Scalable Content-Adressable Network

Pedro Miguel Martins Nunes

Tecnologias de Middleware 06/07
Curso de Especialização em Informática
Departamento de Informática
Faculdade de Ciências da Universidade de Lisboa

10.11.2006

Introdução

- **Tabela de dispersão**: estrutura de dados que mapeia chaves em valores
- **Content-Adressable Network** (**CAN**): tabela de dispersão distribuída, a nível da Internet
- Aplicabilidade da CAN: sistemas que requeiram um eficiente acesso a conteúdos, localizados numa infraestrutura de armazenamento distribuída
  - exemplo: Napster
    - transferência de ficheiros descentralizada (*peer-to-peer*)
    - localização de ficheiros efectuada de forma centralizada (via servidor)
    - falta de escalabilidade no processo de mapeamento dos nomes dos ficheiros (chaves) na sua localização no sistema (valores)
- Operações básicas realizadas na CAN: inserção, pesquisa e eliminação de pares (Key,Value)

Introdução

- **Tabela de dispersão**: estrutura de dados que mapeia chaves em valores
- **Content-Adressable Network** (**CAN**): tabela de dispersão distribuída, a nível da Internet
- Aplicabilidade da CAN: sistemas que requeiram um eficiente acesso a conteúdos, localizados numa infraestrutura de armazenamento distribuída
  - exemplo: Napster
    - transferência de ficheiros descentralizada (*peer-to-peer*)
    - localização de ficheiros efectuada de forma centralizada (via servidor)
    - falta de escalabilidade no processo de mapeamento dos nomes dos ficheiros (chaves) na sua localização no sistema (valores)
- Operações básicas realizadas na CAN: inserção, pesquisa e eliminação de pares (Key,Value)

Introdução

- **Tabela de dispersão**: estrutura de dados que mapeia chaves em valores
- **Content-Adressable Network** (**CAN**): tabela de dispersão distribuída, a nível da Internet
- Aplicabilidade da CAN: sistemas que requeiram um eficiente acesso a conteúdos, localizados numa infraestrutura de armazenamento distribuída
  - exemplo: Napster
    - transferência de ficheiros descentralizada (*peer-to-peer*)
    - localização de ficheiros efectuada de forma centralizada (via servidor)
    - falta de escalabilidade no processo de mapeamento dos nomes dos ficheiros (chaves) na sua localização no sistema (valores)
- Operações básicas realizadas na CAN: inserção, pesquisa e eliminação de pares (Key,Value)

Introdução

- **Tabela de dispersão**: estrutura de dados que mapeia chaves em valores
- **Content-Adressable Network** (**CAN**): tabela de dispersão distribuída, a nível da Internet
- Aplicabilidade da CAN: sistemas que requeiram um eficiente acesso a conteúdos, localizados numa infraestrutura de armazenamento distribuída
  - exemplo: Napster
    - transferência de ficheiros descentralizada (*peer-to-peer*)
    - localização de ficheiros efectuada de forma centralizada (via servidor)
    - falta de escalabilidade no processo de mapeamento dos nomes dos ficheiros (chaves) na sua localização no sistema (valores)
- Operações básicas realizadas na CAN: inserção, pesquisa e eliminação de pares (Key,Value)

Descrição genérica

- Espaço de coordenadas cartesianas virtuais (*d-torus*[1])
  completamente lógico, sem relação com qualquer sistema físico
- Dividido dinamicamente entre todos os nós do sistema
- Cada nó tem assignado uma zona no *torus*
- Este espaço virtual é usado para guardar pares de (K,V) da seguinte forma:
  - K é mapeada no ponto P do espaço de coordenadas, usando uma função de dispersão
  - (K,V) é armazenado no nó da zona onde se situa o ponto P
  - usando a mesma função de dispersão para mapear chave K no ponto P, a entrada (K,V) é obtida do nó da zona que contém P
  - se o ponto P não pertence à zona do nó requerente (ou nós vizinhos), o pedido tem de ser encaminhado pela CAN até chegar ao nó da zona onde se situa P

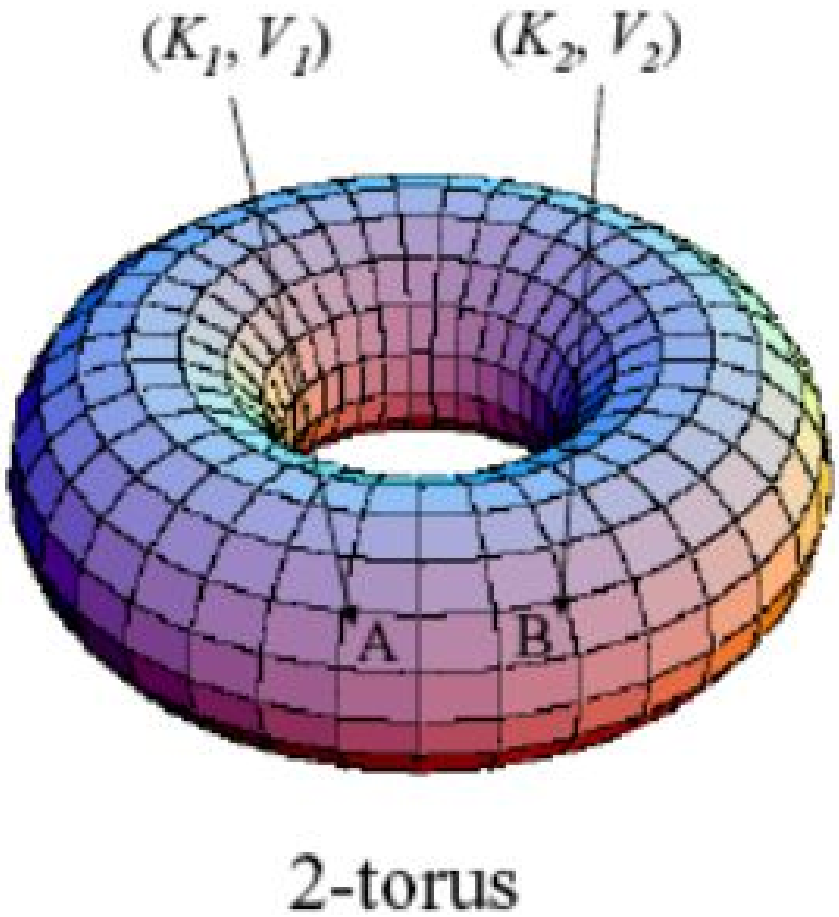
[1] Estrutura obtida da conexão entre os pontos finais e iniciais de uma grelha

