Relatório Final

MAC 5701 – Tópicos em Ciência da Computação

Aluno: Eugênio Akihiro Nassu

Orientador: Prof. Dr. Marcelo Finger

Área de Interesse: Bancos de Dados e Mobilidade

1 Introdução

Apresentamos neste documento os principais resultados e avanços obtidos durante este semestre. Com o início da fase de implementação de nossa tese, novos problemas emergiram, o que será relatado em seguida. Na seção 2 deste trabalho descrevemos a área de estudo. A seguir descreveremos os principais problemas relacionados que encontramos na literatura. A seguir descreveremos o problema que será tratado em nossa tese. Por fim, tiraremos algumas conclusões visando o cumprimento dos objetivos desta disciplina.

Gostaríamos de destacar o fato de termos submetido nosso trabalho para duas conferências nacionais: o SBBD 2001 (Simpósio Brasileiro de Bancos de Dados) e WSCF 2001 (Workshop de Comunicação sem Fio e Computação Móvel).

2 A área de estudo

Nos últimos anos, houve um desenvolvimento significativo no ramo das telecomunicações, com a popularização de equipamentos capazes de receber informações através de ligações sem fio, como satélite, rádio ou telefonia celular. Esses equipamentos, aliados a nova geração de computadores, como os "palmtops", PDA[1] e "notebooks", além dos telefones celulares, criaram possibilidades novas, como o uso de computadores portáteis e poderosos em carros, aviões, helicópteros, em outros meios de transporte, e mesmo por pedestres. Um exemplo de aplicação seria o de controle e fornecimento de informações sobre trânsito em uma grande cidade. Um sistema como esse poderia ser extremamente útil para que um usuário conseguisse um caminho mais rápido para ir ao seu destino, usando informação que poderia ser atualizada em tempo real, por monitores e observadores colocados em pontos estratégicos. Outro uso desse sistema poderia ser o de fornecer informações turísticas, como hotéis próximos, bares, restaurantes ou serviços, como postos de gasolina, oficinas ou borracheiros.

Todos esses dados sobre atrações, serviços e trânsito podem ser armazenados em bancos de dados, acessíveis ao usuário através de uma conexão sem fio. O aplicativo usado pelo computador portátil do usuário pode efetuar consultas ou atualizações nesse banco de dados. Eventualmente, essas consultas dependerão do local onde o usuário se encontra. Por exemplo, o usuário poderia perguntar ao sistema qual o melhor caminho a seguir do ponto onde ele está para a sua casa. Ele poderia também precisar de um posto de gasolina, e perguntar qual o posto mais próximo. Outra pergunta que ele poderia fazer é: "quais os restaurantes pelos quais eu vou passar nos próximos 10 minutos?". Em todos esses casos, a posição do usuário na hora da solicitação é importante e pode ser diferente no momento em que o usuário recebe a resposta devido ao tempo de processamento da consulta, demoras na transmissão de dados, possíveis desconexões ou quedas de sinal, entre outros fatores.

Em todas essas consultas, um valor para a posição do usuário deve ser determinado ou aproximado de alguma forma. Como esse valor não é constante, usaremos o termo variável aqui. Nosso objetivo é estimar o valor da variável aqui de maneira precisa. O método de estimar essa variável depende de cada situação, como veremos adiante. Observamos, entretanto, que a variável aqui possui vários significados, ou semânticas.

[1] Evolução das agendas eletrônicas, que caminham para se unir aos palmtops e telefone celular

Estudaremos, portanto, as semânticas da variável e formas de implementar um sistema que calcule essas semânticas corretamente.

3 Modelo de referência

Neste trabalho, assumiremos que os usuários móveis possuem um computador portátil, do tipo palmtop ou notebook. Os computadores móveis se conectam a uma rede de alta velocidade, através de conexões sem fio, que podem ser via satélite, por telefonia celular, entre outras. Essa conexão sem fio se faz a partir de um computador, denominado *estação base*. A estação base está ligada a uma antena, satélite ou outro receptor/transmissor capaz de se comunicar com os computadores móveis. A região de alcance de um receptor é sempre limitada, e é chamada *célula*. As células são as regiões representadas por nuvens na Figura 1. Durante o movimento do computador móvel, pode haver diversas mudanças de célula. As *trocas de célula* são conhecidas por "handoff" na literatura. Na Figura 1, alguns dos computadores estão trocando de célula. Há momentos em que os computadores móveis estão fora de alcance de qualquer célula, estado em que podem permanecer por tempo indeterminado. As conexões sem fio possuem uma velocidade muito menor que a rede sem fio, acarretando maior demoa na transmissão e recepção de dados. Os computadores móveis, por sua vez, podem ser desconectados da rede por estarem fora de uma célula, ou por serem desligados, seja voluntariamente, seja involuntariamente, por exemplo, quando a bateria ou pilha do computador estiver sem carga. O computador pode ficar em um modo de espera, denominado *doze mode*, em que este apenas recebe sinais, esperando uma mensagem específica. Esse modo de funcionamento é semelhante ao usado no modo de espera dos telefones celulares.

Alguns aspectos da computação móvel precisam ser destacados. Um deles diz respeito à velocidade de transmissão de dados através de meios sem fio, que é menor que os meios tradicionais. Há uma previsão de aumento considerável na velocidade de transmissão de dados, mas ela tende a ser menor do que a dos meios com fio mais modernos. Além da velocidade, pode haver custos de conexão através de meios sem fio. Por esses problemas, é necessário minimizar os dados a serem transmitidos de/para os computadores móveis. Outro fator importante diz respeito às fontes de energia utilizadas por computadores móveis. As baterias ou pilhas tem tempo de vida limitado, e não há perspectiva de aumento significativo nos próximos anos. Nesse caso, enfrenta-se principalmente um problema físico, pois a carga é proporcional ao tamanho da bateria, e deseja-se criar dispositivos pequenos e leves. Para economizar energia são usados os modos de espera e desconectado. Mais informação sobre sistemas de computação móvel pode ser encontrada em [EJF95], [IB93], [JAA99] e [NS95].

