Instituto Alberto Luiz Coimbra
de Pós-Graduação
e Pesquisa de Engenharia
Universidade Federal
do Rio de Janeiro

Programa de Engenharia de Sistemas e Computação

Backbones Ad Hoc

Aluno: Eduardo Hargreaves
Orientador: Luís Felipe M. de Moraes

Dezembro de 2003

Resumo

O objetivo deste trabalho é apresentar propostas para tornar as redes Ad Hoc escaláveis. As redes Ad Hoc são especialmente atraentes porque funcionam sem nenhuma necessidade de infraestrutura pré-existente. Esta rede deve funcionar mesmo em cenários com grande mobilidade. Isto impõe muitos desafios. Artigos mostram que o desempenho destas redes cai consideravelmente a medida que o número de nós aumenta.Este trabalho foca principalmente nas soluções que tentam formar um backbone visando tornar estas redes escaláveis.

1)Introdução

As redes Ad Hoc mostraram um grande potencial por funcionarem sem nenhuma infraestrutura e por serem implementadas de maneira instantânea. Estas redes se auto organizam, se auto configuram além de mudar de topologia constantemente devido a mobilidade. Neste tipo de rede os nós também são roteadores. Todos os nós também compartilham o mesmo canal de comunicação. Os protocolos de roteamento projetados para rede Ad Hoc devem se adaptar a estas mudanças freqüentes e imprevisíveis na topologia da rede utilizando os recursos de forma eficiente.

Os protocolos de roteamento geralmente são baseados em uma rede flat sem nenhum nível de hierarquia. A media que o tamanho de uma rede aumenta é de se esperar que a sobrecarga causada por estes protocolos aumente também.

Os protocolos de roteamento geralmente se dividem em duas categorias:os que usam algoritmos de estado de enlace onde os nós tem conhecimento de todos os nós de uma rede e o que usam algoritmos de vetor de distância. Geralmente os algoritmos que usam o estado de enlace sofrem porque a medida que a rede aumenta fica mais difícil ter o conhecimento sobre a rede toda, além to tamanho da tabela de roteamento aumentar consideravelmente (isto implica em pacotes de roteamento maiores). Protocolos que usam vetor de distância apresentam um overhead menor que os que usam o estado de enlace em compensação na média levam mais tempo para convergir e tendem a criar loops. O desempenho destes protocolos flat piora ainda mais a medida que a mobilidade aumenta. Em grandes redes a troca periódica de mensagens pode consumir grande parte da banda disponível tornando impraticáveis certas aplicações.

Uma saída proposta foi a invenção de protocolos de roteamento que funcionam sob demanda. Neste caso nenhuma atividade de roteamento é presenciada se não houver comunicação. As rotas para um determinado nó só são guardadas enquanto este nó está se comunicando com alguém. Isto funciona bem sob um pequeno tráfego concentrado num pequeno número de nós. Existem dois protocolos bastante difundidos: o AODV e o DSR.

Estruturas hierárquicas se apresentam como uma grande solução para prover escalabilidade em grandes redes. A Internet usa este tipo de solução.Nas redes móveis o problema é muito complicado e a principal contribuição deste trabalho é apresentar algumas soluções encontradas para este problema.

O restante do trabalho está organizado da seguinte forma: a seção 2 fala sobre roteamento sob demanda e explica com mais detalhes o DSR e o AODV a seção 3 explica uma solução para a criação de backbones chamada de VDBP a seção 4 apresenta o MNP a 5 o TBONE na seção 6 são feitos alguns comentários e a conclusão é feita na seção 7.

2)Roteamento sob demanda

Com o roteamento sob demanda só há troca de pacotes de roteamento quando existe uma comunicação entre dois nós. Este comportamento reativo difere do comportamento proativo dos protocolos de roteamento convencionais que sempre tentam encontrar rotas independente se existe comunicação ou não. A principal motivação destes algotitmos é reduzir o overhead de roteamento. Em compensação os algoritmos sob demanda geralmente têm uma fase de descobrimento de rotas onde geralmente ocorre um flooding de pacotes procurando por uma rota até o destino procurado. Existem dois protocolos bastante difundidos que operam de acordo com esta filosofia: o AODV (Ad Hoc On Demand Distance Vector) e o DSR (Dynamic Source Routing) . Apesar dos dois funcionarem sob demanda o modo como eles descobrem as rotas é bastante diferente. O funcionamento deles será explicado na próxima seção.

2.1)DSR-Dynamic Source Routing

Uma das características principais do DSR é que o roteamento é todo feito pela fonte, isto é, a origem tem total conhecimento sobre o caminho desde a origem até o destino. A fonte informa, através do cabeçalho dos pacotes de dados que ela envia, a rota que estes pacotes devem seguir. A grande vantagem deste algoritmo é que os nós intermediários não precisam armazenar informações atualizadas sobre as rotas dos pacotes encaminhados uma vez que o caminho já é informado através do cabeçalho dos pacotes. Este fato aliado ao fato da forma de atuar sob demanda do DSR elimina a necessidade de troca de informações períodicas sobre a descoberta/manutenção de rotas.

Quando uma nó tenta enviar uma pacote para um destino do qual ele desconhece a rota ele incia um processo de descobrimento de rotas para dinamicamente encontrar esta rota. Este descobrimento de rotas é feito através de um *flooding* de um pacote de descobrimento de rotas (*route request RREQ*). Cada nó que recebe um RREQ reencaminha este RREQ (guardando sua identidade) desde que não seja o destino procurado nem saiba a rota até o destino desejado. Se o nó for o destino ou conhecer uma rota até o destino desejado então ele responde com um pacote de *Route Reply* (*RREP*) que é roteado de volta ao nó de origem (usando o *source routing*), desta forma a fonte descobre o caminho desejado até o destino. È importante notar que múltiplas rotas podem ser descobertas. Um nó DSR agressivamente armazena as rotas aprendidas de forma a minimizar o custo associado ao descobrimento de rotas.

Quando é detectado um rompimento de uma rota, como múltiplas rotas são conhecidas o processo de descobertas de novas rotas pode ser adiado. Se não existir nenhuma rota alternativa o processo de descobrimento de rotas deve ser reiniciado.

Para aumentar a eficiência do protocolo ele funciona de modo promíscuo.Com isso ele aprende novas rotas sem participar do processo de roteamento.

2.2)AODV

Este protocolo utiliza as tabelas de roteamento convencionais, e guarda uma entrada desta tabela para cada destino. Cada nó que deseja encontrar uma rota envia uma mensagem perguntando qual é a caminho até o destino procurado. Os nós que recebem esta mensagem "aprendem" o caminho até a fonte desta mensagem (analisam por onde esta mensagem chegou) e guardam esta informação na sua tabela de roteamento e reecaminham esta pergunta. O destino desejado eventualmente recebe a pergunta e responde usando o mesmo caminho que a pergunta atravessou. Desta forma é estabelecido um caminho full duplex entre estes dois pontos. Para redução do overhead sempre que o nó receber uma cópia da pergunta ele descarta este pacote ao invés de reencaminhá-lo. Após a rota ser estabelecida ela é mantida enquanto durar a conexão. Se um enlace falhar isto vai ser reportado recursivamente até chegar a fonte. Isto iniciará um novo processo de descoberta de rotas. Se esta rota não for usada durante um determinado tempo, os nós tiram esta rota da tabela da roteamento. O AODV usa número de seqüências mantidos em cada destino para verificar a validade das rotas estabelecidas e evitar loops. Todo o pacote roteado carrega estes números de seqüência.

