Identificação de Região de Congestionamento através de Comunicação Inter-Veicular

Hélcio Mello[1], **Markus Endler**

[1]Departamento de Informática – Pontifícia Universidade Católica do Rio de Janeiro
Rua Marquês de São Vicente, 225, Gávea, Rio de Janeiro — RJ — Brasil

Abstract. *IVC (*Inter-Vehicle Communication*) is a promising technology for the next-generation of automotive vehicles. The envisaged benefits include safer vehicle control, multi-vehicle cruise control, better traffic planning, Internet connectivity, location-based services, etc.*
A typical application of IVC is traffic jam-avoidance. Recent publications already address this issue, but most of them focus on preventing the occurrence of traffic jams. Conversely, this article addresses metropolitan-area scenarios where drivers are already trapped in a jam and need assistance in how to escape them.
The proposed service employs a distributed algorithm to build an approximate congestion perimeter. This task is carried out by dissemination of positioning data of vehicles vertices of a polygon that is a close approximation of the congestion area. Given the polygon and its current location, a vehicle can then plan its escape-way route from the traffic jam.

Resumo. *IVC (*Inter-Vehicle Communication*) é uma tecnologia promissora para a próxima geração de veículos automotores. Os benefícios imaginados incluem mais segurança ao se guiar o veículo, controle de cruzeiro multi-veicular, melhor planejamento de tráfego, conectividade com a Internet, serviços baseados em localização, etc.*
Uma aplicação típica de IVC é evitar engarrafamentos. Publicações recentes já abordam esse assunto, mas a maioria delas se concentra em prevenir a formação de congestionamentos. Em contrapartida, este artigo discute cenários em áreas metropolitanas, onde motoristas já estão presos em um engarrafamento e precisam de ajuda para sair deles.
O serviço proposto emprega um algoritmo distribuído para construir um perímetro de congestionamento aproximado. Isso é conseguido pela disseminação de dados de posicionamento de veículos que são vértices de um polígono que aproxima a área afetada. Dado o polígono e sua localização atual, um veículo pode planejar sua rota de fuga do congestionamento.

1. Introdução

Comunicação Inter-Veicular (IVC — Inter-Vehicle Communication) promete tornar-se uma tecnologia indispensável nos futuros automóveis. Por isso, nos últimos anos, a maioria dos fabricantes tem investido em grandes esforços na pesquisa e desenvolvimento desses sistemas.

[1]Bolsista CNPq.

O principais objetivos são melhorar a segurança no trânsito, aumentar o grau de automação na condução dos veículos, fornecer informações para motoristas e passageiros, bem como melhorar o fluxo de veículos em estradas e em centros urbanos. A idéia central de IVC é possibilitar a troca contínua de dados entre veículos próximos, bem como de veículos com estações fixas posicionadas ao longo das estradas e ruas, a fim de permitir uma disseminação de dados entre os veículos que agilize uma reação coordenada a eventos.

Portanto, IVC constitui uma rede *ad hoc* móvel com as seguintes características centrais:

- Todos os nós são provedores, encaminhadores e consumidores de dados;
- Os dados difundidos são oriundos de vários sensores e câmeras em cada nó;
- A rede é aberta e de topologia altamente dinâmica, devido a velocidades relativas entre veículos de até 300 km/h. Conseqüentemente, há uma alteração constante na vizinhança dos nós, ou seja, o grupo de veículos com que se pode interagir em cada momento. Como resultado, a rede se fragmenta freqüentemente e sua redundância é comprometida;
- A transmissão sem fio está sujeita a interferências ainda pouco estudadas, principalmente devido às altas velocidades relativas e à obstrução momentânea por prédios, obstáculos naturais e outros veículos;
- Não existem limitações rígidas quanto à fonte de energia;
- Qualquer serviço disponibilizado através de IVC deve ser de caráter complementar e não-crítico, ou seja, a condução (e/ou o fluxo) dos veículos deve ser possível também sem este serviço.

[Blum et al. 2004] apresenta uma discussão interessante sobre alguns dos principais desafios para o desenvolvimento de serviços baseados em IVC. Algumas aplicações, especialmente as relativas à segurança, requerem latência mínima na propagação dos dados, de modo a possibilitar uma reação (coordenada ou não) o mais rápido possível. Essa preocupação motivou diversos trabalhos na área de QoS [Bonnet and Nikaein 2002, Chakrabarti and Mishra 2003].

Uma aplicação típica de IVC é a coordenação de veículos para resolver problemas de tráfego. As soluções encontradas na literatura discutem uma variedade de situações, desde a cooperação para se passar por um cruzamento até a ambiciosa meta de planejamento de tráfego global.

A maioria dos trabalhos sobre esse assunto se propõe a tentar *prevenir* a formação de congestionamentos, ou notificar os motoristas ainda não envolvidos para *evitarem* os já existentes. Porém, situações alheias à coordenação inter-veicular, como as provocadas por fenômenos naturais repentinos (deslizamentos de encostas, quedas de árvores, etc) ou falhas mecânicas nos veículos, podem provocar congestionamentos rapidamente.

Estes dificilmente podem ser previstos e, dependendo do planejamento e da infraestrutura de tráfego da cidade, podem assumir grandes proporções por períodos de tempo prolongados. Esse cenário constitui uma variante do problema de controle de tráfego aparentemente pouco investigada. Este trabalho propõe um serviço que utiliza IVC entre veículos *já envolvidos em um congestionamento* de modo a orientá-los sobre possíveis rotas de fuga. Estudos futuros investigam soluções mais elaboradas, como direção cooperativa e computação em grade.

O restante deste artigo está organizado da seguinte maneira: a Seção 2 discute alguns trabalhos sobre controle de tráfego, assim como publicações relacionadas. A análise de requisitos é feita na Seção 3, enquanto o serviço proposto é apresentado na Seção 4. A aplicação do serviço em um cenário de exemplo, seu desempenho esperado, assim como seus pontos fortes e fracos são analisados na Seção 5. Finalmente, a Seção 6 conclui este artigo, apontando possibilidades de trabalho futuro.