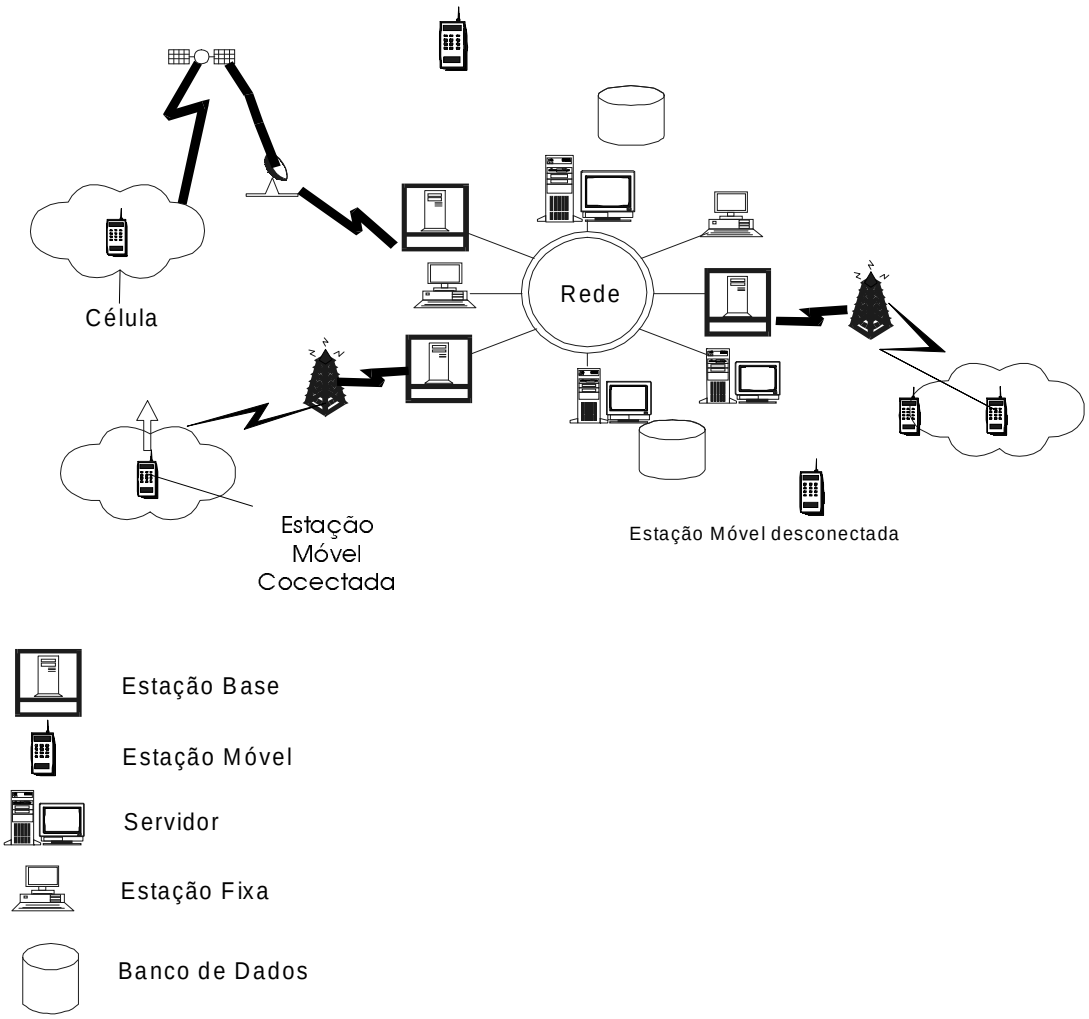


Figura 1 - Modelo de referência

Os programas dos computadores móveis estarão interessados nos dados armazenados em um banco de dados, localizado na rede. Os dados serão obtidos através de *consultas* ao banco de dados. Neste trabalho, vamos supor que o sistema gerenciador de banco de dados seja baseado no Modelo Relacional ou no Modelo Objeto-Relacional, capazes de interpretar comandos na linguagem de consulta SQL ou outra de poder equivalente. Normalmente, o termo linguagem de consulta é usado para linguagens que não só consultam mas também modificam os dados, através de operações de inserção, atualização e exclusão de dados. Permitiremos consultas e modificações de dados em nosso problema. Os conceitos sobre bancos de dados relacionais e objeto-relacionais podem ser encontrados em [EN00] e [SKS99].

Os dados podem estar distribuídos na rede ou podem residir em um único servidor de banco de dados. Do ponto de vista de nosso problema, faremos a suposição que essa distribuição será *transparente*, isto é, o sistema fará consultas a apenas um banco de dados, de estrutura definida, sem a preocupação sobre a alocação dos dados. Esse problema só

precisará ser considerado no momento de estimar os tempos gastos pelas consultas, que trataremos adiante. Podemos encontrar mais sobre bancos de dados distribuídos em [CP84] e [OV99]. Do ponto de vista de distribuição de dados, encontramos dois modelos envolvendo a computação móvel: "single hop[2]", onde o banco de dados está distribuído somente por estações fixas e "multi hop", onde os dados pode estar armazenados em estações móveis. Consideraremos neste estudo o modelo "single hop".

Vamos supor que o usuário móvel dispõe de equipamentos capazes de fornecer a sua posição geográfica, como o GPS[3]. Podemos também obter outras informações, como a velocidade de deslocamento e a sua direção através do mesmo sistema. Se isso não for possível, podemos também obter a direção e sentido de deslocamento através de análise da posição do usuário após alguns momentos. A velocidade normalmente é disponível em automóveis, aviões e outros veículos através do velocímetro. Se o GPS não estiver disponível, podemos obter a posição aproximada do usuário através de triangulação das antenas que captam o sinal, ou por outras técnicas de localização. A direção pode ser obtida por uma bússola, ou ser fornecida pelo usuário.

4 Trabalhos Relacionados

O problema de consultas envolvendo computação móvel e posição do usuário foi levantado diversas vezes anteriormente em [AK93], [DH95] e [IB93]. Nesses trabalhos, são enumerados diversos problemas de computação móvel e bancos de dados. Dentre os problemas citados podemos destacar as transações móveis, as interfaces de usuário limitadas, com telas pequenas e de resolução limitadas, como celular, PDA, etc., O problema de localização de dados, quando é usado o modelo onde o banco de dados ou parte dele pode estar distribuído nos computadores móveis também é citado. Finalmente, também é colocada a necessidade de melhora das linguagens de consulta para consultas envolvendo localidade. Uma grande preocupação é a de obter resultados de forma a minimizar os tempos de transmissão.