2.3)Algumas características do Roteamento sobre demanda

Operações sob demanda acontecem apenas quando existe uma necessidade de comunicação entre dois nós. O Overhead de roteamento está relacionado então ao descobrimento e manutenção das rotas em uso. Sob baixo tráfego (direcionado a poucos destinos) e sob baixa mobilidade, estes protocolos funcionam mesmos em redes bem grandes. Se o número de destinatários aumentar, e mais fontes vão procurar por estes destinos.Se a mobilidade aumentar as rotas descobertas não vão servir mais então novos procedimento de descoberta de rotas serão necessários. Como cada processo de descobrimento é feito através de flooding, os protocolos de roteamento vão causar uma enorme sobrecarga na rede e este overhead tende a crescer com uma taxa muito alta a medida que o cenário vai se tornando mais desfavorável. No DSR a situação é ainda pior porque redes grandes implicam em muitos hops o que aumenta bastante o tamanho do pacote, já que no cabeçalho do pacote vêm expresso todo o caminho desde a origem até o destino.

Além de tudo isso como antes de qualquer transmissão ainda deve-se passar pelo processo de descoberta de rotas os protocolos de roteamento ainda sofrem uma grande latência inicial.

3)Estruturas Hierárquicas

Tipicamente quando o tamanho de uma rede aumenta bastante os algoritmos de roteamento "planos" se tornam impraticáveis devido ao grande overhead gerado pelos protocolos de roteamento. Uma outra forma de reduzir esta sobrecarga garantindo escalabilidade é criar redes hierárquicas. Um exemplo de rede que usa hierarquia é a Internet. A idéia básica é organizar os nós de uma rede ad hoc em grupos e atribuir diferentes funções aos nós pertencentes a este subgrupo. Neste caso as tabelas uasads pelos protocolos de roteamento tem seu tamanho bastante reduzido pois nelas estão guardadas

apenas endereços de parte da rede. A maneira mais popular de implementar isto é agrupar os nós geograficamente mais próximos em clusters. Cada cluster normalmente tem um líder, chamado de clusterhead, que tem funções especiais como por exemplo, executar tarefas de roteamento. Existem diversas propostas de implementação, neste trabalho será dada ênfase aos protocolos que implementam esta hierarquia através de backbones. Estes backbones podem ser implementados de maneira lógica ou de maneira física onde determinados nós com hardwares especiais formam um backbone real.

4)VDBP - Virtual Dynamic Backbone Protocol

Este protocolo foi proposto com o objetivo de redução da sobrecarga causada pelos protocolos de roteamento, como uma forma de garantir a QoS de serviço requerida e de tornar viável o tráfego com endereçamento multicast . Ele tem como objetivo formar um backbone lógico com apenas um conjunto de nós de uma rede Ad Hoc satisfazendo duas características :

1) Todo nó desta rede Ad Hoc ou será um nó que faz parte de um backbone ou será vizinho de um nó que faz parte de um backbone
2) Qualquer par de nós de backbone é conectado através de nós de backbones

Este protocolo tenta criar backbones estáveis e de tamanho razoável mesmo perante cenários de grande movimentação.Como em uma rede Ad Hoc todos os nós são simultaneamente hosts e roteadores o simples fato de apenas alguns nós executarem o papel de roteadores já implica em uma redução do tráfego total nesta rede Ad Hoc. O VDBP consiste de três fases:

1) Processo de seleção dos nós de backbone (*backbone selection process* – BSP)
2) Processo de conexão dos nós de backbone (*backbone connection process* – BCP) – para esta fase existem duas propostas : o BCP-I e o BCP-II
3) Processo de manutenção do backbone (*backbone maintenance process* - BMP)

Todas as fases dão maior prioridade aos nós mais estáveis (nós estáveis são aqueles que não mudam muito de vizinhos, é importante notar que um nó estável não está necessariamente parado, ele pode mover junto com um grupo de nós) e aos nós com um maior número de vizinhos (o que tende a formar backbones com menos nós) enquanto constroem e/ou mantêm o backbone. A seguir, o protocolo será explicado com mais detalhes.

3.1)Modelo da rede

Assume-se que todos os nós transmitem com a mesma potência e possuem antenas omnidirecionais. Os nós compartilham um mesmo canal de comunicação (isto é, mesma faixa de frequência), o protocolo de acesso ao meio pode ser aleatório ou baseado em reservas, os enlaces são bidirecionais e half-duplex. Assume-se também uma rede toda conectada.

3.2)Notação

Para distinguir os nós foram criadas três cores para os nós: os Nós negros que fazem parte do backbone, os nós verdes que são os vizinhos dos nós de backbone mas não fazem parte do backbone e os nós brancos que são os nós que não se encaixam em nenhum dos dois grupos. Inicialmente todos os nós são brancos.

- N: conjunto de todos os nós da rede Ad Hoc
- D: conjunto de nós dominantes (os nós que são escolhidos na fase 1)
- C: conjunto de nós que conectam os membros de D
- B: conjunto de nós de backbones isto é, $B \equiv D \cup S$
- $d_w(j)$, $d_g(j)$, $d_b(j)$: conjunto de nós brancos, verdes e negros vizinhos ao nó j
- d(j): soma de $d_w(j)$, $d_g(j)$, $d_b(j)$ indica o número total de vizinhos (grau) do nó j
- cor(j): denota a cor do nó j
- VAP(j): O nó de backbone que o nó j está associado (VAP – *Virtual Access point*)
- $VAP_{list}(j)$: subconjunto dos nós negros vizinhos ao nó j através dos quais o nó j é capaz de alcançar todos os destinatários possíveis
- NLFF(j): freqüência normalizada de enlaces que falharam *(Normalized Links Failure Frequency*) envolvendo o nó j. Indica o número de links que falharam por unidade de tempo normalizada em relação ao número de vizinhos do nó j indicado por d(j)
- NIT: tabela que guarda informações sobre os vizinhos (*neighborhood information table*). Esta tabela armazena o grau , a cor, o NFLL, e VAP dos nós vizinhos
- SNT: tabela dos números de seqüência (*sequence number table*). Esta tabela armazena o número de seqüência das mensagens globais de difusão (que serão explicadas melhor a diante) recebidas diretamente por seus vizinhos, exceto seu VAP.
- ID(j): Número de identificação (endereço lógico) do nó j.
- T_{BSP}: Contador usado durante a fase BSP pelos nós brancos
- T_{BN}: timer associado ao nó de backbone usado para eliminar certos nós de backbone
- dis(i,j): menor distância (em número de hops) entre o nó i e o nó j

3.3)mensagens de controle

Estipulou-se que o VDBP usaria 4 mensagens de controle:

Hello messages: Cada nó periodicamente envia mensagens de difusão aos seus vizinhos diretos. Esta mensagem contém os seguintes campos: (ID(j), cor(j), NLFF(j), $d_w(j)$, $d_g(j)$, $d_b(j)$, VAP(j))

Global broadcast messages: esta mensagem é gerada periodicamente por todos os nós. O intervalo entre estas mensagens deve ser maior que o intervalo entre as mensagens de Hello. Esta mensagem inclui todos os campos de mensagem de hello mais os seguintes

campos: identidade do nó que reencaminhou a mensagem, número de seqüência, *time-to-live*(TTL), $VAP_{list}(j)$. Sempre que o negro recebe uma mensagem deste tipo , ele retransmite esta mensagem atualizando os seguintes campos: (cor(j), NLFF(j), $d_w(j)$, $d_g(j)$, $d_b(j)$, VAP(j)). Ele também põe a sua própria identidade no campo que indica quem encaminhou a mensagem.