2. Trabalhos Relacionados

Um aspecto comum à maioria das propostas na área de IVC é a disseminação de informação. SODAD (*Segment-Oriented Data Abstraction and Dissemination*) [Wischhof et al. 2005] assume que cada veículo está equipado com um mapa digital, onde as estradas são divididas em segmentos de tamanho conhecido. As informações sobre os segmentos (como a densidade de tráfego) são modeladas como tuplas, e difundidas em *broadcasts* locais entre os veículos (distância de 1 *hop*). Cada instância SODAD, então, analisa a relevância da informação da tupla, baseada na distância ao ponto de onde ela veio e há quanto tempo ela foi produzida. Isso porque, de modo geral, o interesse em eventos ou facilidades próximas é maior do que naqueles mais distantes. Por exemplo, os preços praticados por um posto de gasolina próximo é mais relevante do que os de outro localizado a 100 km. Dependendo dessa relevância, a tupla é armazenada e retransmitida adiante ou descartada.

Uma outra arquitetura [Thomas et al. 2005] divide os veículos em aglomerados (*clusters*), dentro dos quais o roteamento de mensagens é proativo. A manutenção do estado da vizinhança (veículos alcançáveis sem roteamento) é feita por mensagens periódicas (*beacons*), cujos cabeçalhos contêm dados como posição, velocidade e direção atuais do remetente. A informação de tráfego é obtida por meio de *requisições de tarefas*, que especificam de quais veículos e sensores deseja-se obter informação. Por exemplo, pode-se tentar determinar se há gelo na pista em alguma região baseado em um termômetro externo e em informações do sistema de freios ABS de algum veículo na área em questão. O veículo que recebe uma requisição a responde com dados colhidos de seus próprios sensores, possivelmente combinados com leituras obtidas de seus vizinhos. Esses dados são enviados somente para os veículos cujas rotas possivelmente passem pela área analisada. Isso é determinado pela estrutura da estrada (obtida de um mapa digital) e informações sobre posição e velocidade do destinatário.

Técnicas de difusão similares foram empregadas para transmitir informações sobre tráfego pelos veículos, permitindo a realização de Sistemas de Transporte Inteligentes (ITS – *Intelligent Transportation System*) sobre IVC. As versões convencionais desses sistemas têm arquitetura centralizada, implicando retardos na transmissão dos dados sobre o tráfego à estação central [Wischhof et al. 2005].

[Balke et al. 2003] apresenta um sistema altamente personalizável, onde o motorista solicita uma rota até seu destino baseado em restrições sobre distância, condições climáticas e de tráfego, etc. A consulta (*query*) é feita pela Internet, via telefone celular, a uma base de dados central, atualizada assincronamente por estações de rádio locais.

Sob liderança do Instituto de Pesquisa Automobilística do Japão, o projeto *Probe Information System* [Maekawa 2004] transmite as leituras dos mais de 120 sensores de cada veículo para uma estação central. Com base em dados históricos, pode-se estimar

condições de tráfego em determinados pontos da cidade ou planejar a expansão das vias públicas.

As limitações da arquitetura de ITS centralizada motivou trabalhos independentes de infra-estrutura. [Goel et al. 2003] propõe a disseminação das informações de tráfego cooperativamente pelos veículos, eliminando a necessidade da infra-estrutura. Os dados são repassados para o sistema de navegação do veículo, que pode, então, ser consultado pelo motorista para descobrir a melhor rota ao destino desejado. A difusão é feita dividindo-se o espaço geográfico em células, de maneira semelhante à discutida no início desta seção.

Uma publicação mais recente [Wischhof et al. 2005] apresenta o SOTIS (*Self-Organizing Traffic Information System*), que também propõe otimização de rotas no trânsito independentemente de infra-estrutura. Como na maioria das outras propostas, assume-se a disponibilidade de um sistema de posicionamento (como o GPS) e de um mapa digital. A velocidade média de todos os veículos em cada célula é disseminada pela técnica SODAD, já mencionada nesta seção, para que cada veículo compute, individualmente, a melhor rota para seu destino. Resultados simulados mostram que apenas 1–3% dos veículos executando o SOTIS são o suficiente para se obterem bons resultados.

Em contraste ao paradigma da computação de rotas individualmente, a idéia de se usar uma grade (*grid*) *ad hoc* tem sido tema de publicações recentes. A computação em tempo real dos dados do tráfego possibilitariam direção cooperativa, re-roteamento de tráfego, etc. [Gaynor et al. 2004] e [Nekovee 2005] discutem algumas possibilidades nessa área. As principais dificuldades na realização desse sistema seriam armazenar o grande volume de dados resultantes e processá-los em tempo real.

Ao contrário das outras propostas discutidas nesta seção, que são de propósito geral, a disseminação de dados utilizada neste artigo é específica para o problema em que é empregada. Ela também se vale do *broadcast* local (i.e. confinado a uma distância de um *hop*), mas não depende de um mapa digital, nem divide a região em células. Em vez disso, cada nó alterna a transmissão de dados aos seus vizinhos e a combinação dos mesmos com os dados que recebe deles, de modo a construir o perímetro do congestionamento de forma distribuída.

Em contraste a isso, o serviço proposto neste artigo é baseado na computação individual dos dados do tráfego. Contudo, como mencionado na Seção 6.1, a evolução para uma solução baseada em grades e direção cooperativa já está sendo cogitada. Adicionalmente, trata-se de uma solução *complementar* às apresentadas nesta seção, concebida nem para *evitar* congestionamentos, nem para planejar rotas *à priori*, mas para tentar resolver o problema do congestionamento quando ele não puder ser evitado.

3. Requisitos do Serviço

Os diversos serviços disponibilizados em IVC possuem diferentes requisitos, como tempo de propagação das mensagens, vazão da rede e até mesmo a penetração da tecnologia de comunicação no mercado, visto que isso influencia na densidade de nós participantes da rede.

Serviços críticos para a segurança dos passageiros e/ou pedestres demandam garantias rígidas sobre recursos de processamento e transmissão de dados, como latência,

confiabilidade e largura de banda. Outras aplicações, apesar de prejudicadas por flutuações inesperadas das condições de execução, somente afetariam o conforto dos seus clientes, não lhes causando nenhum mal irremediável.