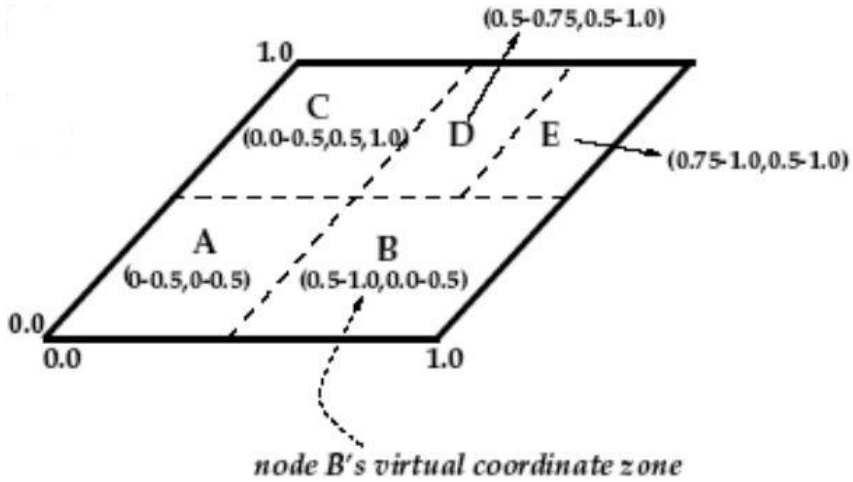
Descrição genérica

- Espaço de coordenadas cartesianas virtuais (*d-torus*[1])
  completamente lógico, sem relação com qualquer sistema físico
- Dividido dinamicamente entre todos os nós do sistema
- Cada nó tem assignado uma zona no *torus*
- Este espaço virtual é usado para guardar pares de (K,V) da seguinte forma:
  - K é mapeada no ponto P do espaço de coordenadas, usando uma função de dispersão
  - (K,V) é armazenado no nó da zona onde se situa o ponto P
  - usando a mesma função de dispersão para mapear chave K no ponto P, a entrada (K,V) é obtida do nó da zona que contém P
  - se o ponto P não pertence à zona do nó requerente (ou nós vizinhos), o pedido tem de ser encaminhado pela CAN até chegar ao nó da zona onde se situa P

[1] Estrutura obtida da conexão entre os pontos finais e iniciais de uma grelha

Espaço de coordenadas

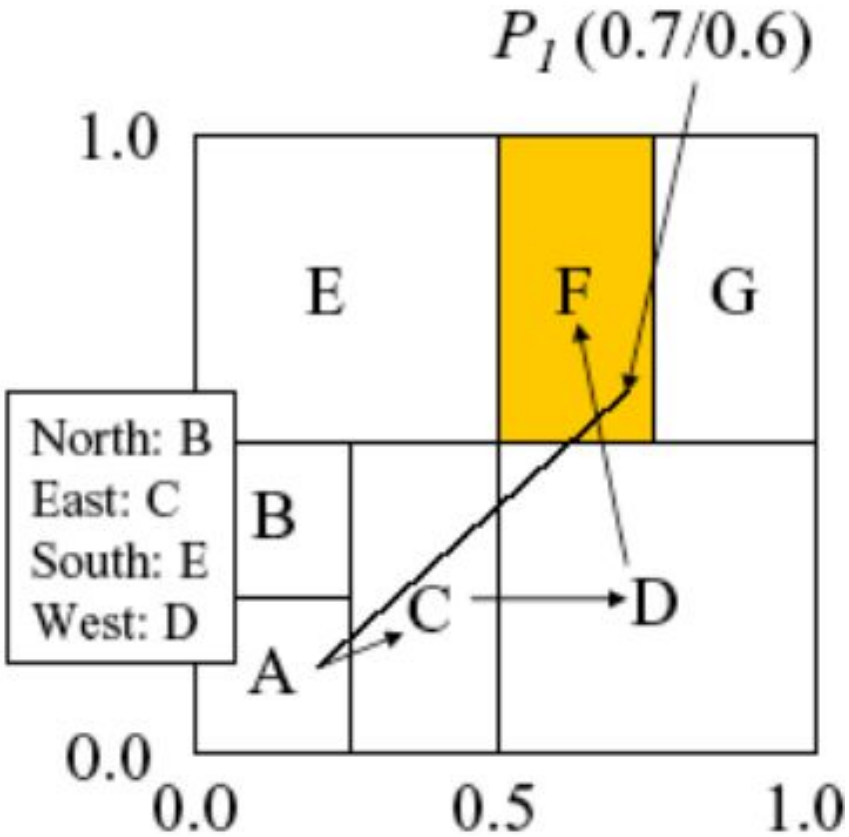




Encaminhamento na CAN

- Consiste em seguir linha directa através do espaço cartesiano, da coordenada de origem à de destino
- Cada nó mantém uma tabela de encaminhamento que alberga **endereço IP** e **coordenadas das zonas** dos nós vizinhos
- O encaminhamento de uma mensagem é efectuado através do nó com as coordenadas mais próximas das coordenadas de destino
- caso um ou mais nós vizinhos falhem, o nó pode encaminhar automaticamente a mensagem pelo melhor caminho seguinte
- Para um espaço de d dimensões dividido em n zonas:
  - nós mantêm $2d$ vizinhos
  - distância média de encaminhamento: $\frac{d}{4} n^{\frac{1}{d}}$

Encaminhamento na CAN



CAN: Adição de novo nó

1. Novo nó localiza um nó de arranque (*bootstrap*) usando DNS
   Nó de arranque mantem lista parcial de nós da CAN que acredita estarem currentemente no sistema
2. Nó de arranque providencia endereços IP de vários nós aleatórios da CAN
3. Novo nó envia um pedido JOIN para um ponto aleatório P
4. A mensagem é encaminhada para o ponto P usando o mecanismo de encaminhamento da CAN
5. Nó da zona que contém P divide a zona e assigna metade ao novo nó
   - novo nó fica com pares (K,V) respeitantes à parte recém-alocada
   - nó obtém tabela de encaminhamento do nó N que ocupava a zona
   - nó N e nós vizinhos actualizam as suas respectivas tabelas de encaminhamento