Outro problema relacionado é o de consultas sobre objetos móveis em sistemas de informação geográfica (GIS). Encontramos mais sobre esse assunto em [HT97], [SJ+00] e [FG+00]. Nesse caso, normalmente são feitas consultas através de interfaces gráficas, pois poderíamos ter resultados imprecisos, devida a movimentação dos objetos. Modelos de Entidades e Relacionamento estendidos para capturar a mobilidade dos objetos são propostas nestes trabalhos. Em nosso caso, as consultas serão sobre objetos estáticos, com usuário que executa as consultas possuindo mobilidade.

Um assunto relacionado com a variável aqui é a variável agora, em bancos de dados temporais [FM98] ou não temporais [CL+97]. Neste último trabalho, supõe-se a possibilidade de armazenarmos um valor especial no banco de dados, agora. Esse valor também pode ser usado em consultas. Um exemplo do uso desses valores seria o seguinte: em um sistema sobre funcionários, poderíamos armazenar a informação histórica sobre o salário do funcionário. Neste caso, poderíamos armazenar valores do salário e data em que

[2] Esse termo representa o número de conexões sem fio que são percorridas. Tais conexões são referidas como "pulos" (hop).
[3] Global Positioning System - Sistema de localização precisa através de satélites.

esse salário vigorou. No caso do salário atual, poderíamos armazenar como data atual agora. Podemos então, por exemplo, consultar o salário atual usando a variável agora na consulta. Notamos que há uma diferença fundamental entre o agora e o aqui: Se dois usuários fizerem uma mesma consulta ao mesmo tempo, exatamente no mesmo instante, o valor de agora deve ser o mesmo. A rigor, dois usuários não poderão estar no mesmo lugar no mesmo instante, o que faz com que o valor de aqui seja distinto para os dois. Em um modelo real, isso nem sempre será verdadeiro, pois dependendo da granularidade considerada, dois usuários podem ocupar a mesma posição. Esta pode ser potencialmente grande, como discutiremos adiante.

Encontra-se também uma grande atividade de pesquisa em consultas dependentes de localidade. Em [LGS00] são discutidas consultas mais gerais que nosso caso, com objetos móveis sendo consultados por usuários móveis. Há soluções em que o padrão de movimentação do usuário pode ser fornecida para cálculo das consultas. A preocupação maior nesse caso é com a minimização de comunicação. Nesse estudo não há a preocupação específica com aqui, mas com dados que podem variar com a mobilidade dos objetos ou usuários.

5 Consultas envolvendo a variável aqui

Primeiramente, devemos definir precisamente o que queremos dizer com a variável aqui, uma vez que apesar de possuir o mesmo nome, se feita por diversos usuários em posições diferentes, tem valores diferentes. Então, precisamente, aqui quer dizer a posição geográfica do usuário que está fazendo uma consulta com essa variável em um determinado instante. Esse tipo de variável, denominada *demonstrativa* é discutido em [KAP89]. Outros exemplos de demonstrativos incluem “eu”, “aí”, “você”, entre outras.

Quando um usuário móvel executa uma consulta, ele pode estar interessado em uma informação que depende de sua posição em diversos momentos. Vamos enumerar algumas das situações de possível interesse:

- O usuário deseja saber se há algum local fixo, como posto de gasolina, no seu caminho. Ele fará uma consulta como: "Qual o posto de gasolina mais próximo daqui ?"
- Pode-se desejar também uma consulta que envolve posições futuras em relação à posição atual do usuário, como: "Liste os restaurantes chineses pelos quais eu vou passar nos próximos 30 minutos (a partir daqui)". A posição do usuário está implícita nesta consulta, bem como a direção do movimento.
- Um guarda de trânsito está se encaminhando para uma emergência, e ele quer relatar um outro fato de relevância para o sistema de informação de trânsito, como: "Aconteceu um acidente sem vítimas aqui". Outra utilidade desse tipo de consulta seria quando utilizamos um automóvel, por exemplo, para transmitir informação de velocidade sobre uma via. Ele poderia fazer a seguinte atualização: “Estou aqui a uma velocidade de 70 km/h”. Quando a atualização se encerrar no banco de dados, o valor que deve ser armazenado é o do local onde foi transmitida a informação.
- Um caminhão, carro ou outro veículo quer informar a sua posição atual. A consulta seria do tipo "Eu estou aqui". Essa informação pode ser exibida em uma tela para acompanhamento visual, para a previsão de colisões, detecção de desvio de rotas ou

outro tipo de acompanhamento em tempo real. Note que apesar de um dos interessados na consulta ser móvel, as consultas sobre esse objetos serão feitos normalmente em

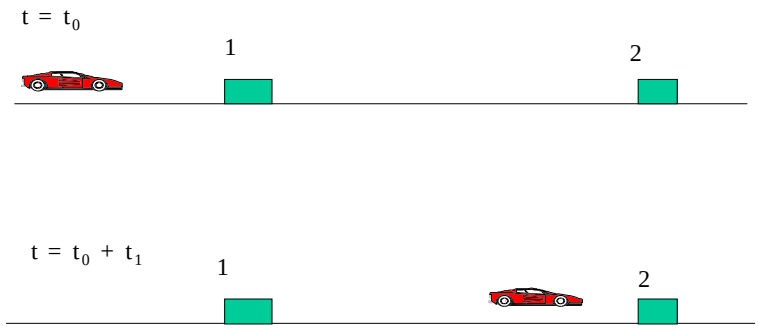


Figura 2- Consulta ao posto de gasolina

estações fixas, o os objetos serão móveis. Esse tipo de consulta é o tipo citado nos trabalhos relacionados

Em todos esses casos, o valor da posição do usuário, aqui, é usada de forma direta ou indireta. Em cada caso, temos um interesse diferente no valor de aqui. Ressaltamos não que elaboramos consultas sobre objetos ou locais móveis. Quem tem mobilidade neste estudo é o usuário.

Quando o usuário faz a consulta, diversos fatores contribuem no tempo de resposta do sistema:

- O tempo de transmissão de dados pode ser alto. A baixa taxa de transmissão dos meios sem fio, a baixa confiabilidade, que pode gerar retransmissões freqüentes, influem nesse tempo.
- A execução da consulta ao banco de dados pode ser demorada, principalmente no caso em que as cidades são grandes, com grande número de ruas e informações. A carga do servidor ou servidores também pode influir na demora da resposta.
- O usuário pode estar desconectado da rede, por vontade própria ou não. O tempo de desconexão pode ser alto, e a reposta pode demorar mais para chegar ao usuário que a solicitou. Em casos extremos, o tempo de desconexão pode ser tão alto que o resultado perde o sentido. Isso deve ser regulado pelo sistema, através de um tempo limite para a resposta.