O serviço aqui proposto se enquadra nessa segunda categoria. Uma vez presos em um congestionamento, os veículos se movem em velocidades baixas, chegando muitas vezes a parar. Como resultado, ocorrem menos mudanças nas vizinhanças dos nós, e demora-se mais para que as posições dos veículos se modifiquem a ponto de justificarem uma nova difusão dessa informação, exigindo menos largura de banda da rede. Pelo mesmo motivo, a rapidez na propagação de dados também não é crítica. Em sendo pouco dinâmicas e possuindo muitos nós concentrados em áreas limitadas, as redes inter-veiculares formadas em um congestionamento tendem a ser conexas. A conectividade também está diretamente relacionada com o percentual de veículos equipados com a interface de comunicação sem fio.

Outro ponto importante é o papel da infra-estrutura de comunicação. Alguns serviços em IVC utilizam nós estacionários, freqüentemente dotados de mais recursos computacionais, para aumentar a conectividade da rede, seu poder de processamento ou armazenamento, e até mesmo oferecer conectividade com a Internet. O serviço descrito neste artigo não depende de qualquer infra-estrutura, de modo a poder ser utilizado em regiões onde ela não está disponível, como é o caso na maioria dos países.

O serviço proposto não exige nenhum *hardware* de rede ou pilha de protocolos *específicos*. Assume-se a existência de uma interface de comunicação sem fio e de um protocolo capazes de transmitir informações entre os nós. Também é necessário que eles disponham de um relógio (que não precisa estar sincronizado com o dos outros veículos) e conheçam sua posição atual, apesar de um erro de vários metros ser tolerável. As tecnologias de posicionamento existentes, como o GPS, já satisfazem esses requisitos.

4. Abordagem Proposta

No escopo deste serviço, podem ser identificados quatro tipos básicos de veículos, de acordo com suas características técnicas e comportamento:

- Veículos Convencionais — Como a tecnologia de comunicação inter-veicular é recente, sua penetração no mercado ainda é tímida. Veículos não equipados com uma interface de comunicação sem fio são incapazes de transmitir suas posições ou rotear pacotes pela rede. Ainda assim, por ocuparem espaço no trânsito, precisam ser levados em consideração na computação das rotas dos outros veículos.
- Veículos Não-participantes — Não seria uma suposição realista assumir que *todos* os nós dotados de IVC executem o serviço discutido neste artigo. Assim, alguns nós não poderão contribuir com dados sobre tráfego em suas imediações, mas terão papel importante no roteamento de pacotes para os veículos participantes.
- Veículos Semi-Bizantinos — Infelizmente, nem todos os motoristas respeitam semáforos ou sinalização. Analogamente, não é razoável esperar que *todos* os usuários deste serviço sigam à risca as rotas sugeridas, pois alguns podem acreditar, em função de suas experiências pessoais no trânsito ou de outros motivos, que outras rotas de suas preferências os levarão mais rapidamente aos seus destinos. Neste artigo tais motoristas são chamados *semi-bizantinos*. Mesmo não seguindo as rotas sugeridas, eles contribuem para o serviço com a coleta das posições dos

seus vizinhos e com o cálculo do *polígono de congestionamento* local, que se assumee estar *correto*. Daí a denominação *semi*-bizantinos.

- Veículos Participantes — Esta categoria compreende aqueles equipados para IVC, executando o serviço discutido aqui e cujos motoristas seguem as rotas sugeridas. Todos os veículos participantes produzem, roteiam e consomem informações sobre tráfego.

A computação das rotas precisa levar em consideração certos problemas relacionados ao comportamento dos motoristas. No pior caso, podem-se considerar aqueles que, por meio de um aplicativo que imite o comportamento do serviço, injetem pacotes contendo informações incorretas sobre o posicionamento dos veículos. Tais motoristas poderiam ser classificados como *totalmente bizantinos*, mas fogem ao escopo deste trabalho, sendo objeto de investigação futura.

Para que o serviço possa sugerir rotas, a posição dos *veículos participantes* precisa ser disseminada pela rede, de modo que o perímetro do congestionamento possa ser determinado. Esta seção descreve em detalhes como isso é feito e quais os conceitos utilizados.

4.1. Polígono de Congestionamento

O estado do congestionamento, visto por um veículo V, é representado por uma estrutura de dados denominada *polígono de congestionamento de V*. Ela consiste em uma lista de coordenadas (vértices) com as seguintes características básicas:

1. As coordenadas de V *sempre* estão localizadas no interior (ou na fronteira) do polígono;
2. Seus vértices são coordenadas de *veículos participantes* ou *semi-bizantinos*;
3. O polígono é convexo.

Dessas características decorrem algumas propriedades interessantes. De acordo com a característica 2, se os vértices do polígono estão envolvidos no congestionamento (por serem *veículos participantes*), então, em circunstâncias normais, os veículos contidos no polígono também estão.

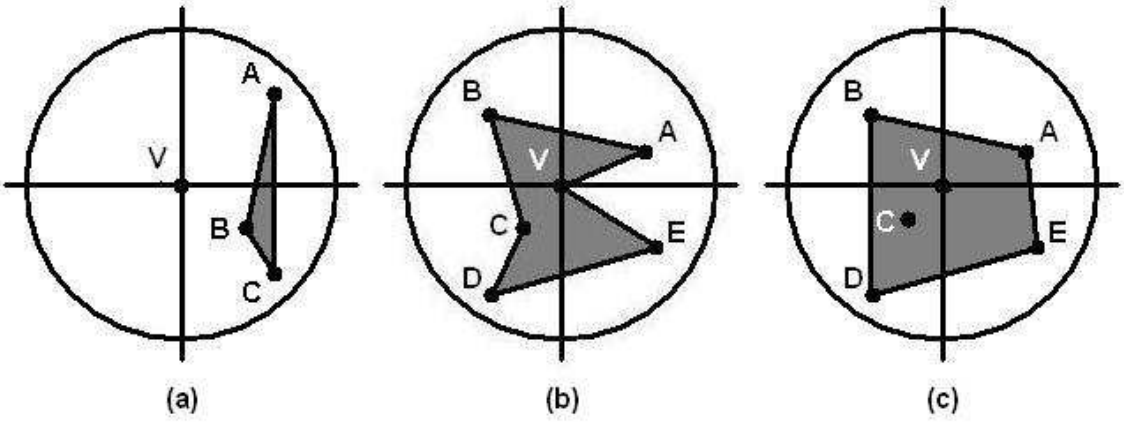

Figura 1. Candidatos a polígonos de congestionamento do veículo V.