Color change message: Esta mensagem só é usada pelo BCP-I e tem a finalidade de eliminar redundâncias durante o processo de conexão do backbone.

Unicast message: Esta mensagem é usada pelo BCP-II Para iniciar o processo de conexão do backbone. Ele serve para forçar alguns nós verdes a se tornarem negros.

Soft selection message: esta mensagem é usada pelo BMP para evitar redundâncias no número de nós negros.

3.4)Funcionamento do VDBP

Como já dito anteriormente o objetivo deste protocolo é criar um backbone estável perante cenários com grande mobilidade e não criar um backbone com muitos nós para diminuir a sobrecarga causada pelos protocolos de roteamento.Para este objetivo ser atingido os autores propuseram duas métricas principais: a partir do campo NLFF é possível descobrir quais são os nós mais estáveis, já que ele indica a freqüência do número de enlaces que falharam normalizada pelo número de vizinhos, é obvio que quanto menor este valor mais estável o nó é, ou seja muda de vizinhos com uma freqüência menor. Para atingir o segundo objetivo uma outra métrica é o número de vizinhos de um nó. Durante a fase BSP usa-se o número de vizinhos brancos de um nó como métrica e durante a fase BCP usa-se o número total de vizinhos. Se houver algum empate , ganha quem tiver a menor identificação. Como cada nó é unicamente identificado, as chances de empate são nulas.

A seguir cada fase será explicada com um pouco mais de detalhes.

Backbone Selection process

O objetivo principal desta fase é eleger um número mínimo de nós líderes chamados de *clusterheads,* usando-se apenas informações obtidos com os vizinhos. Ele se encerra quando todos os nós ou são negros ou são verdes. O algortimo é o seguinte:

Quando o nó j é ligado ele toma as seguintes ações:

1) Marca sua cor como branca e faz seu VAP valer NULL
2) Envia mensagens de hello e GBM
3) Inicializa seu contador T_{BSP} e começa a atualizar o NIT baseada nas mensagens Hello e GBM que recebe
4) Se o contador T_{BSP} expirar ele verifica (baseado nas métricas já explicadas) se ele foi o nó que ganhou a "eleição"
   Se ganhou muda de cor para preto. Envia mensagens hello e GBM (atualizadas com sua nova cor)
   Se não,sua cor continua branca e reinicializa o contador T_{BSP}.Volta para (3)
5) Se receber alguma mensagem hello ou GBM de um nó negro i, faz seu VAP valer i, muda sua cor para verde, e faz T_{BSP} valer infinito.

Os autores mostraram que este algoritmo converge em um tempo finito, que nunca dois nós pertencentes ao grupo dominante são vizinhos, e que 3 hops é a distância máxima entre dois nós dominantes.

Fase 2: Backbone Conection Process

Esta fase constrói o backbone unindo os nós dominantes escolhidos na fase 1.Existem duas propostas:

BCP-I

Nesta proposta o backbone é forma com o auxílio das mensagens GBM que são retransmitidas pelos nós negros. Este protocolo se baseia no seguinte fato: Numa rede conectada todo o nó verde deve receber pelo menos duas cópias de uma mesma mensagem GBM. Se isto não acontecer significa que a rede não está conectada (não existe o backbone).

Cada nó verde armazena uma tabela de números de seqüência (SNT). Nesta tabela são armazenados os campos que contém a identificação da fonte e o número de seqüência de uma mensagem GBM. Somente uma única mensagem é representada por este par. Se o nó j recebe uma mensagem reencaminhada pelo seu VAP este nó descarta as entradas da sua tabela SNT com número de seqüência inferior ou igual ao número de seqüência da mensagem recebida correspondentes a fonte indicada pela mensagem GBM. Se o número de entradas na tabela superar um determinado limite, este nó verde considera a rede desconectada e então se torna um nó negro. Para que muitos nós verdes não se tornem negros simultaneamente um nó verde i que se tornou negro armazena a identidade daquele nó que ele não recebeu a mensagem GBM através do seu VAP(i) . Este nó é então chamado de reason(i).Ele então envia uma mensagem CCM contendo VAP(i) e os vizinhos de i. Um nó j que acabou de ser tornar negro que recebe esta mensagem verifica as seguintes condições:

1)VAP (j) = VAP(i) ou VAP(j) $\in$ NIT(i)
2) reason(j) $\in$ NIT(i)
3)ID(i)>ID(j)

Se estas três condições forem satisfeitas o nó j volta a ser um nó verde.

BCP-II

Nesta proposta quem escolhe qual nó verde se tornará um nó negro são os nós negros. Nesta fase todo nó negro tenta se conectar com os nós negros que estão a uma distância inferior a 3 hops. Como todos os nós negros tentam fazer o mesmo, após esta fase teremos uma rede conectada.

Sempre entre dois nós negros consecutivos ou teremos um nó verde ou teremos dois nós verdes. Para cada caso uma atitude diferente será tomada.

Caso em que existem 2 nós verdes: Suponha que o nó i perceba que existe um nó vizinho j com um VAP diferente do VAP de i. i então envia uma mensagem unicast ao seu VAP indicando o VAP(j). VAP(i) faz as seguintes verificações:1)VAP(j) $\notin$ NIT(VAP(i));

2)Verifica se não existe nenhum k $\in$ NIT(VAP(i)) cujo VAP(k) = VAP(j).Ou seja nestes passos ele verifica primeiro se VAP(j) é vizinho de VAP(i) e depois verifica se VAP(j) está a dois hops de VAP(i). Se estas condições forem verdadeiras VAP(i) decide então transformar i em nó negro e i escolhe um vizinho ótimo k* cujo VAP seja VAP(j) para também se transformar em um nó negro.

Caso em que exista apenas um nó verde: Este caso é mais simples. Se um nó negro i perceber que existe um nó negro j não pertencente a sua vizinhança mas que é vizinho de um de seus vizinhos e que ainda não se iniciou nenhum processo de conexão, o nó i escolhe um nó k que seja vizinho de j para se tornar negro.