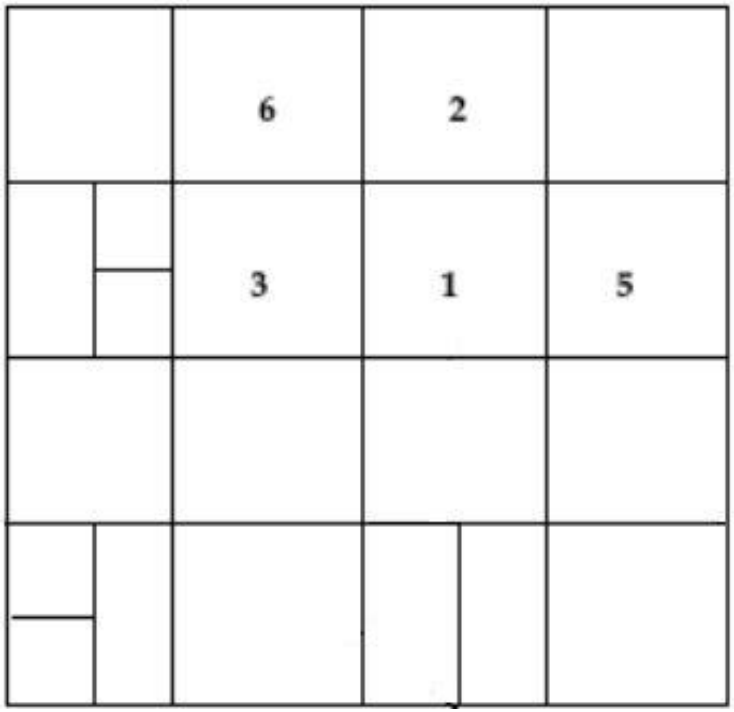
CAN: Adição de novo nó

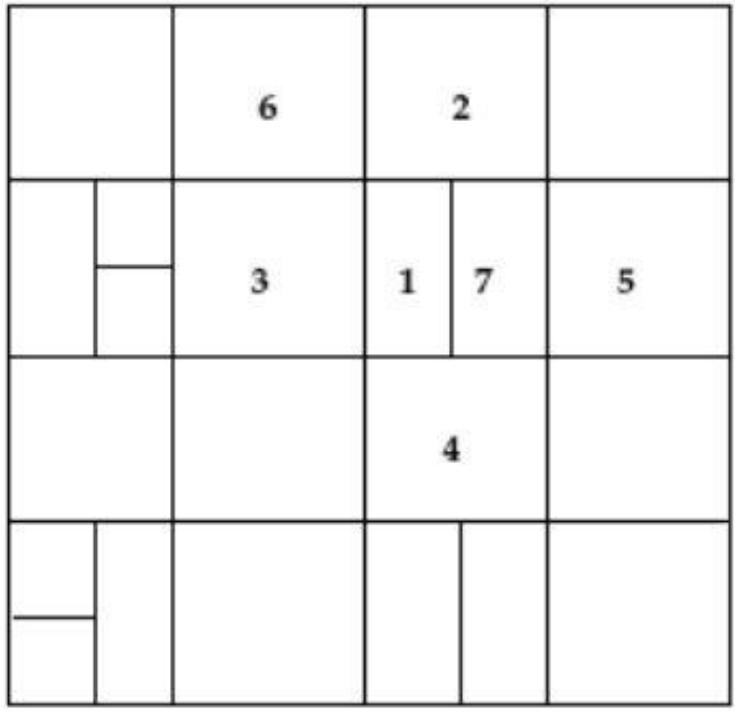
- A adição de um novo nó afecta apenas um pequeno número de nós existentes numa pequena porção do espaço de coordenadas
- O número de vizinhos que um nó mantém...
  - é dependente apenas do **número de dimensões** do espaço de coordenadas
  - é independente do **número total de nós** do sistema

CAN: Adição de novo nó

- A adição de um novo nó afecta apenas um pequeno número de nós existentes numa pequena porção do espaço de coordenadas
- O número de vizinhos que um nó mantém...
  - é dependente apenas do **número de dimensões** do espaço de coordenadas
  - é independente do **número total de nós** do sistema

CAN: Adição de novo nó





Manutenção da CAN

- Abandono de um nó: a zona é entregue a um dos nós vizinhos
  - junção das duas zonas para formar zona válida, se possível
  - se não, zona entregue temporariamente ao nó vizinho com menor zona
- Recuperação de erros: falha no acesso a um nó despoleta mecanismo de recuperação - algoritmo de TAKEOVER
  - cada nó envia mensagens periódicas a cada um dos nós vizinhos, indicando o seu estado
  - ausência prolongada na recepção da mensagem de um nó vizinho sinaliza falha nesse nó
  - os nós que detectam a falha iniciam algoritmo de TAKEOVER

Manutenção da CAN

- Abandono de um nó: a zona é entregue a um dos nós vizinhos
  - junção das duas zonas para formar zona válida, se possível
  - se não, zona entregue temporariamente ao nó vizinho com menor zona
- Recuperação de erros: falha no acesso a um nó despoleta mecanismo de recuperação - algoritmo de TAKEOVER
  - cada nó envia mensagens periódicas a cada um dos nós vizinhos, indicando o seu estado
  - ausência prolongada na recepção da mensagem de um nó vizinho sinaliza falha nesse nó
  - os nós que detectam a falha iniciam algoritmo de TAKEOVER

Manutenção da CAN

- Algoritmo de TAKEOVER:
  1. cada nó inicializa contador de **TAKEOVER** com valor proporcional ao seu tamanho (volume)
  2. após expiração do valor do contador, nó envia o valor do seu volume a todos os nós vizinhos do nó onde ocorreu a falha (mensagem de **TAKEOVER**)
  3. quando um nó recebe mensagem de **TAKEOVER**:
     - envia a sua própria mensagem de **TAKEOVER**, se o seu próprio volume é inferior ao indicado na mensagem, ou
     - cancela o seu contador de **TAKEOVER**
- Através deste mecanismo, os nós asseguram que o nó vizinho com menor volume toma conta da zona
- Caso sucedam falhas em múltiplos nós adjacentes, é efectuada uma procura alargada por nós que residam fora da região de falha, antes de iniciar o algoritmo de **TAKEOVER**

- O desenho da CAN envolve um compromisso entre:
  - estado do nó (número de nós vizinhos)
  - distância de encaminhamento
- A distância de encaminhamento é medido em termos de *hops*[2] (saltos) no caminho CAN
- Nós adjacentes na CAN podem estar geograficamente muito distantes, pelo que a latência[3] associada pode ser elevada
- Pesquisa CAN = **CAN hops** x **latência de cada CAN hop**
- O objectivo primordial das melhorias propostas é a diminuição da latência CAN - aproximá-la da latência IP subjacente
- Simulação de resultados efectuada usando o gerador de topologias GT-ITM[4]

[2] Porção do caminho entre origem e destino; ligações ponto-a-ponto na rede
[3] Montante de tempo que um pacote demora a ir de um ponto rede a outro
[4] Georgia Tech Internetwork Topology Models

