de Grupos para MANETs

Henrique S. dos Santos[1], Raqueline R. M. Penteado[1],

Luiz Carlos P. Albini[2] , Carmem Hara[2]

[1]Departamento de Informática
Universidade Estadual de Maringá (UEM) – Maringá, PR – Brasil

[2]Departamento de Informática
Universidade Federal do Paraná (UFPR) – Curitiba, PR – Brasil

henrique.soares.santos@gmail.com, raque@din.uem.br,

albini@inf.ufpr.br, carmem@inf.ufpr.br

Abstract. *Managing a mobile ad hoc network (MANET) is a complex task given the mobility of nodes and resource limitation of the devices. One strategy for simplifying the development of applications in this context is by providing group management services to abstract the network topology and assist the distributed data management. This paper proposes a group management service based on the "Friend of a Friend" (FOAF) ontology. We implemented an NS-2 agent based on the proposed model and simulate the system in a urban scenario in which vehicle drivers exchange information about the traffic. We considered two metrics for validating the system: energy consumption and traffic load.*

Resumo. *O gerenciamento de redes ad hoc sem fio (MANETs) é uma tarefa complexa devido à mobilidade dos nós e à limitação dos recursos dos dispositivos envolvidos. Uma estratégia para simplificar o desenvolvimento de aplicações neste contexto é fazer uso de serviços de gerenciamento de grupos para abstrair a topologia da rede e auxiliar o gerenciamento de dados distribuídos. Neste trabalho é proposto um modelo de grupo para MANETs baseado na ontologia "Friend of a Friend" (FOAF). Para sua análise foi implementado um agente no NS-2 e simulados alguns cenários em um contexto de troca de informações entre motoristas de uma cidade. As métricas consideradas foram o consumo de energia e troca de pacotes.*

1. Introdução

As redes sem fio datam do início da década de 70, quando as universidades das ilhas do Havaí montaram uma rede utilizando microondas para se interligarem, chamada de ALOHAnet [Abramson 1985]. Entretanto, foram nos últimos anos que as redes sem fio se difundiram e ganharam atenção. Para o usuário, a principal vantagem dessas redes é a mobilidade. Utilizando redes sem fio, os usuários podem se mover livremente sem perder a conexão com a rede.

As redes sem fio podem ser divididas em dois grupos: com infra-estrutura e sem infra-estrutura. As redes sem fio com infra-estrutura são caracterizadas pela presença de

estações-base. Essas estações-base são conhecidas nas Redes Locais Sem Fio (WLANS – *Wireless Local Area Networks*) como pontos de acesso (APs – *Access Points*). Todas as comunicações efetuadas na rede devem passar pela estação-base.

As redes sem fio sem infra-estrutura são mais comumente chamadas de redes ad hoc móveis (MANETs - *Mobile Ad hoc Networks)*. Elas são geralmente formadas de modo dinâmico como um sistema autônomo, por unidades móveis (nós) que se comunicam através de enlaces sem fio. Esses nós se comunicam diretamente uns com os outros, não existindo uma estação-base na rede, ou nenhum outro tipo de unidade centralizadora. Cada nó deve ser capaz de se comunicar com os nós que estão dentro do raio de alcance da sua antena. Devido ao raio de alcance limitado das transmissões via rádio, as redes ad hoc são normalmente multi-saltos e os próprios nós devem agir como roteadores das mensagens dos outros nós [Sesay; Yang 2004]. Essas características tornam estas redes propícias para aplicações em diversos cenários, tais como aplicações militares e locais recém atingidos por catástrofes como furacões e terremotos. Uma característica comum nestes cenários é a troca de informações dentro de "grupos de interesse". Por exemplo, após uma catástrofe, podem ser formados grupos de prospecção de danos na infra-estrutura, ajuda a feridos e distribuição de alimentos.