A Figura 1 mostra três candidatos a *polígonos de congestionamento* de um veículo V. A circunferência centrada nesse nó corresponde ao alcance do seu sinal de rádio, e a

região cinza delimita o polígono de congestionamento em cada caso. *Todos* os *veículos participantes* são elegíveis a vértices do polígono, conforme a característica 2, e são representados por pontos assinalados com letras. Os detalhes relativos ao mapa e demais veículos foram omitidos por motivos de clareza.

O triângulo da Figura 1(a) não é um polígono de congestionamento, pois viola a característica 1. Nesse exemplo, as áreas contidas nos triângulos ABV e BCV, apesar de provavelmente pertencerem ao congestionamento, ficariam fora do polígono se a característica 1 fosse ignorada.

A Figura 1(b) satisfaz às características 1 e 2, mas continua não sendo um polígono de congestionamento, pois desrespeita a característica 3. O candidato da Figura 1(c), por sua vez, é um *polígono de congestionamento*, pois é formado por *veículos participantes*, contém o ponto V e é convexo. Isso é importante porque, na Figura 1(b) os triângulos BCD e AEV não estão contidos no polígono, mas como existem veículos *não-participantes* e *convencionais* espalhados pelo congestionamento, não seria realista assumir que *todos* eles estariam dentro do polígono. Além disso, a Figura 1(b) possui mais vértices do que sua versão convexa (Figura 1(c)). Por esses motivos, os *polígonos de congestionamento* convexos exigem menos memória para serem armazenados, menos tempo para serem processados e menos largura de banda para serem disseminados, tornando-os menos suscetíveis à perda de pacotes.

4.2. Disseminação e Agregação de Dados

Conforme explicado na Seção 4.1, se um veículo está contido em um *polígono de congestionamento* então assume-se que ele esteja preso no tráfego. Essa propriedade, estendida para o nível global, permite aproximar o perímetro da região congestionada pela união de todos os *polígonos de congestionamento*.

A disseminação do estado do congestionamento é feita por *broadcast* de mensagens *POLYGON* pelos *veículos participantes* ou *semi-bizantinos* na vizinhança local, ou seja, a uma distância de um *hop*. Essa mensagem contém uma lista de endereços e coordenadas dos veículos que são vértices do *polígono de congestionamento*.

Ao receber mensagens *POLYGON*, cada nó combina os vértices recebidos com os seus próprios, produzindo um novo polígono convexo que contenha todos os originais. A fusão desses polígonos, em sucessivas etapas de trocas de mensagens, permite estimar a área afetada pelo congestionamento. Um bom algoritmo para isso é o de Graham [Graham 1972, Gries and Stojmenović 1987], cuja complexidade é $\Theta(n \log n)$. Dado um conjunto de pontos, o algoritmo encontra o menor polígono *convexo* que contenha a *todos* eles, tornando-o conseqüentemente um *polígono de congestionamento*.

4.3. Detalhes do Protocolo

O protocolo se inicia quando os veículos detectam que estão em um congestionamento. Isso pode ser feito por um comando do motorista pela interface com o usuário, ou detectado automaticamente por regras baseadas em dados lidos pelos sensores do veículo. Por exemplo, se um carro está freqüentemente parado ou se movendo lentamente com outro veículo à sua frente há algum tempo poder-se-ia concluir que ele está preso no tráfego. Essa regra, porém, pode precisar de ajustes: o motorista pode ter parado seu carro simplesmente porque o semáforo fechou, ou porque decidiu estacionar para fazer compras.

Os detalhes sobre o mecanismo de detecção de congestionamentos não são abordados neste trabalho.

Uma vez iniciado, o protocolo executa a função *onStart ()*, mostrada na Listagem 2. A linha 1 inicializa as variáveis de estado (Tabela 1), conforme a Listagem 1. A seguir, o laço principal se encarrega da construção e manutenção do *polígono de congestinamento*, o que pode ser dividido em três tarefas menores: atualização da posição do veículo (linha 3), eliminação de vértices que deixaram a área (linha 4) e disseminação do *polígono de congestinamento* (linhas 6–9).

isCongested	Variável *booleana* que assume o valor *true* se, e somente se, o nó detectou um congestionamento.
currentLocation	Armazena a posição atual do veículo.
lastPolygonUpdLoc	A posição em que o nó estava quando recalculou seu *polígono de congestionamento* pela última vez.
lastPolygonRecvTimes	Uma tabela que armazena o instante de tempo em que a última mensagem *POLYGON* foi recebida de cada vértice do *polígono de congestionamento* que se encontre na vizinhança.
nextPolygonSendTime	o instante de tempo em que a próxima mensagem *POLYGON* deve ser transmitida.
polygon	conjunto de pares *(endereço, localização)* dos vértices do *polígono de congestionamento*.

Tabela 1. Variáveis de estado de cada nó.

