3
Problemas abordados nesse trabalho e Implementação

O protocolo de roteamento Terminode utiliza a combinação de um protocolo de roteamento geográfico, TRR, com um protocolo de roteamento do tipo *link state*, o TLR. O objetivo é atingir escalabilidade, suportando redes *ad hoc* que cobrem áreas geográficas grandes e com centenas de nós. Além da mobilidade dos nós, existem outros problemas que afetam o roteamento baseado em localização: a existência de áreas de exclusão na topologia de rede, e a desativação freqüente de nós para fins de economia de bateria.

Desse modo, para analisar se o protocolo é capaz de atingir seus objetivos, ele deve ser testado em topologias de redes irregulares, com algumas centenas de nós móveis. Para testar a eficiência do protocolo de roteamento terminode, deve-se compará-lo com os protocolos de roteamento tradicionais existentes para rede *ad hoc*.

Os testes encontrados na literatura [2, 4, 5 e 6], apresentam um bom desempenho do protocolo de roteamento terminode em relação a taxa de pacotes entregues com sucesso, atraso médio da entrega de pacotes, quando comparados aos protocolos tradicionais utilizados em redes *ad hoc* operando em área geográfica restrita. Quando a rede apresenta uma cobertura geográfica extensa, o método de roteamento terminode supera os protocolos de roteamento tradicionais testados. Essa característica positiva fez desse protocolo o alvo do estudo nesse trabalho.

Os testes do protocolo em redes com áreas geográficas extensas foram realizados utilizando um novo modelo de mobilidade, denominado de *restricted random waypoint*. Como esse modelo foi a base dos testes do protocolo, ele será descrito a seguir.

3.1.
Restricted random waypoint

Este modelo de mobilidade procura refletir a realidade da rede de forma mais adequada, onde os terminodes são aparelhos móveis pessoais geograficamente distribuídos em uma área de grande extensão. Nessa rede é pouco provável que, para cada movimento, o terminode selecione um destino aleatório dentro uma área geográfica muito extensa, realizando freqüentemente movimentos de longa distância. Ao contrário, é mais provável que o terminode se movimente algumas vezes dentro de uma área geográfica restrita e então realize um movimento de uma longa distância [6].

Para este modelo será usada uma topologia baseada em cidades e estradas. Cidades são áreas que são conectadas pelas estradas, conforme ilustrado na figura 15. Dentro das cidades, os terminodes se movem de acordo com o modelo de mobilidade *random waypoint*. Após um certo número de movimentos dentro de uma mesma cidade, o terminode se move para uma outra cidade. Terminodes que estão se movendo entre cidades simulam o deslocamento em estradas. Devem ser definidas quais cidades são conectadas por quais estradas, permitindo a viagem direta dos terminodes entre essas cidades. No exemplo da figura 15, a cidades 3 e 4 não possuem estradas entre si e portanto os nós não podem se locomover diretamente entre essas cidades.

O modelo de mobilidade *random waypoint* é amplamente utilizado na simulação de redes móveis e é o modelo usado para descrever a mobilidade dentro de uma mesma cidade. Neste modelo, um nó escolhe um destino aleatório dentro da área de simulação. Então o nó se move com velocidade constante, mas aleatória. Após chegar ao destino, o nó fica parado por um período, e etnão seleciona um novo destino. Para cada novo destino o mesmo procedimento é repetido [6].

A mobilidade dos terminodes depende da sua distribuição geográfica inicial e do modelo de mobilidade para cada terminode. Como visto anteriormente, o

modelo *restricted randon waypoint* utiliza uma divisão da área de simulação entre cidades e estradas. Então o posicionamento inicial dos terminodes deve obedecer a essa divisão. A forma inicial de distribuição dos nós é feita por um modelo, escolhido como um parâmetro do cenário de simulação.