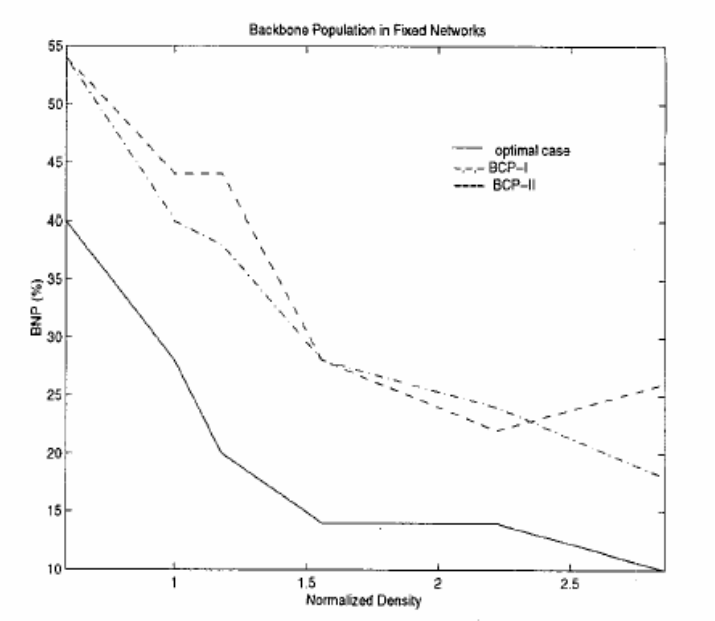
Backbone maintenance process

Esta fase tenta manter o backbone ativo junto com o BCP. O BMP serve para lidar com a mobilidade dos nós. Existem 3 padrões de mobilidade distintos que causam disconexão do backbone: 1) movimentação do VAP 2) movimentação de um nó não VAP 3)movimentação de um nó verde

Para cada ocorrência uma ação diferente é tomada. Caso 1: se o nó VAP i se movimentar e não houver mais nenhum k VAP(k)=i este nó deve mudar sua cor para verde. Caso 2: se nesta nova região o nó i não encontre nenhum nó k tal que VAP(k)=i o nó então se torna verde. Caso 3) o nó procura algum nó negro para se associar, se durente este processo o nó receber alguma mensagem hello indicando que ele deve mudar de cor este nó muda sua cor para negra.

3.5)Resultados

Os autores desta proposta [3] definiram duas métricas: o número de nós de backbones(nós negros) e o percentual do tempo em que o backbone se mantém conectados. As simulações foram feitas usando-se o NS-2. cada estação usa o DCF 802.11 como protocolo de aceso ao meio. Todas as estações usam antenas omnidirecionais com alcance de 250 metros com uma taxa nominal de transmissão de 2 Mbps. A primeira simulação os nós foram aleatóriamente distribuídos mas estão estáticos. Nas próximas simulações utizou-se o random way point como modelo de mobilidade.



ig 1 desempenho do BCP-I e BCP-II em relação ao caso ótimo

Nesta simulação o objetivo foi comparar o número de nós de backbone para o BCP-I e BCP-II com o número ótimo. Os dois protocolos têm um desempenho bastante parecidos.Esta redundância é até certo boa para prover uma certa confiabilidade.

Na segunda simulação testou-se o comportamento dos nós em função da mobilidade. Na figura 2 o tempo de pausa é zero, na figura 3 a velocidade máxima é constante o que varia é o tempo de pausa. Os 100 nós estão distribuídos numa área de 1400m por 500m.

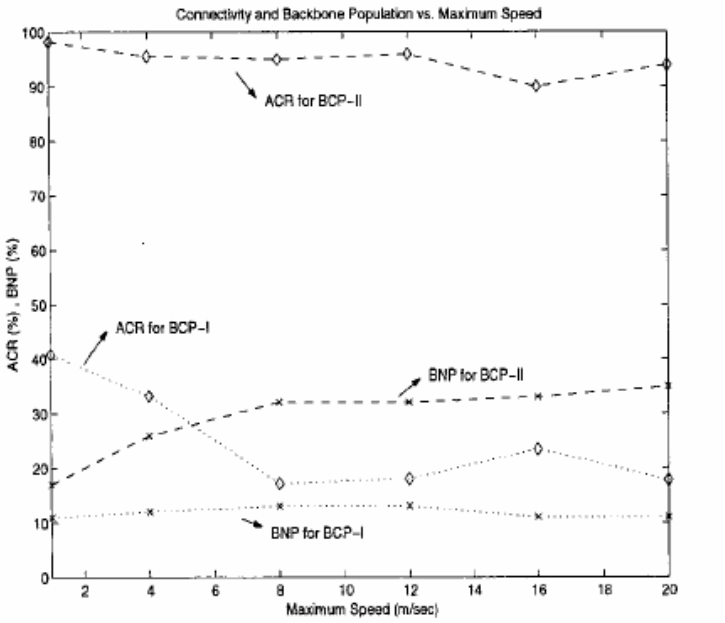


Fig 2 perfomance do BCP-I e BCP-II em função da velocidade máxima

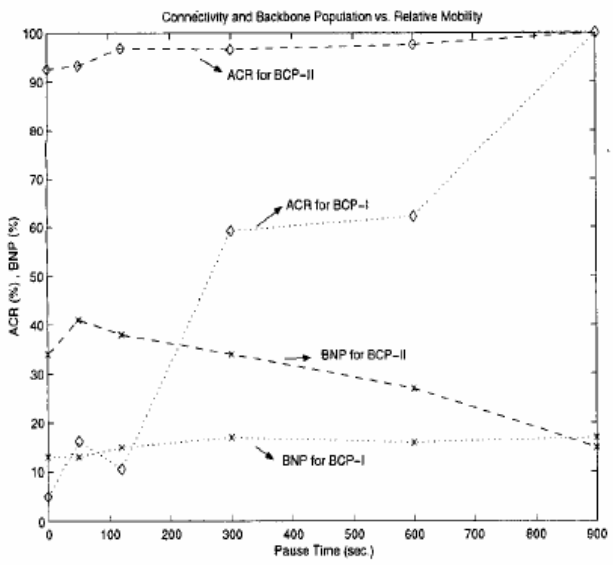


Fig 3. desempenho do BCP-I e BCP-II em função do tempo de pausa

Com BCP-II o backbone fica conectado durante uma grande parte do tempo em compensação o número de nós que fazem parte deste backbone é maior.

4)Mobile Backbone Network (MBN)

Gupta em [5] demonstrou que mesmo sob condições ótimas, isto é, os nós estão separados a uma distância ideal uns dos outros, os nós tem um raio de alcance ideal a capacidade um nó de uma rede Ad Hoc é limitada pela função $(\frac{W}{\sqrt{n}})$ onde W é a largura de faixa das estações e N representa o número de estações. Fica claro que o throughput decai rapidamente com o aumento do número de estações. No entanto espera-se que um número muito grande de estações faça parte de uma rede ad hoc. Uma solução seria dividir uma rede ad hoc em clusters onde cada nó pertencente a um cluster falaria apenas com os nós pertencentes ao seu cluster exceto os cluster-heads. Desta forma este cluster seria uma pequena rede ad hoc . Como o número de nós desta rede é bem menor, a capacidade por nós iria aumentar consideravelmente. A seguir duas implementações serão explicadas com mais detalhes.

Xu et al. em [6] propôs um protocolo para formar um backbone Ad Hoc também com o intuito de aumentar a capacidade das redes Ad Hoc. Os autores propuseram um protocolo hierárquico com alguns nós com rádios de maior capacidade (por maior capacidade entende-se maior alcance e maior largura de faixa). Estes nós com maior capacidade vão servir para formar um backbone real Ad Hoc diferente daquele backbone lógico proposto no VDBP. Devido ao fato do backbone ser real aquela fase de conexão do backbone do VDBP não existe porque um determinado nó de backbone vai ser conectado com todos aqueles nós que estão sob o alcance do seu rádio mais poderoso.