A topologia dinâmica e a limitação dos recursos nos dispositivos em uma MANET, tais como limitações de comunicação, armazenamento, processamento e energia, podem ocasionar partições na rede e dificultar o gerenciamento de dados distribuídos entre seus nós [HARA 2005] [PADMANABHAN et. al. 2008]. Serviços como o de gerenciamento de grupos podem simplificar e facilitar o desenvolvimento de aplicações para uma MANET, permitindo a troca de informação entre grupos específicos. Dessa forma, quando um nó retorna à rede depois de um particionamento, ele deve ser atualizado somente de acordo com os nós do seu grupo, não dependendo dos demais nós da rede. Essa técnica pode economizar tempo e energia dos dispositivos. Os nós devem trocar informações somente com outros que pertençam ao(s) mesmo(s) grupo(s), aumentando a segurança e reduzindo o tráfego na rede.

Uma das maiores dificuldades da criação de serviços para MANETs é a abstração da topologia da rede. Em [BOULKENAFED; SACCHETTI; ISSARNY 2005] um serviço de gerenciamento de grupos genérico é proposto. Neste modelo, os serviços gerenciam o dinamismo de uma rede detectando a entrada e saída voluntárias ou involuntárias de nós em grupos utilizando um modelo de usuário específico. Este modelo é baseado em uma lista de atributos como localização dos nós, grau de confiança entre os nós, proximidade em relação a um ponto específico e distância em número de saltos. Ao contrário deste modelo, no qual o gerenciamento de grupos utiliza um modelo específico, o presente trabalho propõe um modelo que toma como base algumas classes e atributos de uma ontologia padrão, a "*Friend of a Friend*" (FOAF). Essa ontologia tem sido utilizada atualmente em diversas redes de relacionamento para gerenciar perfis, grupos e listas de amigos. Sendo assim, se um nó já possuir uma instância dessa ontologia, o gerenciamento de grupos poderá utilizá-la automaticamente em seus serviços. As contribuições do artigo são listadas abaixo.

- Proposta de um modelo de usuário baseado na ontologia FOAF para o gerenciamento de grupos em MANETs ;

- Implementação do modelo e realização de simulações de cenários na plataforma NS (*Network Simulator*) versão 2.34. Foi considerado um contexto no qual motoristas em movimento trocam informações sobre a fluidez do trânsito de uma cidade. Para garantir a consistência dos dados distribuídos, os cenários consideram replicação total e atualização síncrona e assíncrona entre réplicas;
- Análise dos resultados, considerando as métricas de consumo de energia e troca de pacotes entre os nós de um grupo.

O restante deste artigo está organizado da seguinte forma: a Seção 2 apresenta características da ontologia FOAF e introduz conceitos sobre gerenciamento de dados distribuídos; a Seção 3 descreve o modelo proposto, os cenários simulados e analisa os resultados obtidos; a Seção 4 conclui o artigo e cita direções futuras.

2. Definições Preliminares

Nesta seção é apresentada a ontologia FOAF (*Friend of a Friend* – Amigo de um Amigo), sobre a qual o modelo de grupos deste trabalho foi baseado. São abordados também alguns conceitos de gerência de dados distribuídos.

2.1 Ontologia *Friend of a Friend*

A ontologia FOAF foi criada com o objetivo de facilitar o compartilhamento e a utilização de informações de pessoas e suas atividades. Ela permite que grupos de pessoas criem redes sociais sem a necessidade de um banco de dados centralizado [Brickley; Miller 2010].

O projeto FOAF iniciou-se no ano de 2000 como um projeto experimental de informações interligadas. Atualmente, o projeto é mantido pelo consórcio W3C e é utilizado em diversos sítios da Internet para armazenar dados de usuários e gerenciar perfis, grupos, listas de amigos e transferência de informações entre usuários. Vários exemplos são citados por Golbeck e Rothstein (2008), como o *blog LiveJournal* que possui atualmente mais de 28 milhões de contas criadas e 2 milhões de contas ativas. Redes sociais em destaque atualmente, como o Twitter e o Facebook, disponibilizam ferramentas para que o perfil de um usuário seja convertido para FOAF automaticamente (http://wiki.foaf-project.org/ w/ DataSources#Sites_and_tools_that_export_FOAF).