Aliado a todos esses fatores, o usuário se encontra em movimento, o que faz com que a sua posição, e conseqüentemente o valor de aqui mude a cada momento. Essa mudança pode alterar de forma importante os resultados desejados da resposta, especialmente se a velocidade for elevada. Suponhamos que no primeiro exemplo de consulta, do usuário perguntando sobre o posto, o sistema demore algum tempo para responder. Ao receber a

resposta, o usuário pode ter passado do posto mais próximo que o sistema indicar, que é o posto 1, eventualmente sem chances de fazer um retorno na figura 2, vemos a situação ilustrada. Seria desejável que ele recebesse a resposta a tempo de ir ao local que o sistema indicar. Ou seja, que a resposta fosse o posto 2. Esse caso é o ilustrado na Figura 2.

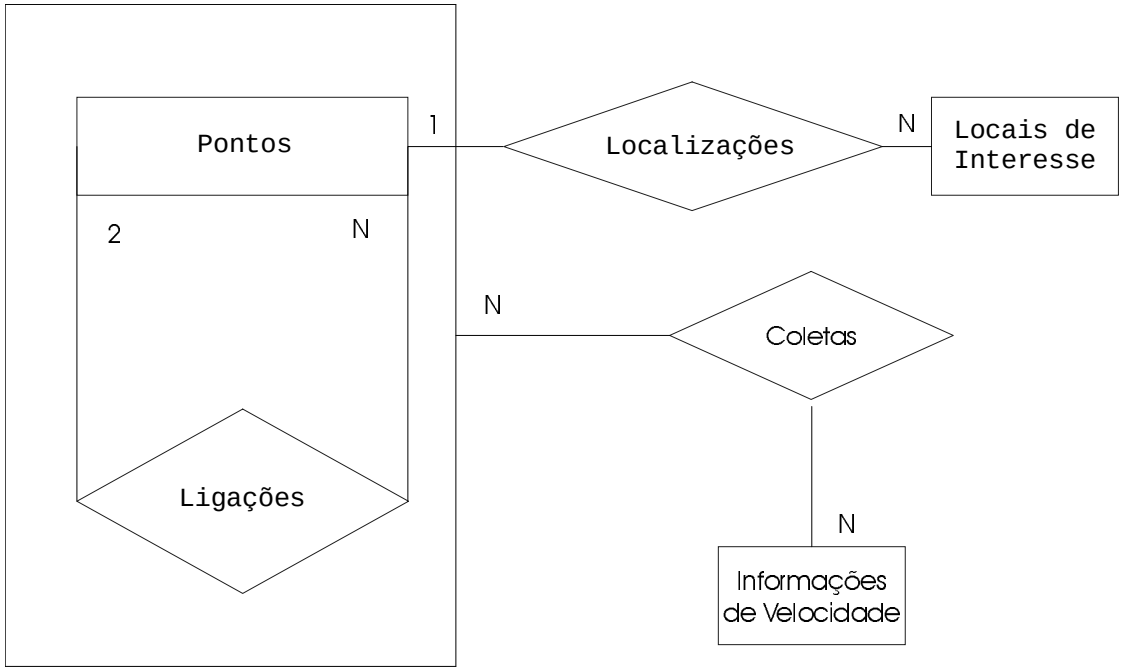


Figura 3 – Esquema ER

5.1 Uma extensão da linguagem SQL para a variável aqui

A linguagem SQL é a linguagem de maior uso entre os bancos de dados comerciais. Neste trabalho, aumentaremos a capacidade da linguagem SQL para tratar a variável aqui. São necessárias algumas hipóteses específicas em relação ao banco de dados para que a variável aqui seja utilizada:

- Sempre que a variável aqui for comparada com outra, o será com um atributo correspondente à posição. O mesmo se aplica às atualizações de dados: aqui só pode ser atribuído a um atributo de localização.
- Uma tabela que contém um atributo de posição sempre precisa estar envolvida na consulta, na cláusula from da SQL.

Na figura 3, mostramos um esquema ER (entidades e relacionamentos), e modelo relacionais, correspondentes aos que serão usados na implementação inicial.

A posição corresponderá a um valor numérico inteiro, cujo valor corresponderá a um código de ponto na tabela de pontos (de referência)

A variável aqui pode aparecer portanto nas cláusulas select e where, além das inserções, atualizações e exclusões de dados. Por motivos estéticos, a variável aqui será

usada em inglês dentro da consulta (here). Por enquanto, consideraremos somente um tipo da variável. Mostraremos a seguir, alguns exemplos de consultas SQL com a variável aqui.

```
select here /* nesta consulta esperamos apenas obter a */
from pontos /* posição do usuário */

select dist(posicao) /* uso da função dist */
from pontos inner join locais /* para descobrir locais */
where locais.tipo = 1         /* próximos */

insert into pontos( código, posição ) /* atualizando */
  values( 1000, here )                /* a posição */
```

Criamos uma função especial dist, que calcula a distância entre o ponto selecionado e a posição atual (aqui). A configuração do sistema incluirá informação sobre quais são as tabelas que tratam de posição do usuário, assim como quais são os atributos correspondentes a posições físicas.

5.2 As possíveis semânticas da variável especial Aqui

Observando-se as situações descritas na seção anterior, podemos considerar os seguintes valores possíveis para aqui:

- A posição do usuário no momento em que a consulta foi iniciada pelo usuário. Chamaremos esta posição de aqui_i (i: início da transação).
- A posição do usuário no momento em que a transação é confirmada (commit). Esta posição será chamada aqui_c (c: confirmação da transação)
- Finalmente, a posição do usuário no momento em que ele recebe o resultado da consulta será chamado aqui_f (f: final da transação).

Cada um desses valores corresponde à resposta desejada em diversas situações. Um primeiro exemplo é quando fazemos uma consulta, envolvendo aqui, como por exemplo: “Qual o posto de combustível mais próximo de onde eu estou ?”