Neste protocolo a rede Ad Hoc é dividida em grupos (clusters) onde os nós são agrupados dinamicamente. Cada grupo elege um líder (cluster-head), é claro que este líder deve possuir ao menos dois rádios, para ser um nó de backbone. Enlaces de mais alto nível são então estabelecidos. Com este enlace de mais alto nível caminhos mais curtos podem ser estabelecidos. A figura abaixo exemplifica uma possível rede.

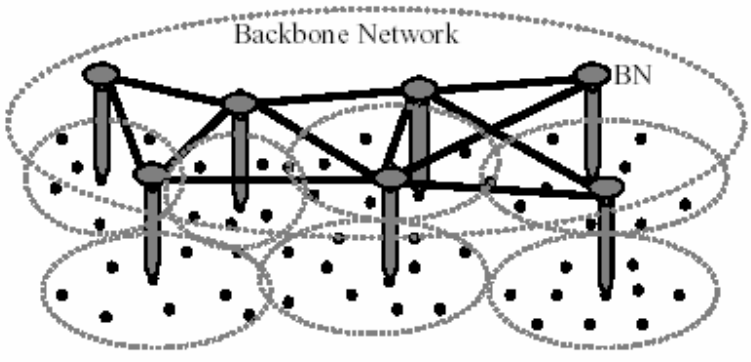


fig 4 uma rede MBN

Para esta rede ser construída este protocolo considera que três decisões devem ser tomadas: qual é o número ideal de nós de backbone? Como eles serão eleitos? Qual será o protocolo de roteamento utilizado?

As formas de lidar com estes problemas serão explicadas mais adiante

4.1)Funcionamento do MBN

Número ótimo de nós de backbone

Seja N o número total de nós pertencentes a uma rede ad hoc, m o número de clusters, W1 a largura de faixa do cluster local, W2 a largura de faixa do backbone. Considerando os nós espalhados de maneira uniforme o número de nós por cluster será dado por N/m. Então a partir de [1] :

$$R_{local} = \Theta(\frac{W_1}{\sqrt{N/m}})$$

e

$$R_{backbone} = \Theta(\frac{W_2}{\sqrt{m}})$$

onde R_{local} é a capacidade do cluster local e $R_{backbone}$ é a capacidade do backbone. Se W1 , W2 e N forem mantidos fixos, R_{local} e $R_{backbone}$ serão função apenas de m. O objetivo portanto é encontrar um valor m* que seja bom as duas funções. Um nó de backbone tem duas interfaces uma para a rede local outra para o backbone. Assumindo o tráfego uniforme, a porção do trafego local que entra ou sai de um cluster é dada por m-1/m, como o objetivo é não congestionar a rede este tráfego deve ser menor do que o tráfego que o nó de backbone é capaz de escoar. Então;

$$\frac{m-1}{m} R_{local} \leq R_{backbone}$$

fazendo-se o gráfico das duas curvas o ponto de interseção entre as duas é o valor ótimo m*

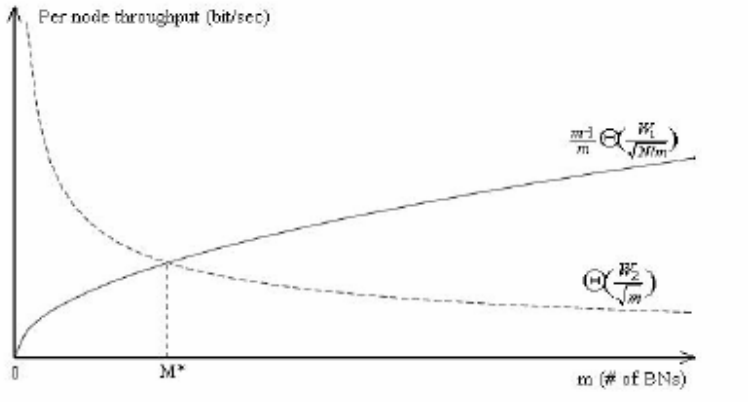


fig 5 vazão em função do número de nós

4.2)Formação dos clusters

Apenas os nós equipados com rádios de alta capacidade participam desta eleição.Estes nós são chamados de nós capazes.Este número deve ser maior que o ideal para prover uma certa confiabilidade pois alguns nós podem parar de funcionar ou se mover para uma outra região. Para esta fase os autores propuseram um algoritmo chamado de *Random Competition based Clustering* (RCC). A idéia principal é que um nó capaz que não pertence a nenhum cluster pode iniciar a formação de um cluster enviando uma mensagem pedindo para ser um cluster-head. Os vizinhos deste nó que escutam esta mensagem desistem do direito de se tornar cluster-head e se tornam membros deste cluster. Cluster-heads periodicamente enviam estas mensagens para continuar a controlar o seu cluster.Como existe um tempo até esta mensagem chegar até todos os nós, alguns nós também podem reivindicar o direito de se tornar cluster-head simultaneamente. Para minimizar a probabilidade de colisão foi criado um contador de duração aleatória. Este contador é inicializado sempre antes de um nó reivindicar pelo seu direito de se tornar líder do cluster. Se um nó escutar uma mensagem de outro nó durante este tempo aleatório ele desiste de ser líder. No entanto mesmo assim ainda existe a chance de haver alguma colisão neste caso, o nó que tiver a menor identificação vence a disputa.

4.2.1)Cluster com múltiplos hops

No caso anterior o líder atinge todos os seus seguidores com apenas um hop. Mas se desejarmos obter o número ótimo de nós de backbone este número só é conseguido se fizermos clusters de k hops. Ajustando o k podemos controlar o número de nós de backbone. k hops significa que o líder atinge todos os seus seguidores se der um número máximo k de saltos.

Para extender o algoritmo para o caso de múltiplos saltos, cada nó deve re-encaminhar o pedido de reivindicação de liderança do seu cluster-head. Um nó vai selecionar o cluster head mais próximo dentro de escopo de k hops para ser seu líder. Se um nó não tiver nenhum líder em k hops ele pede para ser o líder após esperar um tempo aleatório. É importante notar que neste a probabilidade de haver uma colisão é bem maior devido ao retardo de propagação então o contador aleatório tem um papel fundamental nesta história.

4.3) Resultados

Os autores em [6] fizeram algumas simulações eles mediram a estabilidade dos clusters o tempo médio que cada nó permanece associado a um cluster e por último variaram o númro ótimo de BNs em função do número de hops. Os nós estão dispostos numa área de 3200x3200m, todos os nós tem rádios compatíveis com o IEEE 802.11 com alcance e 175m. A banda é de 2Mbps. O modelo de mobilidade utilizado foi o Random way point com tempo de pausa fixo igual 30 s. Comparou-se os resultados com dois algoritmos de clusterização o primeiro é o LID neste caso o nó com menor identificação é eleito líder eo segundo e o HD onde o nó com o maior número de vizinho é eleito o líder.