Múltiplas dimensões

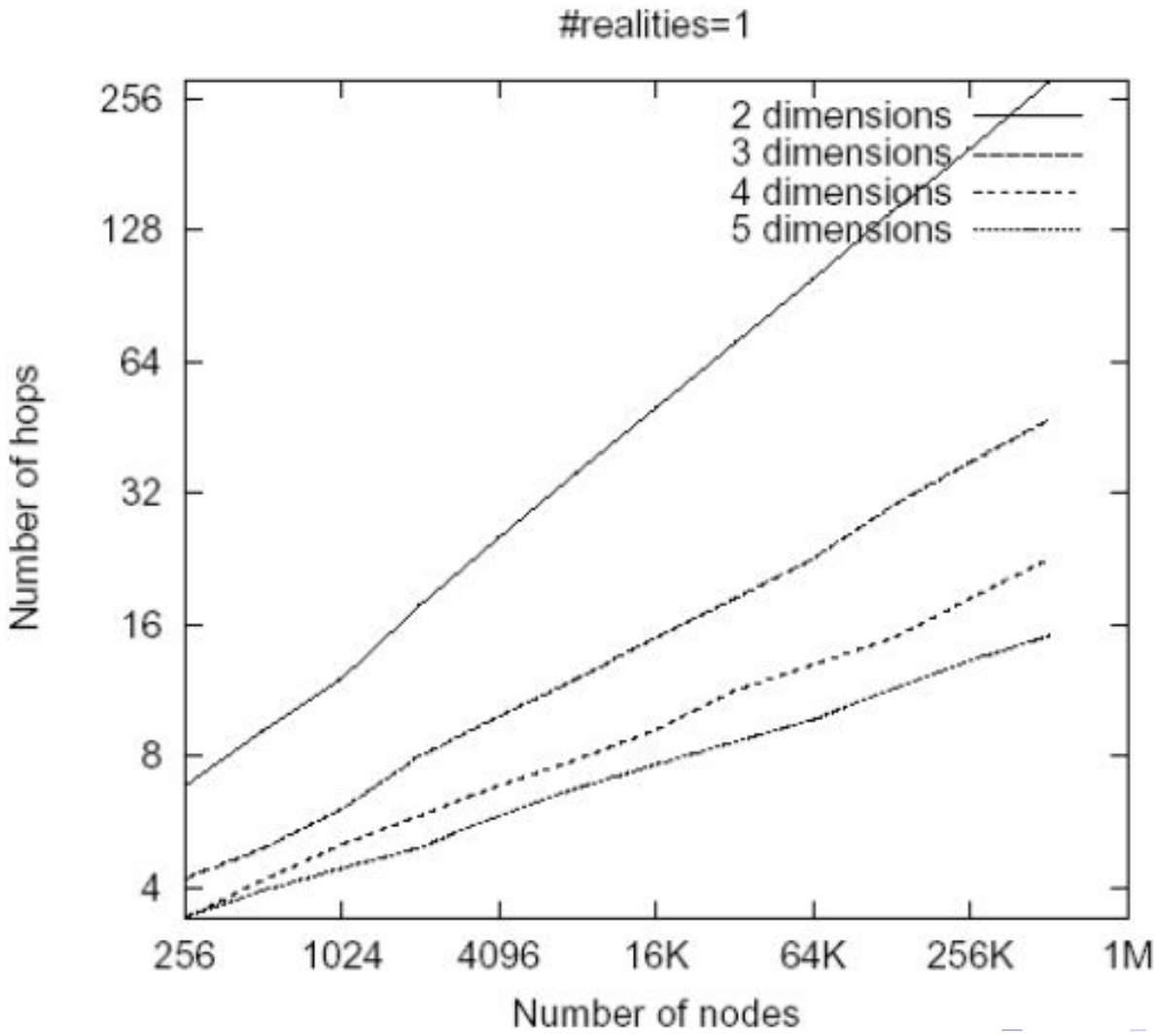
- Desenho da CAN não limita a dimensionalidade do espaço de coordenadas
- Aumentar as dimensões do espaço de coordenadas...
  - reduz distância de encaminhamento (e latência associada)
  - aumenta a tolerância a faltas no encaminhamento
    nó tem um maior nº potencial de nós aos quais a mensagem pode ser encaminhada, em caso de falha
  - origina pequeno incremento no tamanho da tabela de encaminhamento em cada nó

Múltiplas dimensões

- Desenho da CAN não limita a dimensionalidade do espaço de coordenadas
- Aumentar as dimensões do espaço de coordenadas...
  - reduz distância de encaminhamento (e latência associada)
  - aumenta a tolerância a faltas no encaminhamento

    nó tem um maior nº potencial de nós aos quais a mensagem pode ser encaminhada, em caso de falha
  - origina pequeno incremento no tamanho da tabela de encaminhamento em cada nó

Múltiplas dimensões



Múltiplas realidades

- É possível manter múltiplos e independentes espaços de coordenadas - realidades
  cada nó é assignado a uma zona diferente do espaço de coordenadas em cada realidade
- Numa CAN com **r** realidades...
  - cada nó é assignado **r** zonas de coordenadas
  - cada nó mantém **r** tabelas de encaminhamento
  - conteúdos da tabela de dispersão são replicados em cada realidade