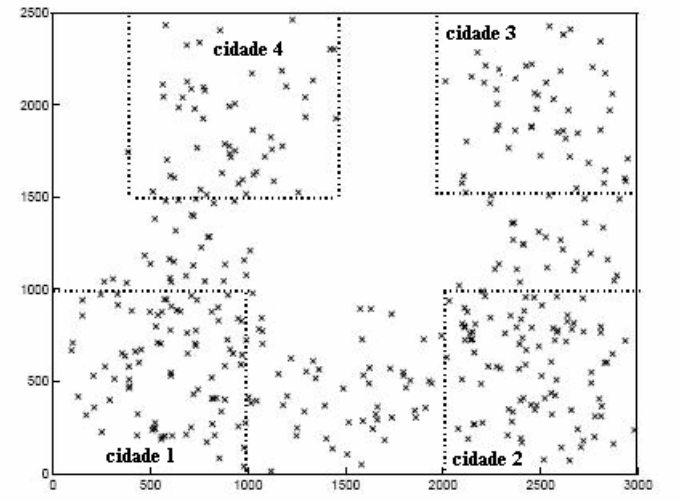

Figura 15 – restricted random waypoint

No começo da simulação, os terminodes são distribuídos em posições aleatoriamente escolhidas em cada uma das quatro cidades. Os terminodes são de três tipos: terminode normal, terminode estacionário e terminode viajante. Um terminode estacionário não se move, e tem probabilidade maior de ficar parado em cidades, mas podem estar nas estradas também. Um terminode normal começa a simulação selecionando um destino na mesma cidade onde se encontra, e então ele se move para aquele destino com velocidade fixa. Após chegar ao destino, o terminode normal fica parado durante o valor definido pelo parâmetro *pause time*, seleciona um outro destino ainda na mesma cidade e repete o procedimento de deslocamento. O movimento do terminode normal dentro da cidade é o que é denominado de *random waypoint*. Ele repete esses movimentos por um número de vezes descrito pelo parâmetro *stay_in_town*. Enquanto o parâmetro *stay_in_town* for maior que 1, o terminode escolhe um novo destino na mesma cidade. Após atingir esse número, o terminode seleciona um destino aleatório dentro de uma nova cidade e se move para lá. A nova cidade é selecionada aleatoriamente dentro de uma lista de cidades que são conectadas à cidade atual por uma estrada. Ao chegar na nova cidade selecionada, o terminode normal volta a seguir o modelo

random waypoint por outros *stay_in_town* movimentos. Os terminodes comuns utlizam o parâmetro *stay_in_town* igual a 2 e o *pause_time* variável [4].

Alguns terminodes são selecionados para viajar com freqüência de uma cidade para outra. Estes terminodes são chamados viajantes e eles asseguram conectividade entre cidades. O movimento desses terminodes é denominado de *restricted random way* com parâmetros *stay_in_town* igual a 1 e *pause_time* igual a 1 segundo. Eles selecionam um destino aleatório dentro da cidade e se movem com velocidade uniforme. Ao atingir seu destino, os terminodes ficam parados por um tempo, que é menor que os terminodes ordinários. Então ele seleciona uma outra cidade que seja conectada à cidade atual, seleciona um destino dentro da nova cidade e se move para lá. Ele fica parado por um pequeno intervalo na nova cidade e se move de novo para uma outra cidade [6].

3.2. Problemas abordados nesse trabalho

Nas avaliações do protocolo de roteamento encontradas na literatura [2, 4, 5 e 6], o modelo de mobilidade *restricted random waypoint* é também o responsável por gerar as adversidades na topologia da rede. Para analisar o desempenho do protocolo de roteamento em relação ao problema da mobilidade dos nós, esse modelo é completamente útil. Porém, existem outros problemas característicos das redes *ad hoc* que devem ser avaliados, e para isso será necessário modificações no modelo de mobilidade e o desenvolvimento de alguns novos mecanismos de simulação, descritos mais adiante. Isto irá permitir analisar como o protocolo de roteamento reage a alguns problemas específicos na topologia de rede e testar sua robustez.

O objetivo deste trabalho é o de testar o desempenho deste sistema frente as seguintes situações: terminodes sendo ativados e desativados em instantes aleatórios, existência de áreas de exclusão e erros de localização. Para isso será necessário desenvolver métodos específicos que permitam avaliar o desempenho do protocolo de roteamento nessas condições, que são a seguir detalhadas.

3.2.1.
Ativação e desativação aleatória de terminodes

Em uma rede auto-organizável freqüentemente são adicionados ou removidos alguns nós dessa rede. Isto ocorre de forma não controlada, e o protocolo de roteamento deve estar preparado para lidar com isso. Essa situação pode ser modelada como terminodes sendo desligados e religados em instantes aleatórios.

Quando um terminode desliga (para economizar bateria, por exemplo), ele deve parar de transmitir e receber qualquer sinal do canal e também parar de gerar qualquer tipo de dados. Porém o terminode continua se movendo de acordo com o modelo de mobilidade escolhido. Os terminodes desligam de forma independente e sem avisar aos outros terminodes. Isto causa problemas de roteamento, pois um terminode que se desliga possivelmente participa de algumas rotas de roteamento. Logo, ao se desligar, um terminode pode tornar alguns outros terminodes inatingíveis, ou diminuir o número ou qualidade das rotas até esses terminodes. Sendo esse um fato comum, é necessário avaliar como o protocolo de roteamento se comporta diante dessa situação.