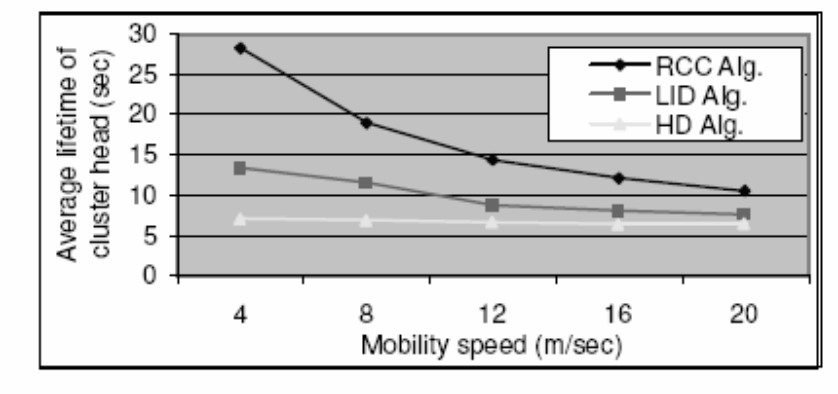


Fig 7 tempo médio de vida do cluster head

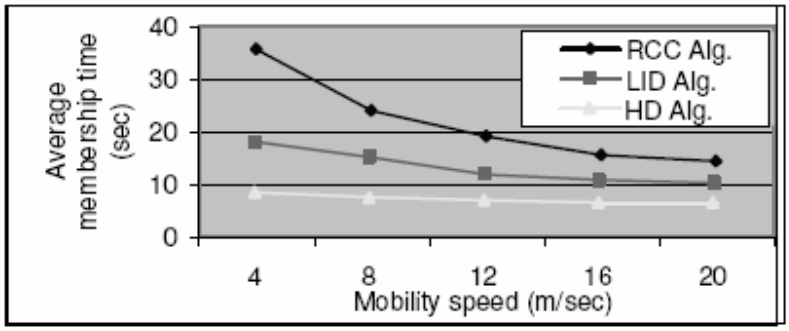


Fig 8 tempo médio de permanência de um nó em um cluster

A partir destes resultados percebe-se que o RCC apresenta um desempenho melhor, mas os resultados ainda não são muito satisfatórios

Na figura 9 a a razão entre a banda dos dois nós variou e encontrou-se o número ótimos da hops para cada razão.

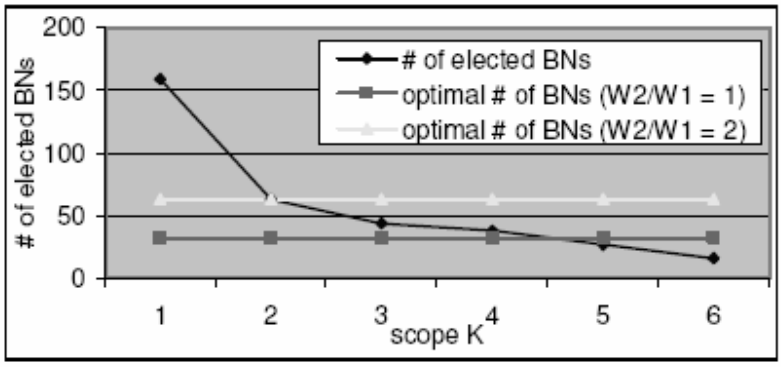


Fig 9 número de nós de backbone em função do número de hops k

5)TBone

Esta proposta de backbone foi feita por Rubin et al. [7]. Junto com esta proposta, foram idealizados mecanismos de garantia de QoS para as aplicações. Neste caso também existem nós de alta capacidade e nós de baixa capacidade(os nós de alta capacidade também devem ter dois rádios, um para o backbone outro para o cluster, e o número de nós capazes de fazer parte do backbone deve ser relativamente alto para garantir uma certa confiabilidade ao sistema. Um nó quando coberto por um backbone tem assegurado um determinado número de timeslots para se comunicar com nós pertencentes ao seu cluster ou para se comunicar através do backbone de longo alcance. Se não fizer parte de nenhum cluster, ele se comunica através de uma rede ad hoc flat convencional.Os autores dividiram este backbone móvel em três redes distintas:

Bnet: backbone
Anet:rede de acesso ao backbone
Rede flat convencional: as tradicionais redes ad hoc

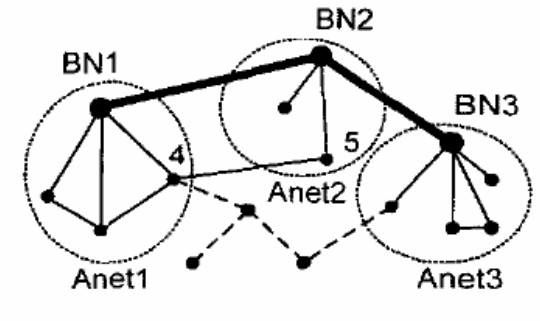


fig 10: decomposição de uma rede ad hoc em uma bnet, 3 anets e uma rede flat

Nesta proposta todos os clusters usam a mesma freqüência de transmissão, e é permitido aos nós de uma anet falarem diretamente com seus vizinhos pertencentes a outra anet,exemplo na figura xx os nós 4 e 5 são capazes de se comunicar diretamente sem passar pelo enlace BN1 BN2. Para isto acontecer é necessário uma coordenação entre as anets para não haver interferências entre elas. Estes clusters são então chamados de MC-anets (*MAC Coordinating*) e os nós de backbones que fazem esta coordenação são chamados MC-BNs.

Para que este backbone ad hoc se adapte as variações das condições da rede foram definidos dois tipos de eventos: os eventos causados por modificações topológicas (mobilidade, variação do número e nós) e/ou modificações causada por grandes flutuações do tráfego(variações causadas por diferentes requisitos de QoS ou induzidas por falhas na rede). Estas mudanças levam a violações de um ou mais atributos analisados fazendo com que o sistema tome as devidas ações para se adaptar ao novo cenário.Estes eventos são divididos em categorias distintas e cada evento leva a uma determinada ação que será explicada mais adiante.

5.1)Funcionamento do protocolo

A formação do backbone é feita em 4 etapas:

1) Eleição dos nós de backbone (só participam os nós capazes)
2) Criação dos enlaces de alta capacidade
3) Formação dos clusters
4) Formação da Anets

Uma lista local de registros controlada pelo nó de backbone contém a identidade de todos os seus nós associados de baixa capacidade. Cada nó de baixa capacidade guarda uma lista com todos os seus vizinhos (nós de backbone, nós capazes de fazer parte do backbone e nós de baixa capacidade fazem parte desta lista).Cada nó que faça parte do backbone(chamado de BN – backbone node) ou que seja capaz de fazer parte do backbone (será chamado de BCN – *backbone capable node*) guarda uma lista com seus vizinhos (atingidos pelos seus nós de alta capacidade) BN e BCN. Cada BCN periodicamente envia mensagens contendo esta tabela através de seu rádio de alta capacidade.

Os nós que fazem parte do backbone estabelecem em cada quadro, chamados de Anet frames, um determinado número de segmentos para cada um de seus nós associados. Cada nó recebe um determinado número de segmento baseado na sua necessidade. Os nós de backbone também alocam um certo número de segmentos para atender as suas necessidades. Um certo número de segmentos também é reservado para transmissões usando-se o acesso aleatório. Cada quadro enviado contém pré-âmbulos e mensagens de controle. Quadros de sincronismo são periodicamente enviados.A estrutura de um quadro está exemplificada na figura abaixo:

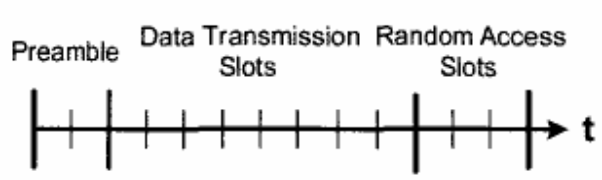


fig 11 – estrutura do quadro do Tbone

Cada nó de baixa capacidade informa dinamicamente ao seu nó de backbone sua demanda por transmissão. Para cada fluxo iniciado deve-se especificar a classe de serviço e alguns descritores de tráfego. O BN então calcula o número de segmentos necessários por quadro e se hover capacidade disponível estes segmento são alocados. A BN também deve avaliar a quantidade de recursos disponíveis tanto na Bnet quanto na Anet de destino das mensagens. Conseqüentemente para garantir a QoS fim a fim é importante perceber que a decisão de alocar ou não recursos depende da anet de origem ,depente da bnet e depende também da anet de destino.