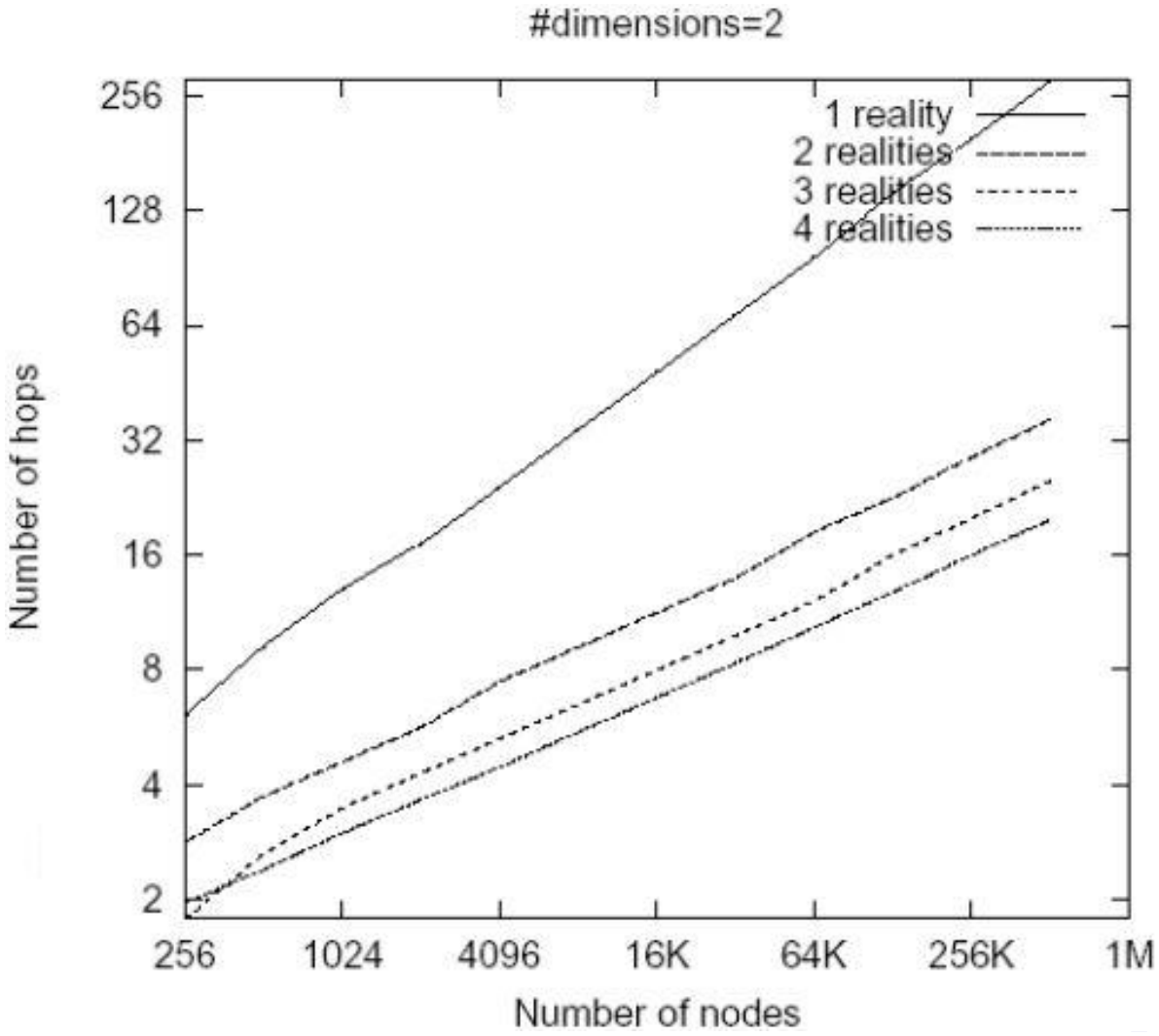
Múltiplas realidades

- É possível manter múltiplos e independentes espaços de coordenadas - realidades
  cada nó é assignado a uma zona diferente do espaço de coordenadas em cada realidade
- Numa CAN com **r** realidades...
  - cada nó é assignado **r** zonas de coordenadas
  - cada nó mantém **r** tabelas de encaminhamento
  - conteúdos da tabela de dispersão são replicados em cada realidade

Múltiplas realidades

- Aumentar o número de realidades...
  - aumenta disponibilidade dos dados
    par (K,V) apenas indisponível quando os **r** nós estão indisponíveis
  - aumenta tolerância a falhas no encaminhamento
    em caso de falha numa realidade, as restantes realidades asseguram o encaminhamento
  - reduz distância de encaminhamento (e latência associada)
    nó verifica qual o nó vizinho (no conjunto de todas as realidades), cujas coordenadas estão mais próximas da localização desejada
  - múltiplas realidades implicam múltiplas réplicas de (K,V)

Múltiplas realidades



Dimensões vs Realidades

- Aumentar o número de dimensões/realidades...
  - diminui distância de encaminhamento
  - aumenta número de nós vizinhos por nó
- Aumento do número de dimensões...
  - maior efeito em termos de distâncias de encaminhamento
- Aumento do número de realidades...
  - maior tolerância a falhas
  - maior disponibilidade dos dados

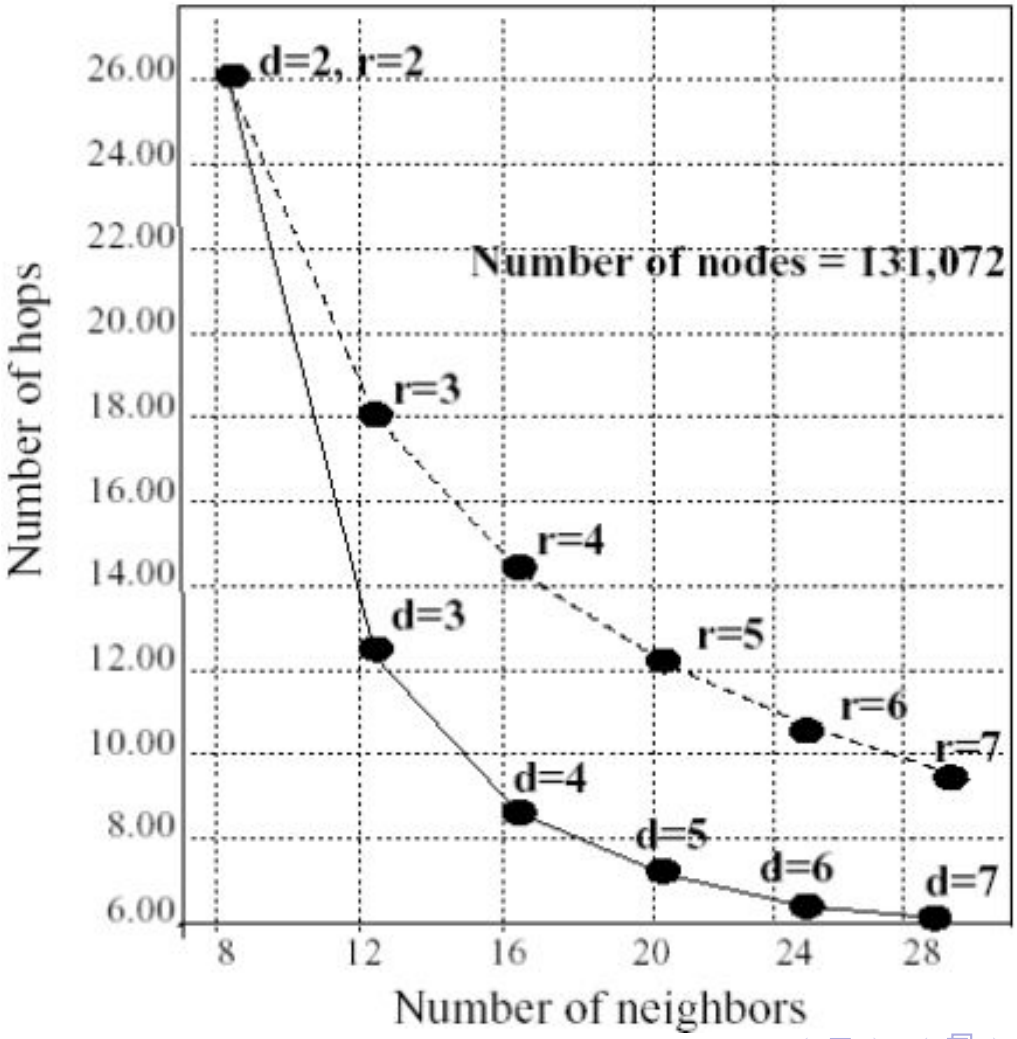
Dimensões vs Realidades

- Aumentar o número de dimensões/realidades...
  - diminui distância de encaminhamento
  - aumenta número de nós vizinhos por nó
- Aumento do número de dimensões...
  - maior efeito em termos de distâncias de encaminhamento
- Aumento do número de realidades...
  - maior tolerância a falhas
  - maior disponibilidade dos dados

Dimensões vs Realidades

- Aumentar o número de dimensões/realidades...
  - diminui distância de encaminhamento
  - aumenta número de nós vizinhos por nó
- Aumento do número de dimensões...
  - maior efeito em termos de distâncias de encaminhamento
- Aumento do número de realidades...
  - maior tolerância a falhas
  - maior disponibilidade dos dados