A situação complementar a essa é a que acontece quando terminodes ligam em um instante aleatório, com a rede já em funcionamento. Nesse caso, novas rotas podem ser criadas, ou no pior caso pode acontecer interferência em rotas já estabelecidas, causando perda de eficiência do protocolo de roteamento.

Ao interromper a recepção, o terminode perde todos os dados a ele direcionados, incluindo os pacotes de controle. Dependendo do tempo que o terminode fica desligado, todas as informações obtidas da rede que precisam ser constantemente atualizadas são perdidas. Portanto, o terminode pode perder as informações contidas na sua tabela TLR (que informa quais são seus vizinhos de um e de dois saltos) e pode perder a conexão com terminodes amigos (FAPD *responders*), além de perder as informações de caminhos com âncoras e as informações de mapeamento de LDA e EUI para terminodes conhecidos.

Ao interromper a transmissão, o terminode além de não enviar pacotes de dados para rede, deixa de enviar os pacotes de controle, como o pacote HELLO do protocolo de roteamento. Assim ele será apagado da tabela TLR de seus vizinhos e de qualquer rota que ele participar após algum tempo sem transmitir. Em uma área muito densa, um terminode desligar pode ter efeito positivo, pois diminui a interferência no meio de transmissão. Porém, em uma área pouco densa esse terminode pode ser essencial para a construção de algumas rotas, e portanto seu desligamento pode resultar em várias rotas inexistentes e grande quantidade de pacotes perdidos.

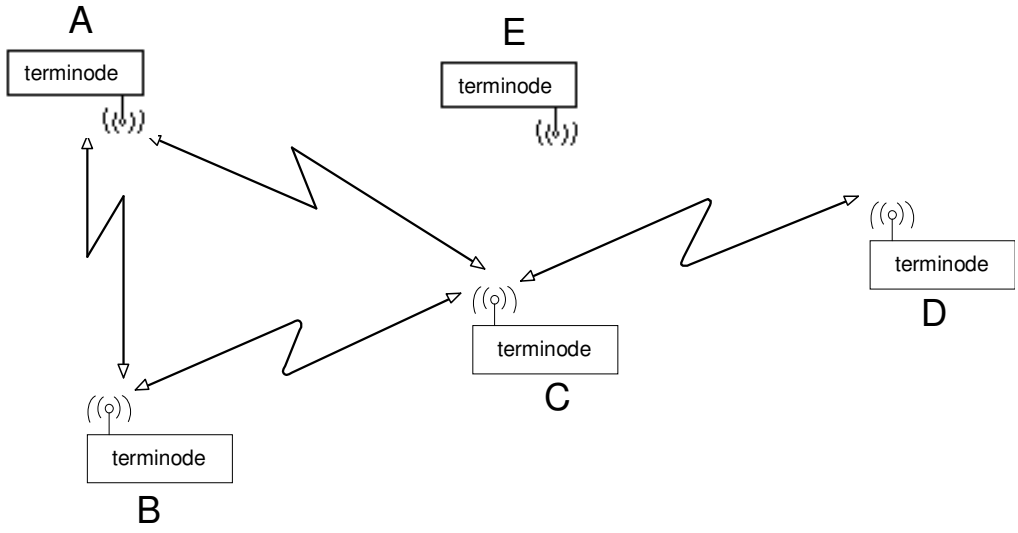

Figura 16 – Situação onde pode ocorrer perda de rotas se um terminode desligar

A figura 16 apresenta um exemplo em que o desligamento de um terminode pode ocasionar perda de rotas. No exemplo, os terminodes A e B só podem alcançar o terminode D através de rotas que passem pelo terminode C. Portanto, se o terminode C se desligar por qualquer razão, o terminode D se torna inalcançável para A e B. Em uma rede mais densa, possivelmente devem existir outras rotas. Um dos objetivos do presente trabalho é avaliar a capacidade do protocolo encontrar rotas alternativas nesse tipo de rede. O terminode E que se encontra inicialmente desligado, pode em algum momento se religar a rede. Isso irá criar novas rotas na rede, provocando o aumento da interferência na área que ele se localiza.

3.2.2.
Áreas de exclusão de terminodes

Um problema para redes distribuídas sobre uma área geográfica muito extensa é a possibilidade de existência de áreas problemáticas, onde os terminodes são impedidos de se locomover por elas. Essas áreas ocorrem devido à existência de alguma barreira geográfica, como por exemplo, uma lagoa. Isto causa "buracos" na topologia que podem dificultar a descoberta de rotas. Esses buracos forçados na topologia formam uma área de exclusão de terminodes, que são chamadas de desertos. Dessa forma, é necessário contornar essas áreas para prover forma de roteamento de pacotes.