Supondo que o usuário continua em movimento enquanto a consulta é processada, seria interessante que a resposta fosse aqui_f. Se utilizássemos o aqui_i, poderíamos ter uma resposta errada, como no exemplo da figura 2.

Se a consulta envolve uma atualização, podemos ter outro resultado de interesse. Por exemplo, se um usuário ou policial quer informar o sistema que há algum evento no local onde ele está passando, a escolha deve ser aqui_i, por motivos óbvios. Um exemplo de consulta desse tipo é: “Aconteceu um acidente grave aqui”. Nesse caso, inclusive, o aqui_f é de pouco interesse. Não faz diferença o local onde o usuário se encontra quando ele sabe que a atualização foi bem sucedida.

Existe um outro tipo de atualização em que o aqui_c parece ser mais adequado. Num sistema de monitoramento, por exemplo, vamos supor que um caminhão ou automóvel faça com freqüência atualizações do tipo “Eu estou aqui”. Neste caso, se o valor gravado for o aqui_c, se alguém fizer uma consulta logo depois da atualização, terá a resposta mais próxima da realidade do que se fosse utilizado o aqui_i. Podemos ter dificuldades para calcular aqui_c e aqui_f em caso de atualizações em bancos de dados distribuídos. Nesse caso, se forem usados protocolos de confirmação do tipo 2PC ou 3PC (confirmação em 2 ou 3 fases), pode ser absolutamente imponderável o tempo que o dado levará para ser confirmado, devido a falhas de rede, quedas do servidor, etc. Poderíamos estabelecer margens de erro aceitáveis para os valores gravados, e após a confirmação, verificar os valores.

5.3 O cálculo dos diversos valores de aqui

A posição do usuário após alguns momentos é dependente de muitos fatores, dentre os quais podemos citar:

- Velocidade
- Aceleração.
- Tipo do movimento, como uniforme, uniformemente variado, etc.

Direção

- Sentido
- No caso de nossa aplicação de referência, o trajeto
- Trajetória, ou seja, que tipo de movimento o agente está executando, como linear, circular, etc.
- Tempo que a transação leva para ser executada.

Na definição da arquitetura do sistema fizemos a suposição que o sistema tem acesso a uma forma razoavelmente precisa de fornecer sua localização, através de instrumentos do tipo GPS, por exemplo. Esse tipo de equipamento está se tornando comum e mais acessível, sendo uma hipótese razoável de ser exigida. Em alguns tipos de sistema, podemos fornecer informação de outra forma, como por exemplo, os quilômetros de uma estrada, ou posição em relação a ruas de uma cidade. Nestes casos, pode haver uma necessidade de ajuda do usuário para o fornecimento da posição. Essas ajudam incluem a informação do trajeto pelo usuário, que ajuda a eliminar possibilidades de movimentação, ou a informação do destino, que apesar de dar menos informação que o trajeto, também ajuda no descarte de posições futuras.

Vamos simplificar um pouco algumas das variáveis, sem muita perda de precisão. Em vez de utilizar velocidade e aceleração variáveis, vamos considerar a velocidade média durante o tempo de execução da consulta, simplificando o tipo de movimento para uniforme. Assim sendo, para calcular a distância percorrida pelo usuário (s) basta multiplicar a velocidade (v) pelo tempo decorrido (t):

$s = vt$

Outra restrição é para a trajetória. Em alguns tipos de sistemas, não se pode definir uma trajetória provável para o usuário, como por exemplo em um sistema de monitoramento de movimento de algum animal, ou outro objeto voador sem rotas definidas. Nesses casos os agentes podem mudar de direção a qualquer momento, e a única coisa que se pode prever é que o usuário pode estar no máximo a uma determinada distância da posição de início da transação, ou seja, dentro de um círculo de raio vt, como exibido na figura 4.

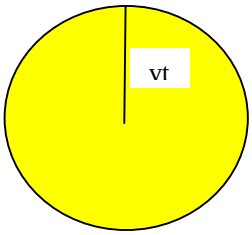


Figura 4: área em que o usuário pode se encontrar apos o tempo t

Nesse caso, fica muito difícil fazer uma previsão da posição final, uma vez que há infinitas possibilidades.

Neste estudo, estaremos interessados particularmente em problemas envolvendo o tráfego urbano, e nesses sistemas, o tipo de movimento é um pouco mais restrito. Os carros e caminhões podem trafegar somente através de ruas e estradas, e podemos modelar este

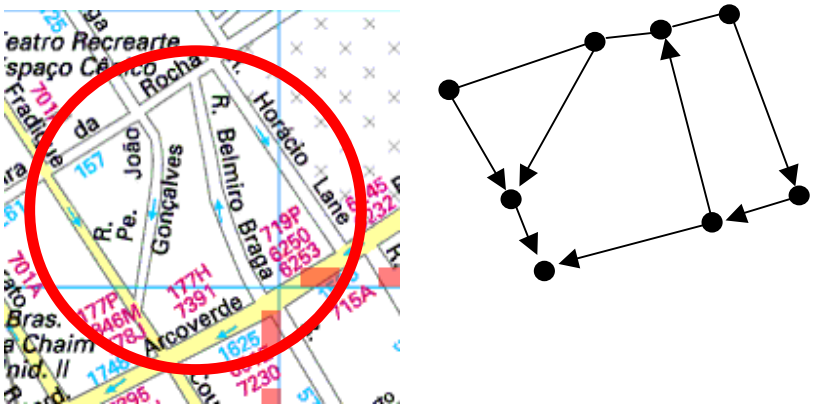


Figura 5 - Techo de um mapa e grafo misto correspondente

tipo de movimento através de um grafo, onde cada vértice adjacente a um determinado vértice, correspondente à posição atual, corresponde a um possível destino do veículo. Esta representação também se justifica pela não necessidade de uma precisão absoluta na resposta, além de não haver interesse em uma resposta milimétrica. Um erro na ordem de alguns metros pode ser aceitável, se estivermos lidando com informações sobre tráfego de automóveis. O grafo pode ainda conter informações sobre o fluxo de veículos atual em suas arestas, e destinos ou pontos de referência nos vértices. O tipo de grafo usado será o grafo *misto*, com algumas arestas com uma direção, representando ruas com mão única, e arestas

sem direção, que denota uma rua de mão dupla. Na figura 5, mostramos um trecho de um mapa e seu grafo correspondente. Podemos, sem perda de informação significativo, usar somente grafos dirigidos para nosso estudo, uma vez que somente utilizaremos algoritmos para encontrar caminhos no grafo.