Dimensões vs Realidades



Métrica de encaminhamento: RTT[5]

- A métrica de encaminhamento referida anteriormente corresponde ao progresso em termos de distância cartesiana até determinada localização de destino
- Esta métrica pode ser melhorada, reflectindo a topologia IP subjacente:
  1. cada nó mede RTT relativo a cada nó vizinho
  2. mensagem encaminhada para o nó vizinho com menor RTT
- Encaminhamento feito com base no peso RTT de cada nó...
  - visa reduzir a latência de *hops* individuais

[5] Round Trip Time: montante de tempo que medeia entre o envio de um pacote e a recepção da respectiva resposta

Métrica de encaminhamento: RTT[5]

- A métrica de encaminhamento referida anteriormente corresponde ao progresso em termos de distância cartesiana até determinada localização de destino
- Esta métrica pode ser melhorada, reflectindo a topologia IP subjacente:
  1. cada nó mede RTT relativo a cada nó vizinho
  2. mensagem encaminhada para o nó vizinho com menor RTT
- Encaminhamento feito com base no peso RTT de cada nó...
  - visa reduzir a latência de *hops* individuais

[5] Round Trip Time: montante de tempo que medeia entre o envio de um pacote e a recepção da respectiva resposta

Métrica de encaminhamento: RTT[5]

- A métrica de encaminhamento referida anteriormente corresponde ao progresso em termos de distância cartesiana até determinada localização de destino
- Esta métrica pode ser melhorada, reflectindo a topologia IP subjacente:
  1. cada nó mede RTT relativo a cada nó vizinho
  2. mensagem encaminhada para o nó vizinho com menor RTT
- Encaminhamento feito com base no peso RTT de cada nó...
  - visa reduzir a latência de *hops* individuais

[5] Round Trip Time: montante de tempo que medeia entre o envio de um pacote e a recepção da respectiva resposta

Métrica de encaminhamento: RTT[6]

Number of dimensions	Non-*RTT* weighted routing (ms)	*RTT* weighted routing (ms)
2	116.8	88.3
3	116.7	76.1
4	115.8	71.2
5	115.4	70.9

[6] valores indicativos de latência *per-hop* média

Sobrecarga de zonas

- A abordagem inicial de construção da CAN indica que uma zona está assignada a um único nó, em qualquer instante
- Com esta técnica de desenho, essa assumpção é modificada:
  - Múltiplos nós podem partilhar a mesma zona - *peers*
    MAXPEERS - número máximo de *peers* permitidos por zona
  - Quando nó A junta-se ao sistema, inquire o nó B a que se destina se o valor de MAXPEERS já foi atingido:
    Se o número de *peers* na zona é inferior a MAXPEERS, adiciona-se à zona (sem dividi-la) e obtém a lista de *peers* e vizinhos de B
  - Nó tem de conhecer todos os *peers* da sua zona, mas não de zonas adjacentes
  - Periodicamente, um nó envia um pedido de lista de *peers* a todos os nós vizinhos; depois mede RTT para cada um
    guarda informação sobre quais os nós vizinhos com menor latência

Sobrecarga de zonas

- A abordagem inicial de construção da CAN indica que uma zona está assignada a um único nó, em qualquer instante
- Com esta técnica de desenho, essa assumpção é modificada:
  - Múltiplos nós podem partilhar a mesma zona - *peers*
    MAXPEERS - número máximo de *peers* permitidos por zona
  - Quando nó A junta-se ao sistema, inquire o nó B a que se destina se o valor de MAXPEERS já foi atingido:
    Se o número de *peers* na zona é inferior a MAXPEERS, adiciona-se à zona (sem dividi-la) e obtém a lista de *peers* e vizinhos de B
  - Nó tem de conhecer todos os *peers* da sua zona, mas não de zonas adjacentes
  - Periodicamente, um nó envia um pedido de lista de *peers* a todos os nós vizinhos; depois mede RTT para cada um
    guarda informação sobre quais os nós vizinhos com menor latência

Sobrecarga de zonas

- Conteúdos da tabela de dispersão podem ser:
  - replicados entre os nós de uma zona
    maior disponibilidade de dados, maior capacidade de armazenamento e de consistência de dados
  - divididos pelos nós de uma zona
    não necessita capacidade de armazenamento nem mecanismos de consistência, não melhora disponibilidade de dados

Sobrecarga de zonas

- Permitir a sobrecarga de zonas...
  - reduz distância de encaminhamento (número de *hops*) e consequentemente a latência associada
    colocar múltiplos nós por zona tem o mesmo efeito que reduzir o número de nós no sistema
  - reduz latência por *hop*
    um nó tem agora múltiplas escolhas em termos de nós vizinhos e pode seleccionar os mais próximos em termos de latência
  - melhora tolerância a falhas
    uma zona está vaga apenas quando todos os nós falham simultaneamente
  - aumento da complexidade do sistema
    nós têm de manter adicionalmente uma lista de *peers*