A ontologia pode ser instanciada por meio de um vocabulário e documentos que podem ser processados de forma automatizada. Estes documentos contêm dados sobre pessoas, suas ligações e atividades ou objetos que elas criam, fazem e possuem interesse. Os participantes da rede podem utilizar os perfis FOAF para encontrar, por exemplo, todas as pessoas que vivem no Brasil, ou ainda uma lista de todas as pessoas que um amigo conhece. Isso é realizado através da definição de ligações entre as pessoas.

A ontologia FOAF define classes, propriedades e associações entre classes. Exemplos de classes incluem: *Agent*, *Person*, *Document*, *Group*, *Image* e *Organization*. Dentre as propriedades de uma pessoa (instância da classe *Person*), encontram-se o nome (*firstName* e s*urname*), imagem (*img*) e interesses (*interest*). Há também associações entre classes. A Figura 1 apresenta um exemplo de documento que segue esta ontologia. Na linha 2, a classe *Group* é instanciada criando o grupo "*dois*". Na linha 4, uma instância da classe *Person*, "*henrique*", é associada ao grupo por meio da propriedade *member* da classe *Group*. Nas linhas que seguem é definida a associação no

sentido inverso, entre a pessoa “*henrique*” e o grupo “*dois*”.

Trabalhos anteriores, tais como *Transhumance* [Paroux et. al. 2007] e *AdhocFS* [Boulkenafed; Issarny 2003] consideram grupos para gerenciamento de dados. Entretanto, eles não deixam explícitos os atributos considerados para o gerenciamento de grupos. O principal objetivo do modelo proposto neste trabalho e apresentado na Seção 3, é distinguir quais nós de uma rede podem ou não trocar mensagens entre si por meio de agrupamentos de nós definidos através da ontologia FOAF.

```
1  <!-- http://xmlns.com/foaf/0.1/dois -->
2  <Group rdf:about="&foaf;dois">
3     <rdf:type rdf:resource="&owl;NamedIndividual"/>
4     <member rdf:resource="&foaf;henrique"/>
5 </Group>
6 <!-- http://xmlns.com/foaf/0.1/henrique -->
7 <Person rdf:about="&foaf;henrique">
8    <rdf:type rdf:resource="&owl;NamedIndividual"/>
9     <member rdf:resource="&foaf;dois"/>
10</Person>
```

Figura 1: Exemplo na ontologia FOAF

2.2 Roteamento e Replicação

Os nós de uma MANET devem ser capazes de se comunicar mesmo quando estão em movimento. Isso cria um novo desafio à computação distribuída, que é a possibilidade de acessar informações em qualquer tempo e lugar. Sendo assim, aplicações criadas para esses ambientes devem definir quais parâmetros deverão ser levados em consideração para o gerenciamento de dados. Além disso, devem ser consideradas as limitações dos dispositivos, bem como a disponibilidade de banda, as frequentes alterações na topologia e o consumo de energia. As aplicações devem ser estruturadas de forma a lidar com a rápida mudança de dados e a variação de desempenho e confiabilidade da conectividade.

Visto que a comunicação entre os nós é ponto-a-ponto, o roteamento de pacotes pode ser do tipo *unicast* ou *multicast*. De uma forma geral, segundo Kant e Awasthi (2010), se um nó quer enviar um pacote para um grupo com três nós, no *unicast* o emissor envia o pacote três vezes, uma para cada receptor. Já no *multicast* o emissor envia somente um pacote para todos os membros do grupo. Entretanto, para usar comunicação *multicast*, diversos outros parâmetros são necessários aumentando a sobrecarga da rede. Dentre os protocolos do tipo *unicast* pode ser citado o AODV [Perkins; Royer; Das, 1999], sendo o PUMA [Garcia; Vaishampayan 2004] um exemplo de protocolo de roteamento *multicast*.