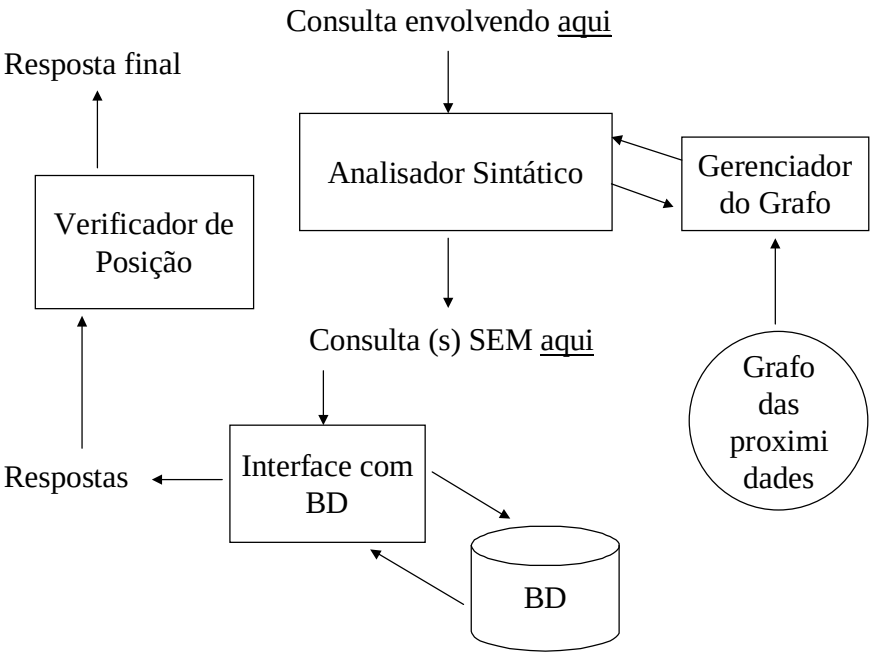
Desta forma, o número de pontos onde o usuário pode estar localizado no recebimento da resposta se torna finito. Nos resta o cálculo do tempo decorrido desde o início da consulta e o momento em que ocorre a semântica de aqui de interesse.

No caso mais demorado, quando nos interessa o valor de aqui_f, o tempo de execução será composto pelos seguintes valores:

- t_1 = O tempo que a consulta leva para chegar até a estação base.
- t_2 = Tempo decorrido até o sistema enviar a consulta ao computador onde estão os dados de interesse.
- t_3 = O tempo de execução da consulta no banco de dados
- t_4 = envio da resposta à estação base correta
- t_5 = envio da resposta ao usuário móvel

Em e t_2 , t_4 embutimos um tempo extra de processamento e tratamento das informações recebidas.

A figura a seguir mostra a arquitetura de nosso sistema:



5.4 Formalização

Podemos considerar que aqui pode ser avaliado através de duas funções. A primeira recebe a consulta, a posição do usuário, a vizinhança onde ele se encontra, informações sobre a movimentação, e tempo estimado de espera. Esta função retorna um conjunto de valores que são as respostas prováveis. Essas respostas prováveis serão filtradas através da outra função, que escolherá o valor único da resposta. A primeira função corresponde ao pré-processamento, enquanto que a segunda ao pós-processamento.

Mais precisamente, teremos:

F: Q, BD, P, M, T U $\rightarrow 2^P$

F(q, bd, p, m, t, u) $\subseteq$ P obtém posições possíveis para o usuário

Onde:

q – consulta envolvendo aqui

bd – banco de dados

p – posição atual do usuário

m – movimentação do usuário, incluindo direção, sentido, velocidade, destino, etc.

t – tempo previsto para execução da consulta

u – usuário móvel.

G: $2^P \rightarrow$ P

G(p, u) Escolhe a posição mais precisa do usuário

Em linhas gerais, nossa implementação será feita através dessas duas funções. A função F retorna um conjunto de valores possíveis para a resposta, enquanto que G escolhe qual o valor mais próximo do usuário móvel.

5.5 Proposta de implementação

As consultas envolvendo a variável aqui seria de grande valia para a aplicação de referência do projeto SIDAM [ES+99], uma vez que permitiria que usuários fizessem consultas sobre trânsito sem se importar o local onde ele está, pois eventualmente podem até estar perdidos.

É desejável que haja *transparência* na variável aqui, ou seja, que o usuário não precise dizer ao sistema sobre qual aqui ele está interessado. De acordo com a consulta, então, nosso sistema pode escolher a semântica correta para cada caso. O programador poderá, entretanto, escolher quais dos valores que ele deseja utilizar na aplicação. Serão dadas opções de valores a serem usados na configuração do sistema, por exemplo, sempre aqui_i, escolhido pelo sistema, escolhido pelo programador, etc.

Uma forma de atribuir um valor de aqui seria usar a primeira solução, que é usar o aqui_i. Quando a velocidade do veículo for grande, este método tenderá a ter resultados mais imprecisos. Em se tratando de tráfego urbano (especialmente São Paulo, em hora de pico) espera-se que as velocidades sejam menores, reduzindo a imprecisão desta alternativa.

O sistema tentará classificar as consultas, quando solicitado, de forma de alguns tipos de consulta usem como aqui aqui_i, aqui_c ou aqui_f. Quando temos uma consulta que envolve aqui, como no caso do posto de gasolina, é adequado o uso de aqui_f. Quando há uma atualização de dados, se esta tratar sobre a posição do próprio usuário, deveríamos usar o aqui_c, senão, se a modificação for a respeito de um objeto que não é móvel (um acidente), devemos usar o aqui_i. Essas duas formas são difíceis de detectar automaticamente, uma vez que ocorrem no mesmo tipo de operação no banco de dados.

Quando uma consulta for executada, o computador móvel deverá enviar a consulta, posição e o horário em que a consulta foi executada. Note que a posição é aqui_i. Se o computador móvel estiver ligado a um GPS, por exemplo, o cálculo de aqui_i é trivial.