Listagem 1 *init ()*: Inicializa as variáveis de estado.

```
1: isCongested ← true;
2: currentLocation ← getLocation ();
3: lastPolygonUpdLoc ← undefined;
4: lastPolygonRecvTimes ← {};
5: nextPolygonSendTime ← getTime ();
6: polygon ← {(getAddress (), currentLocation)};
```

Listagem 2 *onStart ()*: Disparado quando o congestionamento é detectado.

```
1: init ();
2: while isCongested do
3:     updateLocation ();
4:     refreshVertices ();
5:
6:     if getTime () ≥ nextPolygonSendTime then
7:         recalcPolygon ();
8:         broadcast (polygon);
9:     end if
10: end while
```

A atualização da posição do veículo é feita pela função *updateLocation ()*, presente na Listagem 3. O objetivo é detectar possíveis alterações no polígono decorrentes

do movimento do próprio motorista. As constantes e primitivas utilizadas se encontram, respectivamente, nas Tabelas 2 e 3. Os seguintes casos foram previstos, e são ilustrados na Figura 2:

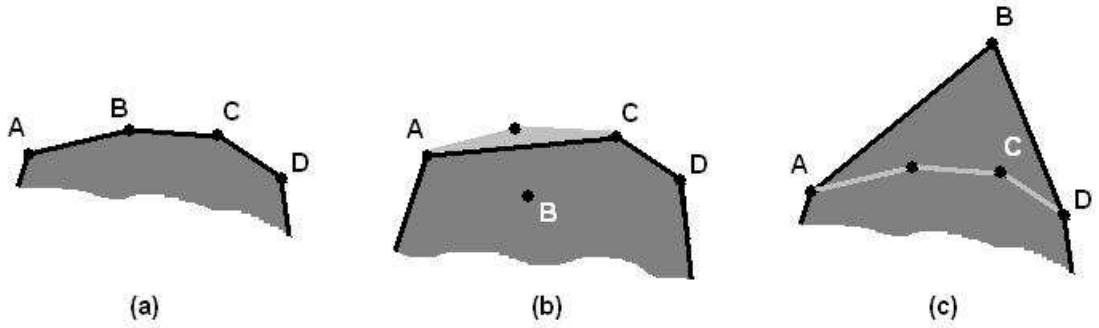

Figura 2. Alterações no polígono em função do movimento de um nó.

1. Veículo permanece dentro do polígono (linhas 3–7). Se ele era um vértice (linha 3) e se deslocou mais do que um certo limiar (linha 4), o polígono é recalculado (linha 5). A comparação das Figuras 2(a) e 2(b) ilustra esse caso com a movimentação do nó B para dentro do polígono, eliminando a região cinza-clara. Se o veículo não for vértice do polígono recém-calculado (como ocorre na Figura 2(b)), ele será eliminado com a passagem do tempo (discutido mais adiante).
2. Veículo sai do polígono (linhas 8–14), o que modifica pelo menos um dos seus vértices. Por isso, se o motorista ainda não tiver saído do congestionamento (linha 8), e tiver se deslocado de uma distância de pelo menos ΔL (linha 10) agenda-se a próxima mensagem *POLYGON* (linhas 11–13). A diferença entre as Figuras 2(a) e 2(c) mostra como o nó B, ao se mover, altera o perímetro do polígono, do qual agora o nó C não faz mais parte. Assim como no caso anterior, ele será removido do polígono automaticamente.

Δt_{POL}	Intervalo de tempo mínimo entre duas mensagens *POLYGON* consecutivas.
Δt_{ELIM}	Tempo mínimo que um veículo precisa esperar, sem ter recebido mensagens de outro nó, antes de concluir que esse nó tenha falhado ou deixado o congestionamento.
ΔL	Deslocamento mínimo que um vértice precisa sofrer para provocar a difusão de mensagens de atualização do polígono de congestionamento.

Tabela 2. Constantes utilizadas pelo protocolo.

A próxima tarefa do laço principal é detectar vértices que são suspeitos de terem falhado ou deixado em definitivo a área congestionada. Isso é feito pela função *refreshVertices ()*, que considera um vértice suspeito caso ele passe muito tempo (Δt_{ELIM}) sem enviar mensagens (Listagem 4). A variável *lastPolygonRecvTimes* (Tabela 1) é utilizada para esse propósito. Todos os nós considerados suspeitos (linha 3) são inseridos em uma lista (linha 5) para serem retirados do polígono (linha 9).

A eliminação de vértices também é causada por mensagens *REMOVE*, tratadas na Listagem 5. Essencialmente, se algum vértice pertence ao polígono (linha 3) ele é

broadcast (msg)	Envia a mensagem *msg* a todos os nós dentro do alcance direto de transmissão, ou seja, sem roteamento;
getSender (msg)	Retorna o endereço do remetente da mensagem *msg*;
getAddress ()	Retorna o endereço do nó;
getLocation ()	Informa a localização atual do veículo, um par ordenado $L(x, y)$;
getTime ()	Obtém a hora atual, com resolução em segundos.

Tabela 3. Primitivas de sistema requeridas pelos veículos.

Listagem 3 *updateLocation ()*: Verifica se o movimento do veículo altera o *polígono de congestionamento*.

```
 1: currentLocation ← getLocation ();
 2: if isInside (polygon, currentLocation) then
 3:    if (getAddress (), lastPolygonUpdLocation) ∈ polygon then
 4:       if distance (currentLocation, lastPolygonUpdLocation) > ΔL then
 5:          recalcPolygon ();
 6:       end if
 7:    end if
 8: else if escapedCongestion () then
 9:    isCongested () ← false;
10: else if distance (currentLocation, lastPolygonUpdLocation) > ΔL then
11:    if nextPolygonSendTime = ∞ then
12:       nextPolygonSendTime ← getTime () + Δt_POL;
13:    end if
14: end if
```

removido (linha 4) e incluído em uma lista (linha 5). Caso haja a remoção de pelo menos um vértice (linha 9), a lista dos que foram efetivamente excluídos é propagada adiante (linha 10) e o polígono é atualizado (linha 11). Com efeito, a mensagem *REMOVE* é disseminada pela rede sem que um mesmo veículo a transmita mais de uma vez, pois ela só é passada adiante caso tenha havido a remoção de *pelo menos um* vértice.

A inclusão ou atualização de uma entrada na tabela *lastPolygonRecvTimes* se dá quando uma mensagem *POLYGON* é recebida. Como mostra a Listagem 6, o polígono é recalculado (linhas 2–3) e, caso o remetente seja um dos vértices do novo polígono, o instante do recebimento é registrado na tabela (linhas 5–8).