Segundo Hara (2005), a replicação de dados é um ponto importante no gerenciamento de dados em sistemas distribuídos e pode solucionar problemas importantes, como o particionamento de uma rede. Por exemplo, considere que o nó A está executando uma transação utilizando dados que estão alocados no nó B. Caso o nó se movimente, desconectando-se da rede, se não houver réplicas dos dados de B em outros nós da rede, A não poderá continuar a transação. Havendo réplicas, o nó A continuará seu processamento normalmente usando uma réplica. Posteriormente, o nó B deverá ser atualizado para garantir a integridade dos dados distribuídos. Isso leva a um outro ponto importante que é a consistência entre réplicas. A cada alteração dos

dados nos nós, todas as réplicas devem ser atualizadas, garantindo a sua consistência.

O método de replicação pode ser total ou parcial. Na replicação total, os dados são mantidos em todos os nós integrantes de uma rede. Essa abordagem possui um protocolo de propagação simples e não exige um gerenciamento complexo, uma vez que todas as alterações dos nós são enviadas a todos os integrantes da rede. Já na abordagem de replicação parcial, os dados não são replicados para todos os nós da rede, sendo criados subconjuntos de réplicas. Isso poupa espaço na memória dos dispositivos e diminui a quantidade de mensagens para sincronização das réplicas. Porém, ela necessita de um protocolo de propagação mais complexo que o método de replicação total.

Os mecanismos para controle de consistência entre réplicas podem ser classificados de acordo com dois parâmetros: local (quais réplicas podem ser atualizadas) e tempo (quando as réplicas são atualizadas) [Martins; Pacitti; Valduriez 2006]. Quanto ao primeiro, os protocolos de replicação podem ser classificados como *single-master* ou *multi-master*. Nos protocolos *single-master* um item de dado pode ter várias réplicas, porém somente uma delas pode aceitar operações de leitura e escrita. As demais réplicas aceitam somente operações de leitura. Já nos protocolos *multi-master* todas as réplicas aceitam leitura e escrita. Protocolos *multi-master* permitem a concorrência de dados, porém tornam o gerenciamento mais complexo que nos protocolos *single-master*.

Quanto ao parâmetro de tempo, a atualização das réplicas pode ser classificada como síncrona ou assíncrona. Na atualização síncrona, todas as réplicas de um item de dado são sincronizadas antes que o dado seja efetivamente atualizado. Já na assíncrona, as réplicas são sincronizadas periodicamente. Nas duas metodologias existem vantagens e desvantagens. Na atualização síncrona, pode-se garantir que em um determinado momento, todas as réplicas de um item de dado estão com os mesmos valores, enquanto que na assíncrona isso não é possível. Por outro lado, a atualização síncrona pode se tornar um gargalo na execução do sistema, o que pode ser aliviado com a utilização do método assíncrono.

3. *GroupModel*: Um Modelo de Gerenciamento de Grupos

O principal objetivo do modelo proposto neste trabalho, o *GroupModel*, é prover a funcionalidade de definir grupos de interesse, que podem ser constituídos por membros relativamente estáveis ou transitórios. Estes agrupamentos são definidos através da ontologia FOAF e determinam quais nós (membros) da rede podem trocar mensagens entre si, restringindo assim o número de nós nos quais determinados dados são compartilhados. Os estudos experimentais realizados mostram que esta funcionalidade pode proporcionar uma economia no consumo de energia de até 42% em dispositivos móveis.