O modelo de mobilidade *restricted randon waypoint* permite criar barreiras de mobilidades entre as cidades. Por exemplo, a figura 15 mostra a existência de uma barreira de mobilidade entre as cidades 3 e 4. Porém, essa forma de bloqueio de mobilidade é muito simples, e não permite definir um deserto de terminodes em alguma área específica e limitada. O deserto pode se localizar dentro de uma das cidades ou no caminho entre elas.

3.2.3.
Erros de localização

Uma avaliação de desempenho importante é que diz respeito a imprecisão no sistema de localização e seus efeitos no protocolo de roteamento. Essa avaliação é importante porque o protocolo depende fortemente da informação da posição para realizar o roteamento em longa distância. Por exemplo, o projeto Terminodes desenvolveu o método *virtual home region* (VHR) para o sistema de gerência de mobilidade. Uma das funções desse sistema é a localização de um terminode a partir de seu endereço permanente. Porém, essa informação pode chegar de maneira incorreta até o terminode a que requisitou. O protocolo foi projetado para possuir uma robustez aos erros de localização utilizando um roteamento local (TLR) e o método de busca RLF em complemento ao roteamento geográfico, conforme descrito no capítulo anterior. Conhecer a

robustez do protocolo permite avaliar quais métodos de localização podem ser usados em uma rede terminode sem prejuízo de seu desempenho.

Por meio da medida da tolerância do protocolo aos diversos valores de erros, é possível escolher para cada situação um sistema de posicionamento mais adequado. Assim é possível avaliar a necessidade do uso de um sistema de gerência de mobilidade mais ou menos robusto, e determinar seu custo em termos do número de mensagens de controle geradas.

3.3. Método de simulação

Para testar o desempenho de uma rede móvel *ad hoc* de acordo com o projeto Terminode foi usado o simulador GloMoSim [21]. Esse tem sido o simulador utilizado para os testes do protocolo de roteamento terminode e também do modelo de mobilidade nas referências [2, 4, 5 e 6]. Isso permite utilizar os protocolos aqui discutidos e que foram incorporados ao simulador e comparar os resultados obtidos com os encontrados na literatura.

O GloMoSim é um ambiente de simulação escalável para redes sem fio. O GloMoSim utiliza o Parsec (uma linguagem de simulação baseada em C) [22] para fornecer capacidade de simulação discreta a situações que envolvam eventos paralelos. Ele tem a capacidade de simular redes com até mil nós, que podem ser conectados por enlaces heterogêneos. Sua capacidade inclui o suporte para redes ad hoc sem fio e os protocolos Internet, assim como modelos de mobilidade dos nós.

Uma vantagem desse simulador é sua construção em camadas similares às utilizadas no modelo OSI. Para a troca de mensagens entre camadas são utilizadas APIs padronizadas, permitindo uma rápida integração de modelos desenvolvidos em diferentes camadas. Essa característica do GloMoSim permite adicionar novos modelos desenvolvidos para testes de novos protocolos, fazer alterações pontuais em algumas camadas do simulador. Os protocolos são escritos basicamente em

linguagem C com algumas funções Parsec para controlar o tempo da ocorrência de eventos [21].

O GloMoSim possui uma variedade de protocolos e algoritmos para todas as camadas, dando uma grande flexibilidade à simulação. Os parâmetros escolhidos para o cenário de simulação são passados para o simulador por meio de um arquivo de configuração. O Anexo 1 mostra algumas características do GloMoSim, e que ajudam a entender as soluções desenvolvidas.

O protocolo de roteamento é dependente dos sistemas de posicionamento e de localização. A flexibilidade do GloMoSim permite isolar cada uma dessas funcionalidades, gerando resultados mais precisos do desempenho do protocolo de roteamento. Para isso, nas simulações é utilizado um sistema de posicionamento ideal (ou seja, cada nó possui a informação exata da sua localização), e um sistema de localização que não gera pacotes de controle na rede. Dessa forma, os erros que esses sistemas geram para o protocolo de roteamento podem ser controlados e medidos.

3.3.1. Método para comutação dos terminodes

Apesar de ser comum o processo de ativação ou desativação de um terminode com a rede em funcionamento, essa situação não estava originalmente prevista no simulador. Portanto, foi necessário fazer pequenas alterações no seu código para permitir simular essa condição.