Como todas as anet compartilham o mesmo canal de comunicação é necessário haver uma coordenação entre anets vizinhas. Estas MC-BNs devem sincronizar o processo de alocação dos segmentos para evitar interferências nas transmissões.

Existem dois tipos de colisões nas MC-anets: as permitidas que ocorrem se e somente os nós onde são detectadas as colisões não estão envolvidos nesta troca de mensagens e neste caso não há perda de pacotes. Todos os outros tipos de colisões são consideradas não permitidas.

Cada BCN armazena uma lista que é dinamicamente atualizada contendo a métricas a respeito deste nó. As métricas são: informações a respeito da mobilidade, velocidade de processamento, capacidade do backbone e da anet , tamanho da fila, atraso de pacotes e alguns indicadores de congestionamento.

Pode ser que nenhum nó faça parte Bnet, nem seja admitido em nenhuma anet, neste caso, a comunicação é feita da mesma forma que é feita em uma rede ad hoc comum.

O protocolo é constituído de 4 algoritmos acionados por eventos distintos. Foram definidas 8 classes de eventos:

Classe 1 (conectividade):Cada BCN envia uma lista contendo seus vizinhos de alta capacidade periodicamente ao seu BN. Se o BN detectar que nesta lista está incluído algum BN que não faça parte da sua lista de vizinhos de alta capacidade ele conclui que a Bnet está desconectada e então envia uma mensagem a este BCN mandando ele se tornar um BN. Inicializa o processo de conversão BN-BCN

Classe 2 (Mínima): Cada BN periodicamente compara sua lista local de registros com a dos seu BN vizinhos. Se cada nó de sua Anet tem pelo um BN na sua vizinhança e se todos os seus BN vizinhos vão se manter no mesmo estado este nó então se converte num BCN através do algoritmo de conversão BN-BCN

Classe 3 (eficiência): Se devido a mudanças na topologia e/ou a mudanças no perfil do tráfego um nó de baixa capacidade encontrar um BN vizinho que tenha um dynamic weighted label que seja pelo menos D unidades maior que a do seu atual BN este nó se considera um nó de baixa capacidade desassociado e inicia o algoritmo de associação a uma Anet controlada por este BN vizinho.

Classe 4 (cobertura dos nós de baixa capacidade): Se um nó de baixa capacidade identifica algum vizinho seu como um BN ele inicializa o algoritmo de a associação a uma Anet controlada por este BN vizinho.

Classe 5 (cobertura dos BCN) : Se um BCN identifica algum vizinho seu como um BN ele inicializa o algoritmo de a associação a uma Anet controlada por este BN vizinho

Classe 6 (cobertura dos BCN) : Se um BCN não escuta nenhuma mensagem de seu BN durante um período pré definido de tempo mas detecta outro BCN na sua vizinhança ele inicia o algoritmo de eleição do BN.

Classe 7 (cobertura dos BCN) : Se um BCN desassociado não escuta nenhuma mensagem por um período pré definido de tempo ou só escuta mensagem de nós de baixa capacidade e/ou de nós BCN associados ele inicia o algoritmo de eleição do BN. È claro que neste caso este nó vai ganhar uma vez que ele é o único nó que participa da eleição.

Classe 8 (adaptabilidade das aplicações) : Se o tráfego de alguma estação é alterado o nó pertencente ao cluster envia seus novos requisitos de QoS, perfil de tráfego ao seu BN e este então recalcula o número de segmento por quadro necessários

A seguir está representada uma figura representando os eventos e cada algoritmo que é acionado por este evento.

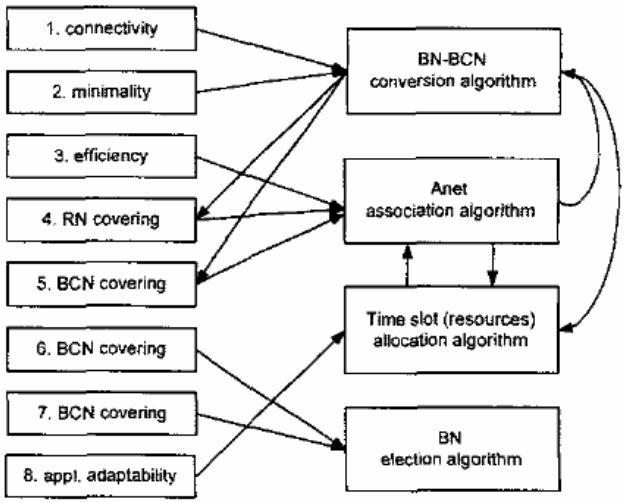


fig 12 diagrama do protocolo TBONE

5.3)Descrição dos algoritmos:

O Tbone consiste de 4 algoritmos: Algoritmo de associação a uma anet, algoritmo de eleição de líder, algoritmo de conversão BN-BCN, e algoritmo de alocação de recursos

5.3.1)Algoritmo de associação a uma anet

Este algoritmo provê mecanismos de associação de nós baseado em critérios que levam em conta tanto requisitos de QoS quanto topológicos. Um nó de baixa capacidade que escuta algum vizinho BN ou BCN escolhe aquele que tiver o maior *dynamic weighted label* e envia uma mensagem *join request* . Nesta mensagem consiste vem a identificação deste nó, a identificação do nó que se deseja juntar, a lista de seus vizinho de baixa e de alta capacidade e o número de time slots requeridos para satisfazer suas necessidades. Então ele passa por um processo de admissão onde o nó de backbone verifica se existe recursos disponíveis para satisfazer a demanda deste nó. Se houver, o nó é admitido, senão este BN verifica se alguns dos vizinhos de alta capacidade do nó de baixa capacidade tem condições de atender (isto é possível porque a troca de informação entre os nósde alta capacidade é mais completa). Se houver e se nó capaz de atender for um nó BN o nó de alta capacidade se associa a este nó, se o nó capaz de atender for um nó BCN este nó deve se tornar um BN. Se nenhum vizinho for capaz de atender, o nó de baixa capacidade fica neste processo até encontrar um nó capaz de atender as suas necessidades.

Se o nó desassociado for um BCN a seqüência é parecida, só que este neste caso o nó passa por um processo de pré admissão onde o BN verifica se a Bnet está conectada (evento de classe 3). Se não estiver este nó se é transformado diretamente em um nó negro. Se a rede estiver conectado segue os mesmos passos do nó de baixa capacidade.