Para a definição do modelo, algumas classes e atributos da ontologia FOAF foram considerados:

- O atributo *Group* foi definido com base nas classes *Group* e *Person* da ontologia FOAF. *Person* representa um nó que pode ser incluído em um grupo existente por meio da propriedade *member* da classe *Group*. A classe *Group* representa os membros dos grupos na ontologia. Esse atributo é usado no momento do envio de um pacote. O pacote é enviado somente para os membros de um grupo. Portanto o nó já tem conhecimento prévio dos nós que poderão receber o pacote,

seguindo a constituição do grupo.

- O atributo *Interest* do modelo foi definido com base na classe *Person* e no atributo *Interest* da classe *Agent.* A classe *Agent* representa uma superclasse de *Person* na ontologia FOAF. Com a utilização desse atributo, um nó indica qual o interesse, de sua lista, que é considerado para cada pacote enviado. O pacote é então enviado para todos os pares da rede e recebido por aqueles que possuem o mesmo interesse que aquele estabelecido pelo nó origem. O pacote é descartado pelos nós que não tem interesse no pacote.

Sendo assim, um gerenciador de grupos que segue o modelo proposto pode considerar dois métodos para o envio de mensagens: por grupo ou por interesse. O primeiro considera o atributo *Group* e sugere a formação de grupos cujos membros são conhecido a priori e, portanto, com participação "estável". O segundo considera o atributo *Interest*, o que indica uma participação transitória; ou seja, um determinado nó pode, em um determinado instante, ter interesse em receber informações a respeito de um determinado assunto, mas deixar de tê-lo a qualquer momento. Cada nó pode participar de vários grupos e possuir vários interesses.

Para o roteamento de pacotes, foram considerados dois protocolos de roteamento: o protocolo *unicast* AODV e protocolo *multicast* PUMA. Quando o grupo é formado através do atributo *Interest*, é utilizado o protocolo AODV e todos os nós da uma rede recebem o pacote. Quando o envio é feito considerando o atributo *Group*, o protocolo PUMA viabiliza uma entrega direcionada de pacotes para os membros de um determinado grupo.

3.1 Simulações

Para a análise do *GroupModel* foi implementado um agente no simulador de redes NS-2.34, chamado de *GroupAgent*,. Seis cenários foram simulados na ferramenta, considerando uma aplicação do mundo real, que permite a motoristas de uma metrópole trocar informações sobre a situação do tráfego nas regiões nas quais já estiveram. Foram gerados 10 nós (representados pelos carros), distribuídos em três grupos (representados por bairros): Zona Oeste, Centro e Zona Leste. A Figura 2 ilustra o cenário da simulação.

Para todas as simulações, foi adotada a replicação total dos dados, ou seja, os nós replicam seus dados para todos os pares envolvidos. Conforme o cenário, a atualização das réplicas é síncrona ou assíncrona, considerando ou não o modelo *GroupModel*. Além disso, a política de atualização adotada é que todos os nós tem permissão para alterar as réplicas.

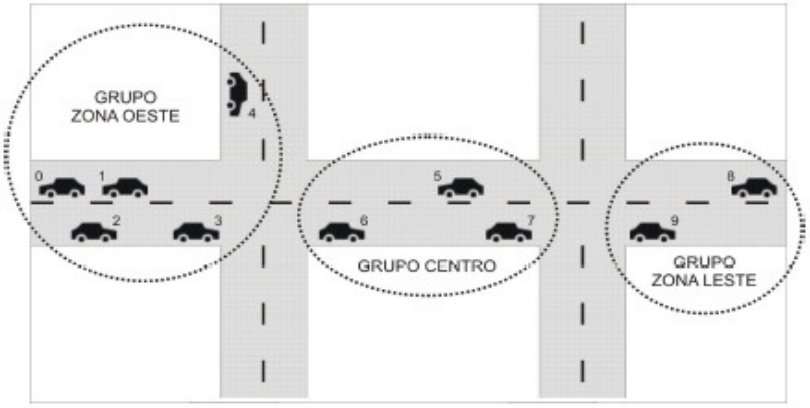

Figura 2. Cenário das simulações

Durante a simulação, os nós movimentam-se arbitrariamente seguindo o modelo *Random Waypoint Model*. Em determinados instantes, alguns nós alteram dados armazenados e replicam suas informações para os demais. Na atualização assíncrona, a propagação é realizada a cada 5 segundos. Já na atualização síncrona, a cada alteração todas as réplicas são atualizadas.