A última tarefa da função *onStart ()* é transmitir o *polígono de congestinamento* toda vez que o instante de tempo *nextPolygonSendTime* for alcançado. A primeira transmissão é garantida pela inicialização dessa variável na Listagem 1. As subseqüentes são feitas toda vez que o polígono muda de forma significativa (constante ΔL, definida na Tabela 2), ou quando um veículo é vértice do *polígono de congestionamento* recalculado. Veículos nessa condição precisam enviar mensagens *POLYGON* regularmente, pois são eles quem delimitam o perímetro estimado da área congestionada, que é a informação de que os demais usuários precisam para planejar suas rotas de saída. Esse procedimento é necessário mesmo se o nó não tiver mudado de posição significativamente, sob pena de ser suspeito de ter falhado ou deixado a área congestionada (Listagem 4).

Listagem 4 *refreshVertices ()*: Remove vértices que não enviam *POLYGON* há muito tempo.

```
1: removeList ← {};
2: for all (V, t) ∈ lastPolygonRecvTimes do
3:    if getTime () > t + Δt_ELIM then
4:       lastPolygonRecvTimes ← lastPolygonRecvTimes − {(V, t)};
5:       removeList ← removeList ∪ {V};
6:    end if
7: end for
8:
9: onRemove (removeList);
```

Listagem 5 *onRemove (removeList)*: Chamada no recebimento de mensagens *REMOVE*, ou manualmente pela função *refreshVertices ()*.

```
1: removedVertices ← {}
2: for all V ∈ removeList do
3:    if (V, ∗) ∈ polygon then
4:       polygon ← polygon − {(V, ∗)};
5:       removedVertices ← removedVertices ∪ {V};
6:    end if
7: end for
8:
9: if removedVertices ≠ {} then
10:   broadcast (REMOVE (removedVertices));
11:   recalcPolygon ();
12: end if
```

A função *recalcPolygon ()*, descrita na Listagem 7, é responsável por recalcular o *polígono de congestionamento*. Isso é necessário quando um nó recebe polígonos dos outros veículos (Listagem 6), ou ainda quando um motorista cruza uma aresta do polígono, expandindo-o (Listagem 3). As linhas 2–4 atualizam a posição do próprio veículo no conjunto de pontos e aplica o algoritmo de Graham (omitido por motivos de espaço) para obter o novo *polígono de congestionamento*. Finalmente, as linhas 6–10 agendam a próxima transmissão caso o veículo seja vértice do novo polígono.

O ciclo de troca de mensagens termina quando o motorista finalmente se livra do congestionamento. Essa detecção é feita pela função *escapedCongestion ()*, chamada na linha 8 da Listagem 3. Mais uma vez, os detalhes sobre como isso é feito estão além do escopo deste trabalho, mas analogamente ao discutido sobre detecção de congestionamentos no início desta seção, o próprio motorista pode desativar o serviço manualmente, pela interface com o usuário. Alternativamente, regras que analisam se o veículo mantém uma certa velocidade (por exemplo, 40 km/h) durante algum tempo (tipicamente de um a dois minutos) poderiam ser usadas para inferir que ele provavelmente não se encontra mais preso no tráfego.

Nesse meio tempo, o motorista deve ser orientado com base em sua posição e no *polígono de congestionamento*. Em sua versão atual, o serviço calcula qual a aresta

Listagem 6 *onPolygon (newPolygon)*: Chamada no recebimento da mensagem *POLYGON*.

```
1: sender ← getSender (newPolygon);
2: polygon ← polygon ∪ newPolygon;
3: recalcPolygon ();
4:
5: if (sender, *) ∈ polygon; then
6:     lastPolygonRecvTimes ← lastPolygonRecvTimes − (sender, *);
7:     lastPolygonRecvTimes ← lastPolygonRecvTimes ∪ (sender, getTime ());
8: end if
```

Listagem 7 *recalcPolygon ()*: Recalcula o *polígono de congestionamento*.

```
1: lastPolygonUpdLoc ← currentLocation;
2: polygon ← polygon − {(getAddress (), *)};
3: polygon ← polygon ∪ {(getAddress (), currentLocation)};
4: grahamScan (polygon);
5:
6: if {(getAddress (), currentLocation)} ∈ polygon then
7:     if nextPolygonSendTime = ∞ then
8:         nextPolygonSendTime ← getTime () + Δt_POL;
9:     end if
10: end if
```

do polígono mais próxima da posição atual do veículo e orienta o motorista a seguir nessa direção. A interação com o usuário é feita de acordo com os recursos disponíveis no veículo. Em um extremo, este serviço pode ser integrado com o próprio sistema de navegação automotivo, passando-se para ele as coordenadas do ponto de saída do congestionamento e deixando-o calcular a melhor rota baseado em informações de um mapa digital. No outro extremo, em não existindo a bordo tal sistema ou mapas digitais, o serviço indicaria apenas um curso e a distância ao destino, como por exemplo "30 graus à Leste, 750 m", atualizados periodicamente.

5. Exemplo

A Figura 3 ilustra um cenário simplificado de congestionamento, onde cada ponto assinalado por uma letra representa um *veículo participante*. Os demais veículos, assim como detalhes sobre ruas e avenidas, foram omitidos por motivos de clareza. Cada nó está ao alcance de transmissão de outro se, e somente se, os respectivos pontos estiverem ligados por uma aresta. Assume-se que o enlace de comunicação seja bidirecional.

Num instante inicial, mostrado Figura 3(a), os veículos detectam o congestionamento, e o polígono de cada um deles possui um único vértice[1]: o próprio veículo. Não é necessário que os nós iniciem o protocolo simultaneamente, nem que estejam sincronizados, para que o serviço funcione corretamente. Essas suposições estão sendo feitas para manter o exemplo simples.

[1]Os autores reconhecem que a Matemática não define polígonos com menos de três vértices. Porém, por praticidade, este artigo considera que qualquer conjunto não-vazio de vértices defina um polígono.