Para calcularmos aqui_c e aqui_f precisamos usar estimativas de tempo gasto na execução da consulta e transmissão de dados para o usuário, ou seja ($t_1 + t_2 + t_3 + t_4 + t_5$). Normalmente, os otimizadores de consultas conseguem estimativas de tempo para as consultas durante a otimização. Estes tempos podem ser usados para estimar a posição. Poderão ser usadas outras informações, vindas dos sistemas de monitoramento, que podem ajudar a estimar tempos de transmissão, localização, tráfego de rede, entre outros atrasos.

Dada a posição do usuário podemos construir a partir do grafo que representa os possíveis destinos os vários locais em que o usuário poderá estar no momento em que ele receber a consulta. Os tempos de execução de consultas e transmissão de dados precisam ser a princípio previstos, mas a medida em que o sistema for sendo usado pode-se medir os tempos efetivos, adaptando o sistema ao uso real.

A implementação será feita com um processador de consultas SQL que receberá uma consulta SQL, e fazendo uma interpretação de seu conteúdo. O interpretador poderá gerar várias consultas que serão feitas ao sistema, e processar esses resultados, usando o grafo anteriormente citado, efetuando uma busca em largura. Nosso módulo, pode então, devolver respostas, de acordo com configurações escolhidas.

O processador de consultas será implementado com auxílio de ferramentas da família do Lex e Yacc [LMB92], mais especificamente o Jflex e Byacc, que geram código Java.

O número de respostas a serem devolvidas pode ser configurado, de acordo com o poder computacional do computador móvel. Quanto mais poderoso, podemos efetuar uma busca mais extensa, para aumentar a probabilidade de encontrar a reposta correta. Se o computador móvel for pouco poderoso, devolveremos um número menor de respostas, para diminuir o processamento necessário para escolher a resposta correta.

Figura 6: Usuário móvel fazendo uma consulta em uma rua

Será estabelecido também um tempo limite (“time-out”) para execução do processamento da consulta. Isso é importante, pois poderíamos obter um número de respostas maior, aumentando o tempo de processamento. E aumentando o tempo de processamento também teríamos um número maior de respostas.

Vamos supor que o usuário faça essa pergunta ao sistema. Primeiramente, será transmitida ao sistema a posição do usuário, correspondente a aqui_i. Todas as informações sobre o movimento também devem ser enviadas, juntamente com a consulta. Velocidade e direção, se possível. Através da informação sobre seus possíveis destinos, de acordo com a granularidade possível no mapa que está no sistema, elaboraremos uma série de consultas, tentado achar o local mais próximo. Isso pode ser feito se tivermos as informações sobre os locais principais da localidade, através de uma consulta possível de ser elaborada em SQL. Feito isso, serão devolvidos uma série de resultados que para serem corretos dependem do local onde o usuário recebe as respostas. O usuário então seleciona a reposta que mais se aproxima da posição em que ele se encontra.

Vamos supor que o usuário esteja em uma determinada rua, no ponto preto na figura 5.

No desenho está indicada a posição do usuário no momento em que ele efetua a consulta, aqui_i. Os postos estão indicados pelos quadrados. Vamos supor que o sistema estima o tempo gasto para todo o processamento, e conclui que ele pode se mover aproximadamente por duas quadras. Nesse caso, os seguintes destinos podem ser um exemplo dos destinos possíveis. Vamos supor que o sistema já levou em consideração eventuais mãos contrárias, retornos, etc.

Neste caso, faremos uma consulta levando em conta as três posições possíveis. Cada uma delas pode devolver uma resposta diferente, que será escolhida de acordo com a posição real em que o usuário se encontrar quando ele receber a resposta.

Podemos então fazer três consultas:

- Qual o posto mais próximo do ponto 1
- Qual o posto mais próximo do ponto 2
- Qual o posto mais próximo do ponto 3

Podemos então receber a seguinte resposta, em nosso exemplo:

- O posto mais próximo de 1 é a ou b
- O posto mais próximo de 2 é a
- O posto mais próximo de 3 é b

Podemos simplificar esta resposta para ser envidada da seguinte forma:

“Se você estiver em 1 ou 2, vá para o posto a, se você estiver em 1 ou 3, vá para b”.

Desta forma, é possível escolher o posto mais próximo, de acordo com a posição atual. Se o usuário não estiver em nenhum dos pontos, pode-se escolher o mais próximo, como resposta aproximada. As informações sobre trajeto e destino podem ainda, serem usadas para descartar respostas muito pouco prováveis. Todas essas etapas podem ser executadas no pós processamento, que corresponde ao cálculo da função G.

Podemos resumir o processo da seguinte forma:

```
Recebe consulta em SQL estendida
Interpreta consulta, através de análise sintática:
Se a consulta não contém aqui, execute-a normalmente
Senão
  Gera consultas equivalentes que não envolvem a variável aqui
  Resposta ← ∅
  Para cada resposta r_i obtida por essas consultas
    /* Neste trecho, é feito o pós-processamento */
    Confronte o aqui obtido com a posição atual do usuário
    Se a resposta for igual a posição
      Resposta ← Resposta ∪ r_i
Devolva Resposta
```

5.6 Simulação

Testar esse sistema é uma tarefa difícil de ser feita em condições reais. Seria necessário testar consultas a diversas velocidades, usando computadores móveis de diferentes configurações, utilizando diversos caminhos, efetuando diferentes consultas. Algumas situações seriam de difícil reprodução, tornando o teste real inviável.

O projeto SIDAM possui um simulador, denominado MobiCS [RE00] e [RE00a], que simula diversas condições de envio e recebimento de mensagens através do sistema, com o uso de simulações de canais sem fio. O MobiCS foi desenvolvido principalmente para testar protocolos para computação móvel, e para esse fim é pouco importante saber a posição exata de um usuário móvel. Basta saber em que célula o usuário se encontra, o que o associa à estação base correspondente. A mobilidade do usuário se torna importante somente quando há a troca de célula.

Para nossa simulação, é importante ter uma posição simulada do usuário dentro do grafo correspondente ao local onde ele se encontra. Podemos fazer isso através de extensões ao script usado pelo MobiCS. O nosso problema envolve tempo real, e portanto, em alguns pontos da simulação, é necessário que as consultas sejam executadas sem interrupção. Podemos marcar tais pontos, para que sejam mostrados resultados logo após esses pontos críticos. Podemos então definir que o ambiente de simulação possui uma cópia do grafo correspondente às redondezas do local onde a estação móvel simulada está localizada. O local corresponderá a um dos vértices do grafo. Podemos então, mover o usuário conforme um padrão pré-estabelecido de um vértice ao outro, colocando atrasos nesse deslocamento que corresponderão à velocidade imprimida pela estação móvel. Podemos estabelecer um relacionamento entre um subgrafo das redondezas e as respectivas células, para efeito de teste dos protocolos.