Os cenários considerados nas simulações são descritos abaixo:

- não considerando o *GroupModel*, com atualização síncrona entre as réplicas;
- não considerando o *GroupModel*, com atualização assíncrona entre as réplicas;
- considerando o atributo *Group*, com atualização síncrona entre as réplicas;
- considerando o atributo *Group*, com atualização assíncrona entre as réplicas;
- considerando o atributo *Interest*, com atualização síncrona entre as réplicas;
- considerando o atributo *Interest*, com atualização assíncrona entre as réplicas.

Quando um cenário considera o atributo *Group*, a comunicação entre os nós só pode ocorrer se eles fizerem parte de um mesmo grupo. Por exemplo, entre nós do grupo '*Zona Leste*'. Quando o atributo é *Interest*, os nós devem ter um interesse em comum em seu perfil. Por exemplo, dois nós podem trocar informações se planejam ir para a região oeste da cidade e expressam interesse na '*Região Oeste*'.

Com relação ao modelo utilizado, é possível fazer as seguintes observações a respeito da utilização dos atributos *Group*, *Interest* e sem a utilização de qualquer tipo de agrupamento:

- Nas simulações que não utilizam o *GroupModel*, quando um dado é alterado em algum nó, a sincronização das réplicas é feita para todos os nós da rede. Nesse método, existe um alto tráfego na rede e não existem restrições de acesso aos dados entre os integrantes da rede, uma vez que os nós conhecem todas as informações de todos os pares da rede.
- Quando utilizado o atributo G*roup* do modelo, o nó tem conhecimento, no momento do envio dos pacotes, para quais nós ele deve enviar um pacote. Esse método aumenta a segurança na rede, pois somente os membros de um mesmo grupo podem trocar mensagens entre si. Outra vantagem é que o tráfego na rede é diminuído, pois a replicação de dados não é feita para todos os nós, mas somente para os membros do grupo. A diminuição no tráfego de dados resulta em economia de energia para os nós.
- Com a utilização do atributo *Interest*, o nó necessita enviar um pacote a todos os pares da rede, pois ele desconhece quais pares compartilham o mesmo interesse. Com isso, essa simulação inicialmente se assemelha às simulações que não utilizam o modelo de grupo. Porém, nela um grupo pode ser formado espontaneamente de acordo com o interesse dos nós.

Com relação à sincronização das réplicas, é possível observar que a atualização síncrona pode diminuir a inconsistência dos dados, uma vez que a sincronização é realizada sempre que ocorre uma alteração em um nó. Porém, os recursos da rede são utilizados com mais freqüência, aumentando o tráfego e o consumo de energia dos nós. Muitas vezes ocorrem sincronizações que não precisariam ser feitas, pois nenhum outro

nó necessita da informação replicada. Nos cenários simulados utilizando a sincronização assíncrona, ela é realizada a cada 5 segundos. Mesmo quando ocorrem várias atualizações em um mesmo dado durante esse período, somente a última ocorrida antes do momento da sincronização é propagada a seus pares. Esse método tem por objetivo a redução do tráfego de pacotes na rede, resultando em economia de energia para os nós. Porém, esta abordagem aumenta a possibilidade de ocorrererem leituras de dados inconsistentes, uma vez que um determinado nó pode estar utilizando valores antigos até que a sincronização seja feita.

Para analisar os cenários, primeiramente foi comparado o consumo de energia dos nós e, em seguida, a quantidade de pacotes trocados pelo *GroupAgent*. A análise foi pontual, ou seja, somente o comportamento do agente implementado foi considerado.