5.3.2)Algoritmo de eleição do cluster head

Neste algoritmo ganha a eleição aquele nó que tiver mais recursos disponíveis. Através do dynamic weighted label sabe-se qual nó tem a maior capacidade. Sempre que ocorre um evento de classes 6 ou 7 este algoritmo é iniciado. Ele funciona da seguinte forma:

Sempre que um BCN inicia uma eleição ele começa a transmitir mensagens contendo sua identificação e seu dynamic weighted label. Cada BCN que receber esta mensagem também faz o mesmo. O nó que iniciou o processo espera um tempo pré definido para coletar informações, se durante este tempo ele ouvir uma mensagem de um BN através do seu rádio de baixa capacidade a eleição se encerra. Se não ouvir e se este nó tiver o maior dynamic weighted label ele se converte em BN, senão ele espera novamente por um período de tempo pré definido.

5.3.3)Algoritmo para fazer a conversão BN-BCN

A idéia deste algoritmo é encontrar BN redundantes e convertê-los em BCN. Se ocorrer um evento de classe 2 este algoritmo é chamado. Ele funciona da seguinte forma:

Se um BN através das trocas de mensagem entre os nós de alta capacidade perceber que ele é totalmente redundante ele se converte em BCN e o algoritmo se encerra.Se não ele procura os seus MC-BNs capazes de absorver ele e todos os seus nós de baixa capacidade. Se não houver nenhum o algoritmo se encerra, se houver ele transmite uma mensagem de associação a uma anet em seu favor e em favor de cada um de seus nós associados para um nó escolhido dentro deste conjunto. Após todos os nós serem admitidos inclusive o BN este então se converte em um BCN.

5.3.4)Algoritmo de alocação de time slots

O objetivo deste algoritmo é alocar recursos aos nós de baixa capacidade associados em função dos pedidos feitos e dos recursos disponíveis. Ele também é usado como subrotina do processo de admissão durante um pedido de admissão a uma anet.Funciona deste jeito:

Se um Bn recebe um pedido de segmentos ou um join request este Bn verifica a atual alocação de slots na sua anet. Ele também verifica a capacidade disponível tanto na bnet quanto na anet de destino. Também verifica com seus MC-BNs a possibilidade de coordenação para reduzir as interferências. Baseado nos resultados o Bn verifica se existe condição ou não de aceitar este pedido. Se houver ele aloca os recursos necessários e informa aos seus MC-BNs senão o pedido é negado.

6)Comentários e propostas

É fato que a criação de backbones surge como a grande solução para tornar as redes Ad Hoc escaláveis. O problema é que surge um trade off entre a complexidade para criar e/ou manter estes backbone. Diversas variáveis extras são criadas, trocas de mensagens adicionais são requeridas. Esta troca de mensagens não é tão grande no Tbone e no MBN porque os nós capazes encontram-se em número reduzido. Por outro lado isto também torna mais árdua a tarefa de conectar a toda a rede.

No caso de VDBP a principal melhora é a redução do número de nós fazendo o papel de roteadores. O overhead causado pelo controle do backbone pode ser reduzido se as mensagens de controle forem transmitidas junto com os pacotes de roteamento. Neste caso devido a natureza periódica das mensagens de controle, protocolos de roteamento pró-ativos são os mais indicados.

A proposta de Xu é bem simples, pretende aumentar a capacidade de uma rede ad hoc simplesmente dividindo esta rede em subredes. O número ótimo de subredes é encontrado em função do número de hops de cada cluster. Gerla et al. em [8] argumenta que ao invés de temer a mobilidade você deve explorá-la. Ele acredita (o que é bem sensato) que os nós tendem a andar em grupos (ex. uma tropa militar) e por isso deve-se explorar este efeito através de novos protocolos de roteamento. Ele propõe um protocolo de roteamento chamado LANMAR. Este protocolo utiliza o conceito de landmarks para encontrar os grupos. Um protocolo de vetor de distâncias propaga informações sobre os landmarks da rede. É óbvio que um clusterhead pode ser tratado exatamente como um Landmark. Isto aumentaria a eficiência do backbone. O algoritmo de clusterização utilizado não garante clusters realmente estáveis.Gerla neste mesmo artigo propõe um algoritmo que usa o grau de afinidade entre os nós como métrica para eleger o líder, o VDBP também usa esta idéia quando escolhe o NFLL como métrica.

Rubin já fez uma proposta mais complexa onde os líderes são escolhidos baseados em sua capacidade de processamento, indicadores de congestionamento da rede de acesso, ou seja na capacidade que o nó de backbone tem de atender as necessidades de um determinado nó.O protocolo de acesso ao meio é bastante complexo requer sincronismo entre os de uma mesma rede de acesso e requer sincronismo entre nós de redes de acesso vizinhas! Estudar o Tbone com um protocolo de polling como controle de acesso ao meio traria grandes benefícios e aparentemente simplificaria bastante as coisas. Para implementar cluster de k hops outros protocolos de controle de acesso ao meio seriam necessários. Usar o protocolo de roteamento como o LANMAR no TBONE também seria uma boa solução pelos mesmos motivos do MBN.

7)Conclusão

Estes protocolos de backbone são bastante interessantes e vão ter papel fundamental no desenvolvemento das redes Ad Hoc de alta capacidade. As propostas são recentes mas se mostraram bastante interessantes. Ainda resta muito campo para se trabalhar.

Referências:

[1] X. Hong, K. Xu, M.Gerla."Scalable Routing Protocols for Mobile ad Hoc Networks", IEEE Network, Junho 2002, pag. 11 a 21.

[2] J. Kurose, K. Ross."Computer Networking: a Top-Down Approach featuring the Internet" Addison-Wesley, 2001.

[3] U.C.Kozat, G.Kondylis, B.Ryu, M. Karina. "Virtual Dynamic Backbone for Mobile Ad Hoc Networks" IEEE International Conference on Communications, 2001. ICC2001.,Volume:1,11-14,June2001,Pgs: 250 -255 vol.1

[4] S.R.Das, C.E.Perkins, E.M.Royer. "Perfomance Comparison of Two On-Demand Routing Protocols" Proc. IEEE INFOCOM 2000, Tel Aviv, Israel, Março 2000

[5] P.Gupta, P.R.Kumar. "The Capacity of Wireless Networks" IEEE Trans. On IT, vol IT-46, no2 ,pag. 388-404, Março 2000

[6] K. Xu, X. Hong, M. Gerla. "An Ad Hoc Network with Mobile Backbone" proc. IEEE ICC 2002, Nova York, EUA, abril 2002

[7] I.Rubin, A.Behzad, R.Zhang, H.Luo,E. Caballero."TBONE:A Mobile Backbone Protocol for Ad Hoc Wireless Networks" Aerospace Conference Proceedings, 2002. IEEE , Volume: 6 , 9-16 Março 2002

[8] M. *Gerla.,K. Xu, X. Hong.* "Exploiting mobility in large scale ad hoc wireless networks**"** Computer Communications, 2003. CCW 2003. Proceedings. 2003 IEEE 18th Annual Workshop on, 20-21 Outubro.2003 Pgs: 34 -39

[9] C. Perkins "Ad Hoc Networking", Addison Wesley, 2001

[10] I. Rubin and R. Zhang, "Performance behavior of unmanned vehicle aided mobile backbone based wireless ad hoc network", t *in Proceedings of IEEE Vehicular Technology Conference- VTC 2003,* Jeju, Korea, Abril 22-25, 2003