A forma mais simples de desligar um terminode é simplesmente desligar o seu rádio. Isso foi feito atribuindo um ganho muito baixo para a antena do terminode, o que simula uma desconexão entre a antena e o restante do equipamento de rádio. Dessa forma, é interrompida qualquer possibilidade de transmissão e recepção de qualquer sinal pelo terminode. Além de desligar o rádio do terminode em instantes aleatórios, é necessário interromper a geração de dados na camada de roteamento e de aplicação.

Um dos parâmetros do rádio no modelo usado pelo simulador é o ganho da antena utilizado e que pode ser facilmente acessado. Para desligar o rádio foi usado um ganho de -50 dB na antena. Com isso pode se garantir que nenhum sinal será recebido ou transmitido.

A duração dos tempos em que os terminodes ficam respectivamente ligados ou desligados são tratados como duas variáveis aleatórias com distribuições exponenciais. O ciclo de trabalho do terminode (a relação entre o tempo que o terminode fica ligado e o tempo que ele fica desligado) é controlado através das médias dessas variáveis aleatórias.

O procedimento de desligar e ligar o rádio de um terminode é dividido em duas partes: inicialização e mudança periódica do estado do terminode. Nesse contexto, o estado do terminode pode ser ligado ou desligado. As duas mudanças serão feitas na parte principal de controle do simulador, em *GLOMOPartition*, que é definido no arquivo *glomo.pc.*

A inicialização individual de cada terminode é feita após a inicialização de todas as camadas. Todos os terminodes estão inicialmente ligados. Na inicialização é definido se o terminode ficará mudando o seu estado, e quando ele será desligado pela primeira vez. Os seguintes passos descrevem esse procedimento:

1. Define-se uma porcentagem do total de nós que mudarão de estado, e sorteia-se aleatoriamente quais nós que terão essa capacidade.
2. Se o terminode for selecionado para mudar de estado, essa informação é armazenada em uma variável de mudança de estado para cada nó.
3. Escolhe-se um tempo aleatório para a primeira mudança de estado.

Após a inicialização de cada terminode, o simulador fica aguardando a ocorrência de eventos. Em um instante da simulação será verificado, em ordem:

1. Eventos de mobilidade para todos os terminodes.
2. Mudança de estado para todos os terminodes.
3. Ocorrência de outros eventos para cada terminode.

A mudança de estado do terminode deve ser feita antes de executar os eventos de envio de dados agendados para aquele instante. Isso evita que mensagens sejam enviadas ou recebidas pelo terminode após ter o seu rádio desligado.

- Verifica se o nó deve mudar de estado.
- Verifica se é o tempo agendado para a mudança.
- Muda o estado e agenda nova mudança.

Para desligar os geradores de dados do terminode, não podem ser agendados novos eventos na camada de roteamento e na camada de aplicação. Isso evita o envio de novos pacotes de controle de roteamento e pacotes de dados da aplicação por esse terminode. Esses pacotes não seriam transmitidos, mas alterariam os dados das estatísticas de dados enviados. Testando o estado atual do terminode é possível evitar que novos eventos sejam agendados para um terminode que se encontra desligado.

3.3.2. Determinação de áreas de desertos de terminodes

Para definir os desertos, é necessário impedir que os terminodes circulem ou fiquem parados sobre as áreas de deserto. Isto é feito definindo um novo modelo de mobilidade, baseado no *restricted randon waypoint*. A escolha de basear o novo modelo de mobilidade no *restricted randon waypoint* é interessante por motivos de comparação de resultados, e para aproveitar as vantagens e características já citadas desse modelo. O novo modelo de mobilidade será chamado de *deserts*. A principal característica desse modelo é a presença da área de deserto, de modo que os terminodes não possam circular ou parar sobre essa área. Algumas alterações foram feitas em relação ao *restricted randon waypoint*:

- A função de distribuição inicial dos terminodes sobre a área de simulação deve ser mudada, pois existem terminodes que serão impedidos de se localizar dentro da área de deserto.
- O modelo de mobilidade deve impedir que o destino final do terminode se localize dentro da área de deserto, e que a trajetória do terminode até o destino atravesse o deserto.

A figura 17 ilustra o cenário onde o novo modelo de mobilidade será empregado, onde existem quatro cidades, e apenas um deserto é definido entre as cidades 1 e 2.