- Consumo de energia: cada nó iniciou com energia igual a 100 unidades. A cada transmissão de pacote, ele gastou 0,6 unidades de energia e a cada recepção foram gastos 0,2 unidades de energia. Conforme analisado, a energia utilizada pelos nós foi a mesma não considerando grupos ou grupos formados por *Interest*. Isso aconteceu nos dois modos de atualização de réplicas. Já considerando o atributo *Group* a energia utilizada pelos nós foi aproximadamente 42,72 % menor em relação ao atributo *Interest*.
- Tráfego de pacotes: o número de pacotes enviados/recebidos foi diretamente proporcional ao consumo de energia; ou seja, quanto maior o número de pacotes enviados e recebidos, maior foi o gasto de energia. A troca de pacotes foi menor com o atributo *Group* e maior quando um agrupamento não é considerado ou o atributo *Interest* é considerado. Isso é devido ao fato do atributo *Group* utilizar um protocolo de roteamento *multicast*, enquanto os dois últimos métodos utilizam um protocolo de roteamento *unicast*. Logo, no *multicast* um pacote é enviado uma única vez por um nó e todos os nós de um grupo o recebem. No *unicast* um nó envia um pacote para cada nó destinatário.

Para finalizar a análise, foi gerado um cenário maior e foi realizada uma breve simulação com 50 nós e 20 atualizações de dados replicados em momentos aleatórios. Foram formados dois grupos com 15 nós cada e um terceiro com 20 nós. Nesse cenário foi possível observar uma economia de energia de aproximadamente 94,45% com a utilização do atributo *Group* em relação à utilização do atributo *Interest*. O número de pacotes enviados e recebidos pelo *GroupAgent* foi diretamente proporcional ao consumo de energia. Considerando esse cenário, ficou ainda mais evidente a vantagem da utilização do atributo *Group*, no qual o aumento do número de nós e a disposição dos grupos resultou em um grande ganho de desempenho na rede.

4. Conclusão

Redes sociais populares vêm empregando a ontologia FOAF para definir os perfis de seus usuários. Sendo assim, os milhares de usuários que já possuem um perfil neste padrão poderão utilizar o serviço proposto neste artigo para realizar trocas de informações em uma MANET, sem a necessidade de criar um novo perfil.

Considerando as métricas consumo de energia e troca de pacotes na análise dos cenários que utilizam o *GroupModel*, foi possível concluir que a redução no tráfego da rede e a consequente diminuição do consumo de energia se deram principalmente com a utilização do atributo *Group* do modelo. Minimizar o consumo de energia é sempre

uma preocupação nas aplicações em uma MANET, uma vez que isso implica em menos partições de rede e desconexões de nós. A estabilidade da rede, por consequência, facilita o gerenciamento de dados distribuídos. Com o atributo *Group*, nós de uma rede podem ser agrupados em "sub-redes" facilitando o gerenciamento de dados por parte de sistemas distribuídos.

Com a utilização do atributo *Interest*, as simulações se mostraram equivalentes ao uso de modelos tradicionais. Porém, o uso do atributo *Interest* apresenta vantagens porque grupos podem ser formados espontaneamente de acordo com um interesse de um conjunto de nós. Isto é interessante pois redes podem ser formadas entre usuários desconhecidos que possuem interesses em comum em locais arbitrários.

Um outro ponto que merece destaque é que a segurança em uma rede pode ser mais alta com a utilização do atributo *Group*, uma vez que apenas nós que pertençam à um mesmo grupo podem se comunicar. Já com a utilização do atributo *Interest* ou no modelo tradicional, o envio é feito para todos os nós da rede, dificultando assim a manutenção da segurança.