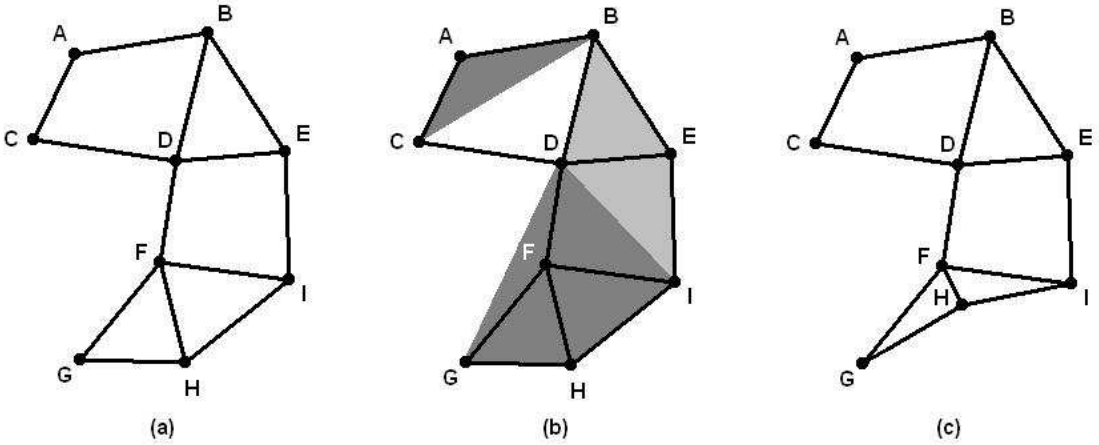

Figura 3. Formação e manutenção do *polígono de congestionamento*.

Nó	1^a. fase	2^a. fase	3^a. fase
A	A	ABC	ABEDC
B	B	ABED	ABEIFC
C	C	ACD	ABEFC
D	D	BEFC	ABEIHGC
E	E	BEID	ABEIHC
F	F	DIHG	BEIHGC
G	G	GFH	DIHG
H	H	GFIH	DEIHG
I	I	EIHF	BEIHG

Tabela 4. Evolução dos polígonos de congestionamento durante o protocolo.

Após a inicialização das variáveis de estado (Tabela 1), cada nó transmite sua primeira mensagem *POLYGON*, contendo apenas seu próprio endereço. Cada veículo atualizará seu polígono com as mensagens que receber dos seus vizinhos. Supondo que nenhuma delas se perca, os polígonos de cada nó serão os mostrados na coluna "2^a. fase" da Tabela 4, e serão transmitidos na mensagem *POLYGON* seguinte. A Figura 3(b) destaca os polígonos calculados pelos nós A, F e E nesse instante, estando este último em tonalidade de cor diferente para efeito de contraste. Após o recebimento da terceira mensagem *POLYGON* todos os nós terão o mesmo polígono, $ABEIHGC$.

A Figura 3(c) ilustra a atualização do *polígono de congestionamento* quando um nó deixa de ser um de seus vértices. Uma vez detectada essa condição, o veículo pára de enviar mensagens *POLYGON* periodicamente, o que causará sua eliminação da lista de vértices (mensagem *REMOVE*). O roteiro passo-a-passo será omitido por motivos de espaço.

5.1. Discussão

Ao contrário dos nós que se movimentam *no interior* do polígono, os dos vértices alteram o perímetro da área congestionada toda vez que se movem. Como explicado na Seção 4.3, esses veículos precisam enviar mensagens *POLYGON* periodicamente, de modo a manter

o polígono atualizado e impedir que os outros nós suspeitem que eles tenham falhado ou deixado a área congestionada.

Esse gasto extra de largura de banda ocorre somente nas bordas do polígono, e pode ser minimizado pela simplificação das mensagens *POLYGON* que os vértices enviam. Esses veículos passariam a transmitir apenas a posição dos vértices mais próximos de si mesmos, em vez de enviar o polígono inteiro. Isso reduziria ainda mais o tamanho da mensagem *POLYGON* e a largura de banda consumida nas bordas do congestionamento, diminuindo a probabilidade de que o pacote se perca e de que seu remetente seja suspeito de ter falhado. Essa redução não traria prejuízo ao protocolo, pois os veículos no interior do polígono ainda estariam habilitados a combinar a vizinhança dos vértices para construí-lo.

Um argumento adicional a favor da atualização periódica da posição dos vértices se deve à importância que ela tem para os veículos próximos às arestas do polígono. Quando o fluxo de veículos na proximidade de um vértice começa a se normalizar, o polígono tende a se retrair nessa região, tornando-a mais atraente para os outros motoristas. Em contrapartida, um aumento na concentração de tráfego nos arredores do polígono causaria sua expansão, tornando as rotas que passem pela região menos eficazes para a fuga do congestionamento. Em ambos os casos, o impacto dessas alterações diminui com o aumento da distância do nó às arestas do polígono. A título de exemplo, se um veículo se encontra a 500m da aresta mais próxima, uma diminuição de 50m na extensão da rota é consideravelmente menos significativa do que é para aqueles que já se encontram a 100m do mesmo local.

Outro ponto relativo à largura de banda é a disseminação de mensagens *REMOVE*. Em sua versão atual, toda vez que um vértice se livra do congestionamento o serviço divulga a alteração do polígono com essa mensagem. Isso não pode ser feito com a mensagem *POLYGON*, pois esta tem caráter aditivo, ou seja, o conjunto de vértices transmitido é unido ao polígono mantido em cada nó. Uma possível solução para esse problema seria o envio de mensagens *REMOVE* com um espaçamento mínimo entre elas, assim como é feito com mensagens *POLYGON* (constante Δt_{POL}, definida na Tabela 2). Isso retardaria a propagação dessas mensagens aos veículos mais distantes do local onde a remoção do vértice ocorreu, mas, conforme já discutido nesta seção, a relevância desse tipo de informação é menor para os veículos mais distantes.