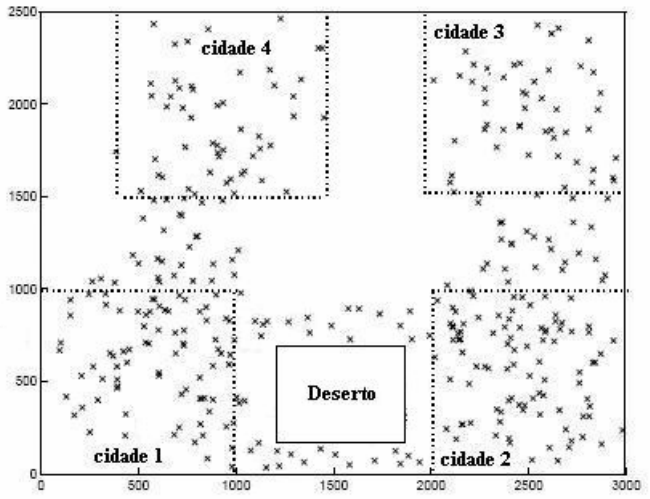

Figura 17 – novo modelo de mobilidade

Este novo modelo que estabelece a localização inicial dos terminodes é implementado por meio de alterações em dois arquivos, *driver.pc,* onde o modelo é usado, e *nodes.pc*, onde o modelo é definido. Esse modelo será chamado de *generate-deserts* e definido da seguinte maneira:

- Define-se a posição geográfica das cidades e quais cidades são vizinhas.
- Define-se a posição geográfica dos desertos.
- Gera-se de maneira aleatória uma posição e a cidade inicial, em que o terminode ser localiza. Se essa posição estiver dentro da área de deserto, uma nova posição inicial é gerada.

O modelo de mobilidade utilizado é definido por meio de um parâmetro de simulação e seu uso deve ser indicado no simulador. Para isso são necessárias as seguintes modificações no simulador:

- Modificar o arquivo *mobility.pc*, e incluir a constante (DESERTS) relativa ao novo modelo de mobilidade, nas funções necessárias.
- Definir as funções do novo modelo de mobilidade, o que foi feito em um novo arquivo chamado *deserts.pc*.

O novo modelo de mobilidade é definido de maneira muito semelhante ao *restricted randon waypoint*, ou seja, o modelo consiste em dividir a área de simulação em cidades que serão ligadas por estradas. Dentro das cidades, os terminodes se movem de acordo com o modelo de mobilidade *random waypoint*. Após um certo número de movimentos dentro de uma mesma cidade, o terminode se move para outra cidade. Os terminodes são de três tipos terminode normal (se move algumas vezes dentro da mesma cidade, antes de escolher uma nova cidade), terminode estacionário (não se move) e terminode viajante (se move constantemente entre as cidades). As tabelas 1 e 2 definem os parâmetros do novo modelo de mobilidade, sendo a única diferença para o *restricted randon waypoint* a definição dos desertos. Os parâmetros de projeto são definidos em tempo de compilação e os parâmetros configuráveis são definidos no arquivo de configuração.

Um terminode não pode ter seu destino final localizado dentro da área de deserto, e sua trajetória até o destino não pode atravessar o deserto. Para isso duas alterações significativas devem ser feitas em relação ao *restricted randon waypoint*: o método do cálculo do destino e o desvio de trajetória, que devem prever e evitar essas possibilidades.

A primeira alteração é efetuada de maneira simples. Após gerar aleatoriamente um novo destino para o terminode, verifica-se se esse ponto pertence a área do deserto. Em caso afirmativo, um novo destino é gerado, até que ele se localize fora da área de deserto.

Tabela 3 – parâmetros de projeto do novo método de mobilidade

Parâmetros de projeto	*Descrição*
Número de nós estacionários	Número de nós que não se movem, podendo se localizar em qualquer uma das cidades.
Número de nós comuns	Número de nós que realizam a maior parte de seus movimentos dentro da área de uma mesma cidade.
Número de nós viajantes	Nós em constante movimentação entre cidades. Responsáveis maiores pela existência das estradas.
Definição da localização das cidades	Isso é feito definindo as coordenadas do centro geográfico de cada cidade.
Definição da existência de estradas entre as cidades	Isso é feito definindo quais são as cidades vizinhas.
Definição da localização dos desertos.	Isso é feito definindo as coordenadas dos quatro vértices do deserto.
stay_in_town	Número de movimentos dentro da mesma cidade. Para os nós viajantes será igual a 0.

Tabela 4 – parâmetros configuráveis do novo método de mobilidade

Parâmetros definidos por arquivo de configuração	*Descrição*
TRANS_MOBILITY-WP-PAUSE	Tempo de pausa dos nós viajantes.
MOBILITY-WP-PAUSE	Tempo de pausa dos nós comuns.
TRANS_MOBILITY-WP-MIN-SPEED TRANS_MOBILITY-WP-MAX-SPEED	Velocidades máximas e mínimas dos nós viajantes em m/s.
MOBILITY-WP-MIN-SPEED MOBILITY-WP-MAX-SPEED	Velocidades máximas e mínimas dos nós comuns em m/s.