Embora o artigo apresente resultados preliminares da utilização do conceito de grupos em MANETs, os resultados obtidos com a utilização do atributo *Group* demostram o potencial da abordagem. Dentre as direções futuras deste trabalho podem ser citados:

- Eleição de um nó líder para cada grupo estável: para que o atributo *Group* seja utilizado por um gerenciador de grupos, todos os nós devem ter seus grupos atualizados. Sendo assim, existe a necessidade de um nó líder para aceitar inscrições de nós em grupos e refletir atualizações necessárias para os nós que pertençam aos grupos alterados. Esse nó líder deve ter uma carga de energia satisfatória para gerenciar alterações de grupos e também deve ter uma alta disponibilidade. Vários protocolos de sistemas distribuídos tratam de eleição de líderes considerando características como nível de energia e partição de redes.
- Controle de vocabulário: dois nós podem ter um mesmo interesse, porém, com palavras-chave de interesse diferentes. Uma extensão interessante seria considerar sinonímia no processo de verificação de interesses em comum entre usuários. Trabalhos na área de processamento semântico de texto tratam deste tipo de detecção.
- Replicação e sincronização: um estudo considerando todas as camadas de uma MANET seria interessante para analisar o modelo proposto como um todo. O estudo de formas alternativas de replicação e sincronização, tais como a replicação parcial e sincronização híbrida [Padmanabhan et. al. 2008] serão investigadas no futuro.

Referências

Abramson, N. (1985). Development of the alohanet. IEEE Transactions on Information Theory, 31.

Boulkenafed, M.; Issarny, V. (2003) "AdHocFS: Sharing Files in WLANS", Second IEEE International Symposium on Network Computing and Applications, Massachusetts (USA).

Boulkenafed, M.; Sacchetti, D.; Issarny, V. (2005) "Using Group Management to Tame

Mobile Ad Hoc Networks", Proceedings of the 6th International Conference on Mobile Data Management, Nicosia (Cyprus), po. 192-199.

Brickley, D.; Miller, L. (2010) "FOAF Vocabulary Specication 0.98", Namespace Document 9 August 2010, FOAF Project, http://xmlns.com/foaf/spce/.

Garcia, J., J.; Vaishampayan, R. (2004) "Efficient and Robust Multicast Routing in Mobile Ad Hoc Networks", Proceedings of the IEEE International Conference on Mobile Ad Hoc and Sensor Systems, Florida (USA), pp. 304-313.

Golbeck, J.; Rothstein, M. (2008) "Linking Social Networks on the Web with FOAF", Proceedings of the 17th International World Wide Web Conference (WWW2008), Beijin (CN).

Hara, T. (2005) "Data Replication Issues in Mobile Ad Hoc Networks", Proceedings of the 16th International Workshop on Database and Expert Systems Applications (DEXA'05), Copenhagen (DNK).

Kant, K.; Awasthi, L. K. (2010) "Unicast and Multicast Routing Protocols for Manets: A Comparative Survey", International Journal of IT & Knowledge Management (IJITKM), ISSN: 0973-4414, Special issue, January.

Martins, V.; Pacitti, E.; Valduriez, P. (2006) "Survey of data replication in P2P systems", Technical Report, Institut National de Recherche en Informatique et en Automatique, Nantes (FRA).

Padmanabhan, P. et. al. (2008) "A Survey of Data Replication Techniques for Mobile Ad Hoc Network Database", The VLDB Journal — The International Journal on Very Large Data Bases, Volume 17 Issue 5, August.

Paroux, G. et al. (2007) "Transhumance: A power sensitive middleware for data sharing on mobile ad hoc networks", International Workshop on Applications and Services in Wireless Networks, Santander (ESP).

Perkins, C.; Royer, E.; Das, S. (1999) "Ad hoc on demand distance vector (AODV) routing", Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications, New Orleans (USA), pp. 90-100.

Sesay, S.; Yang, Z.; He, J. (2004) "A Survey on Mobile Ad Hoc Wireless Network", Information Technology Journal 3(2), pp. 168-175.