A função *refreshVertices ()*, mostrada na Listagem 4, tenta tornar o uso de uma mensagem *LEAVE* desnecessária. Em vez de um nó sinalizar, explicitamente, que está deixando a área, ele é removido pela simples passagem de algum tempo (Δt_{ELIM}, encontrada na Tabela 2) sem ter transmitido seu polígono. Essa estratégia visa resolver dois problemas. O primeiro é um veículo ter detectado que saiu do congestionamento e já não se encontrar mais ao alcance de transmissão dos outros nós, o que os impediria de enviar uma mensagem *LEAVE*. O segundo problema é que tal mensagem poderia se perder, perpetuando a existência de um veículo no polígono. Entretanto, se um nó e *todos* os seus vizinhos imediatos (i.e. alcançáveis sem roteamento) deixam a área congestionada dentro do intervalo de tempo Δt_{ELIM}, o esquema de remoção de vértices falhará.

No entanto, o *polígono de congestionamento* se adaptará a essa falha automaticamente, desde que algum veículo se mova de modo que o perímetro do polígono contenha

a região da falha. Apesar disso, pode ser mais conveniente recalculá-lo desde o início, por meio de uma mensagem *RESET*, quando se passarem grandes intervalos de tempo (por exemplo, uma hora). Essa verificação é objeto de investigação futura, na qual se pretende simular o modelo e estudar analiticamente seu comportamento à perda de mensagens.

6. Conclusão

Este artigo discutiu como a comunicação interveicular (IVC) pode ser usada para auxiliar motoristas a saírem de congestionamentos, dado que eles não puderam ser evitados de antemão. Por se tratar de uma rede *ad hoc* bem particular, os requisitos do serviço são diferentes dos encontrados em outras publicações: há menos dinamismo, maior concentração de veículos, maior facilidade para difusão multi-hop na rede, etc.

A idéia central do serviço é construir o *polígono de congestionamento* e mantê-lo atualizado. Esses são os objetivos do protocolo apresentado, que permite ao motorista planejar sua rota de saída da região afetada de acordo com os recursos à sua disposição, como mapa digital e sistema de navegação.

A proposta discutida neste trabalho aborda o problema do congestionamento sob um novo ângulo, aparentemente pouco explorado na literatura. O estudo de melhorias na solução proposta, assim como o desenvolvimento de novas abordagens para o problema do tráfego, torna os resultados dessa linha de pesquisa promissores.

6.1. Trabalhos Futuros

Devido a limitações de tempo e espaço, vários aspectos do problema do congestionamento não puderam ser discutidos, sendo deixados para trabalhos futuros. Por exemplo, a modelagem do estado do congestionamento, por polígonos convexos, simplifica a difusão dessa informação pela rede. Por outro lado, isso impõe o pressuposto de que o congestionamento seja uniforme em sua extensão e convexo, o que nem sempre é verdadeiro. Além disso, áreas vazias (como lagoas, parques, etc.) poderiam ser erroneamente indicadas como possíveis rotas de saída.

Esses problemas poderiam ser amenizados com o uso de um mapa digital e de polígonos côncavos que maximizassem o custo/benefício do *overhead* das mensagens *POLYGON* contra uma melhor aproximação do perímetro da área congestionada. A heterogeneidade do congestionamento, como áreas em que o trânsito flui mais rapidamente, pode ser melhor representada por polígonos divididos em regiões, onde as propriedades do tráfego seriam consideradas constantes. Naturalmente, o nível de detalhe adequado para cada polígono é uma solução de compromisso entre *overhead* e exatidão nos dados do tráfego.

Os próximos passos seriam, no entanto, simular o protocolo proposto e analisar sua eficiência. Em um horizonte mais distante, a meta é estudar como evoluir a solução individualista proposta neste artigo para uma versão *cooperativa*. O uso grades (*grids*) permitiria a paralelização da computação das rotas, em um esforço de organizar o tráfego na direção do bem-comum.

7. Agradecimento

Trabalho parcialmente financiado pelos projetos CNPq 55.2068/02-2 (ESSMA) e 479824/04-5 (Ed. Universal).

Referências

Balke, W.-T., Kiessling, W., and Unbehend, C. (2003). A situation-aware mobile traffic information system. In *HICSS '03: Proceedings of the 36th Annual Hawaii International Conference on System Sciences (HICSS'03) - Track 9*, page 292.2, Washington, DC, USA. IEEE Computer Society.

Blum, J., Eskandarian, A., and Hoffman, L. (2004). Challenges of intervehicle ad hoc networks. *IEEE Transactions on Intelligent Transportation Systems*, (4).

Bonnet, C. and Nikaein, N. (2002). A glance at quality of service models for mobile ad hoc networks. In *16eme Congrès DNAC (De Nouvelles Architectures pour les Communications), December 2-4, 2002, Paris, France*.

Chakrabarti, S. and Mishra, A. (2003). *Quality of service in mobile ad hoc networks*. CRC Press, Inc., Boca Raton, FL, USA.

Gaynor, M., Moulton, S. L., Welsh, M., LaCombe, E., Rowan, A., and Wynne, J. (2004). Integrating wireless sensor networks with the grid. *IEEE Internet Computing*, 8(4):32–39.

Goel, S., Imielinski, T., Ozbay, K., and Nath, B. (2003). Poster abstract: sensors on wheels – towards a zero-infrastructure solution for intelligent transportation systems. In *SenSys '03: Proceedings of the 1st international conference on Embedded networked sensor systems*, pages 338–339, New York, NY, USA. ACM Press.

Graham, R. L. (1972). An efficient algorithm for determining the convex hull of a finite planar set. *Inform. Process. Lett.*, 1:132–133.

Gries, D. and Stojmenović, I. (1987). A note on Graham's convex hull algorithm. *Inform. Process. Lett.*, 25:323–327.

Maekawa, M. (2004). ITS (Intelligent Transportation System) solutions. *NEC Journal of Advanced Technology*, (3).

Nekovee, M. (2005). Sensor networks on the road: the promises and challenges of vehicular ad hoc networks and grids. In *Workshop on Ubiquitous Computing and e-Research*.

Thomas, M., Peytchev, E., and Al-Dabass, D. (2005). Auto-sensing and distribution of traffic information in vehicular ad hoc networks. *International Journal of Simulation*, (4).

Wischhof, L., Ebner, A., and Rohling, H. (2005). Information dissemination in self-organizing intervehicle networks. *IEEE Transactions on Intelligent Transportation Systems*, (1).