Para resolver a segunda questão é necessário desenvolver uma forma de verificar se uma trajetória em linha reta (utilizada no modelo *restricted randon waypoint*) atravessa ou não a área do deserto. Além disso, caso essa trajetória

atravesse o deserto, ela precisa convenientemente ser alterada. Um algoritmo de contorno da área de deserto foi desenvolvido para realizar a alteração da trajetória. Por simplicidade, os desertos serão retangulares e seus lados paralelos aos limites da área de simulação, conforme mostrado na figura anterior. Isso não obscurece os objetivos a serem testados, e os algoritmos apresentados podem modificados para aceitar desertor de forma arbitrária.

Um método simples de verificar se a trajetória de um terminode atravessa a área do deserto, pode ser obtido utilizando o segmento de reta que define a trajetória e os segmentos de retas que formam as fronteiras do deserto. Dessa maneira, o método se reduz em verificar se o segmento de reta da trajetória se cruza com algum dos segmentos de reta que formam o deserto. Na figura seguinte, a linha tracejada ligando os pontos S e D representa a trajetória idealizada pelo terminode. Como pode ser visto, essa trajetória cruza os segmentos de reta ad e cd, que formam o deserto. Portanto essa trajetória é alterada para contornar o deserto, conforme indicada na figura 18.

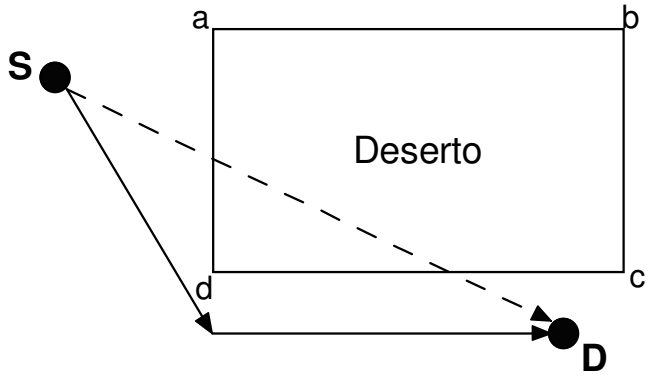

Figura 18 – algoritmo básico de contorno do deserto

O método de contorno escolhido visa minimizar a distância percorrida para contornar o deserto e é implementado dividindo-se a trajetória até o destino em trechos intermediários retilíneos. O algoritmo verifica qual lado do quadrado é cortado primeiro. A trajetória é desviada para um ponto intermediário na proximidade de uma das arestas do deserto, evitando-se assim atravessar o deserto. Ao chegar nesse ponto intermediário, o terminode procura uma trajetória em linha reta para atingir o destino. Esse procedimento é ilustrado na figura 18, onde o terminode se move do ponto S para o ponto D. A trajetória ideal é

mostrada pela linha tracejada. Para contornar o deserto, a trajetória é desviada para um ponto na proximidade da aresta d do deserto. No ponto intermediário, o terminode consegue atingir o seu destino.

Em casos mais extremos, o terminode não consegue atingir o destino desviando-se apenas uma vez sua trajetória original. Um exemplo é mostrado na figura 19. A trajetória ideal do terminode se movendo de S para D é indicado pela linha tracejada. O algoritmo aplicado é mesmo do caso anterior. A trajetória ideal se cruza primeiro com o lado *ad* do deserto, sendo desviada para um ponto intermediário I1 próximo da aresta *a*. Nesse ponto, o terminode busca um caminho em linha reta para D, indicado pela linha pontilhada. O destino ainda não pode ser atingido sem atravessar o deserto. Então o mesmo algoritmo é aplicado novamente, com I1 como o ponto de origem do terminode. A trajetória é desviada para um segundo intermediário I2 na próxima da aresta *b*. Em I2, o destino pode ser atingido por uma trajetória em linha reta.

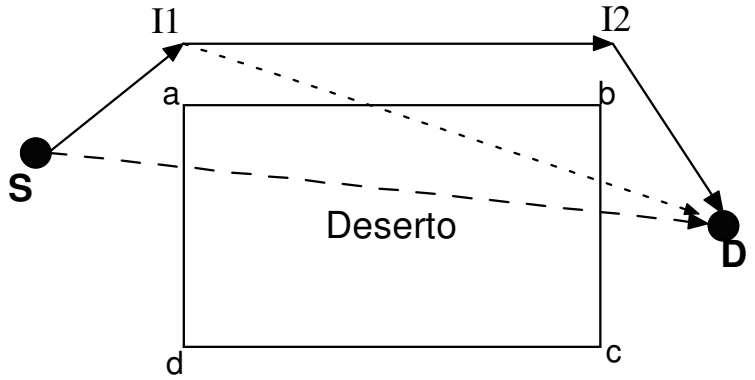

Figura 19 – situação mais complexa para o contorno do deserto

Esse algoritmo é uma forma simplificada de contorno do deserto. Existe uma tendência de concentração de terminodes ao redor das arestas do deserto. Por isso, os pontos intermediários são escolhidos de forma aleatória dentro de uma pequena área na proximidade das arestas, o que diminui um pouco esse efeito. Essa área poderia ser aumentada para diluir ainda mais a distribuição dos terminodes nesses pontos.