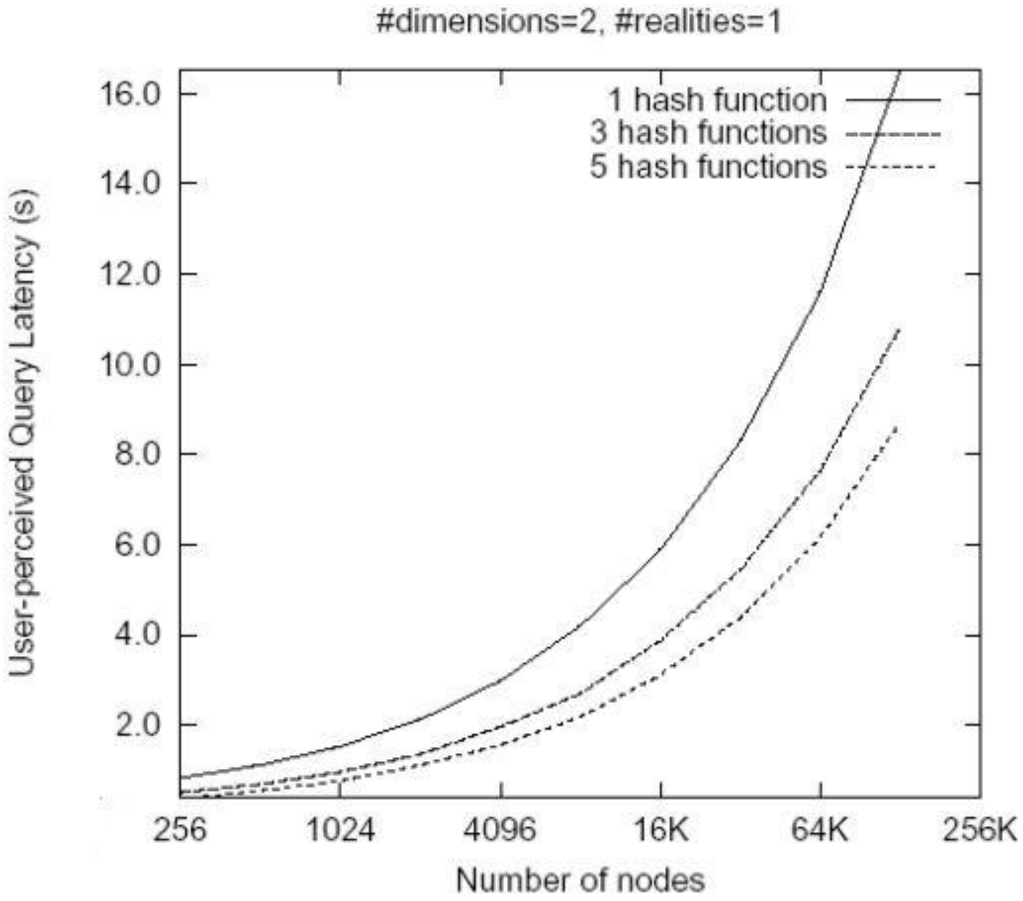
Sobrecarga de zonas

Number of nodes per zone	per-hop latency (ms)
1	116.4
2	92.8
3	72.9
4	64.4

Funções de dispersão múltiplas

- Consiste no uso de **k** funções de dispersão para mapear uma única chave em **k** pontos do espaço de coordenadas
  replicar (K,V) em **k** nós distintos no sistema
- (K,V) apenas está indisponível quando todos as **k** réplicas estão indisponíveis
- Uma entrada da tabela de dispersão pode ser obtida de duas maneiras:
  - através de *queries* em paralelo aos **k** nós
  - selecção do nó mais perto no sistema de coordenadas
- O uso de múltiplas tabelas de dispersão...
  - aumenta a disponibilidade dos dados
  - aumento do tamanho da bd de (K,V) e tráfego relativo a *queries* (quando existente)

Funções de dispersão múltiplas



Sensibilidade a topologia geográfica

- O mecanismo de construção da CAN referido inicialmente aloca nós a zonas de forma aleatória
  os nós vizinhos em termos de CAN não são necessariamente vizinhos em termos da rede IP subjacente
- Os mecanismos referidos anteriormente apenas melhoram a selecção de encaminhamentos, não a estrutura da rede em si
- Esta tentativa de melhoramento consiste em aproximar a topologia CAN à topologia IP subjacente:
  - É assumida a existência de um conjunto de máquinas que actuam como marcos (*landmarks*) na Internet
    por exemplo, servidores DNS
  - cada nó mede o seu RTT para cada um destes *landmarks*
    - os *landmarks* são ordenados em termos de RTT ascendente
    - para m *landmarks*, existem m! ordenamentos
    - o espaço de coordenadas é dividido em m! porções de igual tamanho

Sensibilidade a topologia geográfica

- O mecanismo de construção da CAN referido inicialmente aloca nós a zonas de forma aleatória
  os nós vizinhos em termos de CAN não são necessariamente vizinhos em termos da rede IP subjacente
- Os mecanismos referidos anteriormente apenas melhoram a selecção de encaminhamentos, não a estrutura da rede em si
- Esta tentativa de melhoramento consiste em aproximar a topologia CAN à topologia IP subjacente:
  - É assumida a existência de um conjunto de máquinas que actuam como marcos (*landmarks*) na Internet
    por exemplo, servidores DNS
  - cada nó mede o seu RTT para cada um destes *landmarks*
    - os *landmarks* são ordenados em termos de RTT ascendente
    - para m *landmarks*, existem m! ordenamentos
    - o espaço de coordenadas é dividido em m! porções de igual tamanho

Sensibilidade a topologia geográfica

- O mecanismo de construção da CAN referido inicialmente aloca nós a zonas de forma aleatória
  os nós vizinhos em termos de CAN não são necessariamente vizinhos em termos da rede IP subjacente
- Os mecanismos referidos anteriormente apenas melhoram a selecção de encaminhamentos, não a estrutura da rede em si
- Esta tentativa de melhoramento consiste em aproximar a topologia CAN à topologia IP subjacente:
  - É assumida a existência de um conjunto de máquinas que actuam como marcos (*landmarks*) na Internet
    por exemplo, servidores DNS
  - cada nó mede o seu RTT para cada um destes *landmarks*
    - os *landmarks* são ordenados em termos de RTT ascendente
    - para m *landmarks*, existem m! ordenamentos
    - o espaço de coordenadas é dividido em m! porções de igual tamanho

Sensibilidade a topologia geográfica

- Nó junta-se à CAN num ponto aleatório correspondente ao seu ordenamento por *landmark*
- Nós topologicamente próximos tendem a ter o mesmo ordenamento e consequentemente, a residir na mesma porção do espaço de coordenadas
  nós vizinhos na CAN tendem a ser próximos topologicamente na Internet
- Este esquema de ordenamento resulta:
  - numa melhoria da latência associada à distância de encaminhamento
  - num espaço de coordenadas não uniformemente populado