A simulação seria feita portanto com seqüências de comandos e movimentações intercaladas. Quando a estação móvel receber as respostas de suas consultas, será possível comparar a resposta obtida com a posição atual na simulação. Segue um exemplo de script de simulação proposto.

```
Mh1.sendquery( q ); // assíncrono. Envia a consulta
Delay(50); // demora 50 ms
Mh1.moveto( 3 ); // para se mover para outro ponto
Delay(30)
Mh1.moveto( 4 );
```

Quando a resposta for obtida, podemos comparar o que o sistema fornece como resposta com a posição do usuário no final da simulação. A velocidade e o trajeto que o usuário faz ficam implícitos nos tempos de atraso fornecidos e pontos percorridos.

6 Conclusões

Neste semestre demos seqüência ao trabalho de implementação, através da construção de um, compilador de consultas, além de outros componentes necessários. Foi definido também um esboço de definição formal para nosso problema. Além disso foi detectada a necessidade da construção de um simulador para nosso sistema. Ressaltamos que artigos relacionados a nossa tese foram submetidos ao SBBD 2001 (Simpósio Brasileiro de Bancos de Dados) e ao WCSF 2001 (Workshop de comunicação sem fio e computação móvel), além de contribuir de forma significativa para o avanço de nossa tese. Foi também apresentado um seminário para o projeto SIDAM, com parte integrante da disciplina de seminários.

7 Referências bibliográficas atuais

[AK93] Alonso, R., Korth, H. F., Database System Issues in Nomadic Computing, *SIGMOD Record*, **22**(2):388-392, Junho de 1993

[CL+97] Clifford, J., Dyreson, C., Isakowitz, T., Jensen, C., Snodgrass, R., On the Semantics of "Now" in Databases, ACM Transactions on Database Systems, **22**(2):171-214, Junho de 1997.

[CP84] Ceri, S., Pelagatti, G., Distributed Databases: Principles and Systems, McGraw-Hill, 1984.

[DH95] Dunham, M. H., Helal, A., Mobile Compututing and Databases: Anything New?, *SIGMOD Record*, **24**(4):5-9, Dezembro de 1995

[EJF95] Elmagarmid, A., Jing, J., Furukawa, T., Wireless Client/Server Computing for Personal Information Services and Applications, *SIGMOD Record*, **24**(4):16-21, Dezembro de 1995.

[EN00] Elmasri, R., Navathe, S., Fundamentals of Database Systems, 3rd Ed. Addison-Wesley, 955pp. 2000.

[ES+99] Endler, M. Silva, D. M., Silva, F. J. S., Rocha, R. C. A., Moura, M. A., Project SIDAM: Overview and Preliminary Results, disponível em http://www.ime.usp.br/~sidam/documents.html.

[FG+00] Forlizzi, L., Güting, R., Nardelli, E., Schneider, M., A Data Model and Data Structures for Moving Objects Databases, *SIGMOD Record*, **29**(2):319-330, Junho de 2000.

[FM98] Finger, M., M^{c}Brien, P., On the Semantics of 'Current-Time' in Temporal Databases, 1998.

[GB+00] Güting, R., Böhlen, M., Erwig, M., Jensen, C., Lorentzos, N., Schneider, M., Varzirgiannis, M.; A Foundation for Representing and Querying Moving Objects, ACM Transactions on Database Systems, **25**(1):1-42, Março de 2000.

[HT97] Hadzilacos, T., Tryfona, N., Na Extended Entity-Relationship Model for Geografic Applications, *SIGMOD Record*, **26**(3):24-29, Setembro de 1997.

[IB93] Imielinski, T., Bandrinath, B. R.; Data Management for Mobile Computing, *SIGMOD Record*, **22**(1):34-39, Março de 1993

[IB93] Imielinski, T., Bandrinath, B. R.; Mobile Wireless Computing, *Comm. of ACM*, **37**(10):18-28, Outubro de 1994

[JAA99] Jing, J., Abdelsalam, H., Ahmed, E., Client-Server Computing in Mobile Environments; ACM Computing Surveys, **31**(2):117-157, Junho de 1999.

[KA89] Kaplan, D., Demonstratives, em Themes from Kaplan, pp 481-563, ed. Almog et. Al, Oxford University Press, 1989.

[LMB95] Levine, J., Mason, T., Brown, D.; Lex & Yacc, 364pp., O'Reilly, 2^{a} Edição, 1995.

[NF00] Nassu, E.; Finger, M.; A Semântica de Aqui em Sistemas Transacionais Móveis; Anais do Workshop SIDAM, IME-USP; Novembro de 2000.

[NS95] Brian D. Noble, Satyanarayanan, M., A Research Status Report on Adaptation for Mobile Data Access, *SIGMOD Record*, **24**(4):10-15, Dezembro de 1995

[OV99] Özsu, M., Valduriez, P., Principles of Distributed Database Systems, 2nd ed., Prentice Hall, 1999.

[RE00] Ricardo Couto A. da Rocha, Markus Endler. Flexible Simulation of Distributed Protocols for Mobile Computing, Proc. of 3rd Int. Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWIM), pp. 123--126, Boston. August 11th, 2000.

[RE00a] Ricardo Couto A. da Rocha, Markus Endler. Um Simulador de Protocolos Distribuídos para Computação Móvel,2nd Brazilian Workshop on Wireless Communication (2o. Workshop de Comunicação sem Fio- WCSF), pp. 33--48, UFMG, Belo Horizonte, May 2000.

[SJ+00] Saltenis, S., Jensen, C., Leutenegger, S., Lopez, M., Indexing the Positions of Continuously Moving Objects, *SIGMOD Record*, **29**(2):331-342, Junho de 2000.

[SKS99] Silberschatz, A., Korth, H., Sudarshan, S., Sistemas de Banco de Dados, tradução, 3ª edição, Makron Books, 1999.

[YC+00] Yuen, J. C., Chan, E., Lam, K., Leung, H.; Cache Invalidation Scheme for Mobile Computing Systems with Real-time Data; *SIGMOD Record*, **29**(4):34-39, Dezembro de 2000.