Os procedimentos do novo modelo de mobilidade são implementados nas seguintes funções:

- A função *deserts_mobilityInit()*, é a que inicializa o modelo de mobilidade para cada nó. Define a localização das cidades e dos desertos. Configura os parâmetros mostrados nas tabelas 1 e 2 para cada nó. Define quais são os nós comuns e viajantes, através dos parâmetros configurados para cada nó.

- A função *deserts_mobility()* é a responsável pelos eventos de mobilidade para cada nó. Eventos de mobilidade são executados regularmente e essa função deve fornecer sempre a próxima localização do nó. Para isso realiza as seguintes etapas:

    1. Utilizando as funções *DesertFindNextPos()*, e *DesertFindNextRestrictedPos(),* calcula-se uma nova posição de destino (*Dx,Dy*) para o nó, evitando as áreas de desertos. A primeira função calcula um novo destino dentro da mesma cidade, e a segunda calcula um destino em uma nova cidade.

    2. Com o novo destino calculado para cada nó, deve ser calculada uma trajetória que não atravesse o deserto. Isso feito utilizando o algoritmo descrito anteriormente. A função *findPathToDestination()* calcula os pontos intermediários (*Ix,Iy*) que devem ser percorridos para contornar o deserto.

    3. A cada novo evento de mobilidade, a função fornece um novo ponto (*dx,dy*) geográfico para o qual o nó deve se mover para chegar até o ponto intermediário (*Ix,Iy*) fornecido por *findPathToDestination()*. Os pontos sucessivos (*dx,dy*) procuram formar uma trajetória em linha reta até (*Ix,Iy*). Ao atingir o ponto intermediário (*Ix,Iy*), a função *findPathToDestination()* calcula o próximo ponto intermediário (*Ix,Iy*) necessário para atingir o destino. O último ponto fornecido por esta função será o próprio destino final (*Dx,Dy*). Ao chegar no destino, o passo 1 é repetido.

3.3.3. Erros no serviço de localização

Este teste pode ser realizado de maneira simples porque as implementações do serviço de localização, do sistema de posicionamento e do protocolo de roteamento são realizadas de forma independente. Na implementação atual do protocolo de roteamento, o sistema de posicionamento utilizado é o GPS, com erro de posição igual a zero. O serviço de localização (também chamado de sistema de gerência de mobilidade) utilizado não gera pacotes de controle na rede.

Desse modo, o uso de apenas uma função de mapeamento entre o endereço permanente EUI e a localização (LDA) do terminode se faz necessário. Essa função usada pelo protocolo de roteamento é chamada de *GPS_Locate* e será modificada para os testes de erro de localização.

Na implementação atual da função *GPS_Locate*, ela retorna a posição precisa de um terminode, utilizando apenas o endereço EUI para isso. Essa função foi modificada para que se acrescente um erro aleatório na posição do terminode, gerando uma imprecisão na informação solicitada. Foi criada uma nova função chamada *node_Locate( )*, que é usada pelo protocolo para localizar um nó a partir de seu endereço.

O erro que é adicionado possui uma distribuição gaussiana. A variância do erro é aumentada de forma gradual, permitindo-se calcular as estatísticas de desempenho do protocolo no que se refere a erros de localização.

A função *node_Locate( )* executa os passos descritos abaixo:

1. Recebe como parâmetros: ponteiro para o nó que requisitou a localização e EUI_D do nó a ser localizado. O ponteiro serve para acessar os dados da partição contendo o nó, e por conseqüência todos os nós da partição.

2. Acessa todos os nós da partição e verifica qual possui o EUI_d a ser localizado.
3. Se encontrar o nó, copia a informação de posicionamento do nó.
4. Soma um erro aleatório na posição real do nó.
5. Retorna a localização do nó, já com um erro aleatório.

Essa função acima descrita simula o serviço de localização utilizado. A informação é obtida diretamente da estrutura de dados contendo as informações sobre o nó a ser localizado. A função poderia ser modificada para implementar algum outro método de localização.