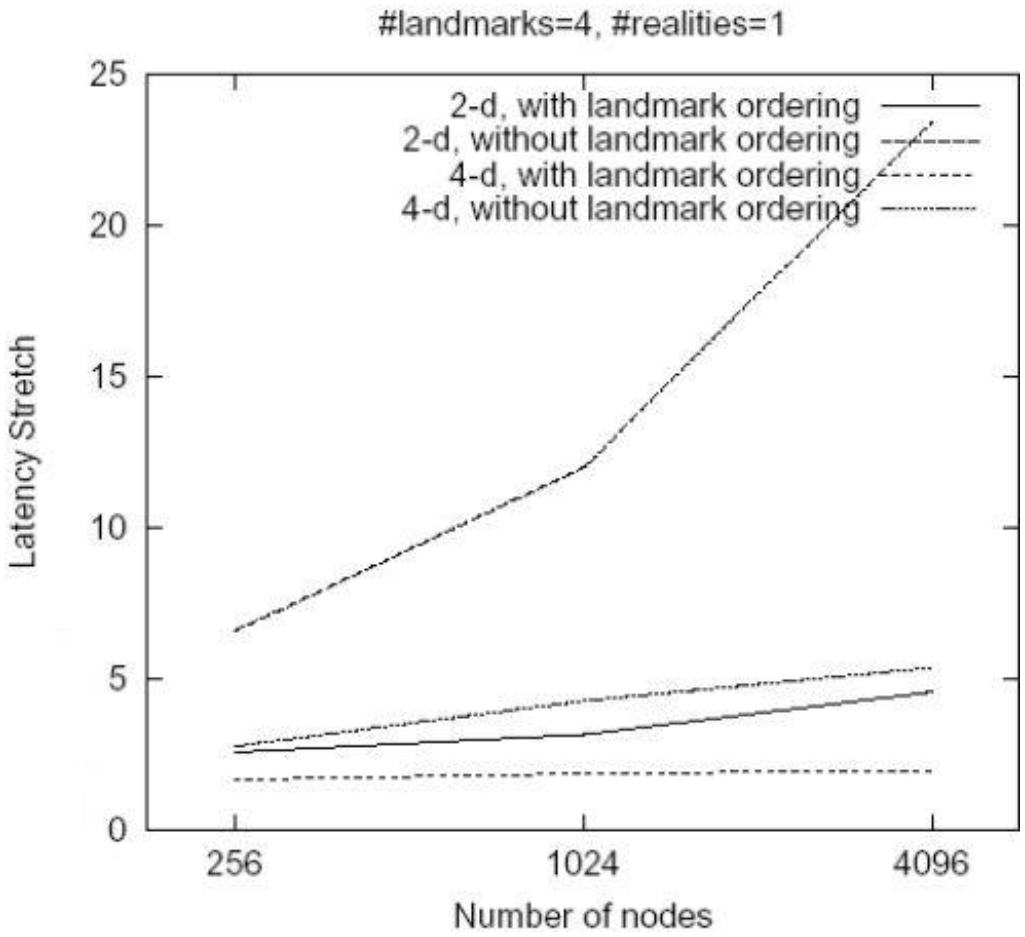
Sensibilidade a topologia geográfica

- Nó junta-se à CAN num ponto aleatório correspondente ao seu ordenamento por *landmark*
- Nós topologicamente próximos tendem a ter o mesmo ordenamento e consequentemente, a residir na mesma porção do espaço de coordenadas
  nós vizinhos na CAN tendem a ser próximos topologicamente na Internet
- Este esquema de ordenamento resulta:
  - numa melhoria da latência associada à distância de encaminhamento
  - num espaço de coordenadas não uniformemente populado

Sensibilidade a topologia geográfica

- Nó junta-se à CAN num ponto aleatório correspondente ao seu ordenamento por *landmark*
- Nós topologicamente próximos tendem a ter o mesmo ordenamento e consequentemente, a residir na mesma porção do espaço de coordenadas
  nós vizinhos na CAN tendem a ser próximos topologicamente na Internet
- Este esquema de ordenamento resulta:
  - numa melhoria da latência associada à distância de encaminhamento
  - num espaço de coordenadas não uniformemente populado

Sensibilidade a topologia geográfica[7]



[7] Latency stretch $= \frac{\textit{latência CAN média}}{\textit{latência IP média}}$

Outras melhorias propostas

- Particionamento uniforme
  - adição de nó efectua-se com base na noção de volume de zona
  - quando um nó existente recebe pedido de junção de novo nó, compara o seu volume de zona com o dos seus nós vizinhos
  - a zona com maior volume é escolhida para acomodar o novo nó
  - origina divisão uniforme da carga (K,V) de cada nó
  - não gera verdadeiro balanceamento de carga, porque alguns pares (K,V) são mais acedidos que outros
- *Caching*
  - Manutenção de (K,V) recentemente acedidos em *cache*
  - Antes de encaminhar determinado pedido para o seu destino, o nó verifica se a chave correspondente existe em *cache*
  - Em caso afirmativo, satisfaz ele próprio o pedido, não o encaminhando
  - O nº de *caches* onde existe (K,V) é proporcional à popularidade da chave
- Replicação
  - Criação de réplicas de (K,V) frequentemente acedidos, em nós vizinhos
  - Nó decide replicar quando existe uma sobrecarga de pedidos por (K,V)
  - Permite que a carga seja dividida por uma região de coordenadas

Outras melhorias propostas

- Particionamento uniforme
  - adição de nó efectua-se com base na noção de volume de zona
  - quando um nó existente recebe pedido de junção de novo nó, compara o seu volume de zona com o dos seus nós vizinhos
  - a zona com maior volume é escolhida para acomodar o novo nó
  - origina divisão uniforme da carga (K,V) de cada nó
  - não gera verdadeiro balanceamento de carga, porque alguns pares (K,V) são mais acedidos que outros
- *Caching*
  - Manutenção de (K,V) recentemente acedidos em *cache*
  - Antes de encaminhar determinado pedido para o seu destino, o nó verifica se a chave correspondente existe em *cache*
  - Em caso afirmativo, satisfaz ele próprio o pedido, não o encaminhando
  - O nº de *caches* onde existe (K,V) é proporcional à popularidade da chave
- Replicação
  - Criação de réplicas de (K,V) frequentemente acedidos, em nós vizinhos
  - Nó decide replicar quando existe uma sobrecarga de pedidos por (K,V)
  - Permite que a carga seja dividida por uma região de coordenadas

Outras melhorias propostas

- Particionamento uniforme
  - adição de nó efectua-se com base na noção de volume de zona
  - quando um nó existente recebe pedido de junção de novo nó, compara o seu volume de zona com o dos seus nós vizinhos
  - a zona com maior volume é escolhida para acomodar o novo nó
  - origina divisão uniforme da carga (K,V) de cada nó
  - não gera verdadeiro balanceamento de carga, porque alguns pares (K,V) são mais acedidos que outros
- *Caching*
  - Manutenção de (K,V) recentemente acedidos em *cache*
  - Antes de encaminhar determinado pedido para o seu destino, o nó verifica se a chave correspondente existe em *cache*
  - Em caso afirmativo, satisfaz ele próprio o pedido, não o encaminhando
  - O nº de *caches* onde existe (K,V) é proporcional à popularidade da chave
- Replicação
  - Criação de réplicas de (K,V) frequentemente acedidos, em nós vizinhos
  - Nó decide replicar quando existe uma sobrecarga de pedidos por (K,V)
  - Permite que a carga seja dividida por uma região de coordenadas

Conclusões

- O desenho de CAN proposto é:
  - completamente **distribuído**
    não existe qualquer forma de controlo centralizado
  - **escalável**
    nós mantêm um diminuto controlo de estado, independente do número de nós no sistema
  - **tolerante a falhas**
    encaminhamento efectuado à volta das zonas de falha
- O desenho proposto não impõe nenhuma forma de estrutura hierárquica de nomes para atingir escalabilidade
- As melhorias propostas resultam em melhorias em termos de:
  - latência/distância de encaminhamento
  - disponibilidade dos dados
- Resultados obtidos por simulação satisfatórios
  para uma CAN com 260.000 nós, a latência de encaminhamento foi menos do dobro da latência IP subjacente

- **A Scalable Content-Adressable Network**
  S.Ratnasamy, P.Francis, M.Handley, R.Karp, S.Shenker
  http://www.di.fc.ul.pt/~ler/docencia/tm/papers/p13-ratnasamy.pdf
- **A Scalable Content Addressable Network**
  S.Ratnasamy, P.Francis, M.Handley, R.Karp
  www.cs.ucsb.edu/~sep/cs271-pres/can.ppt
- **Modeling Topology of Large Internetworks**
  http://www.isi.edu/nsnam/ns/ns-topogen.html#gt-itm