"A Routing Scheme for Content-Based Networking"

Apresentado por:

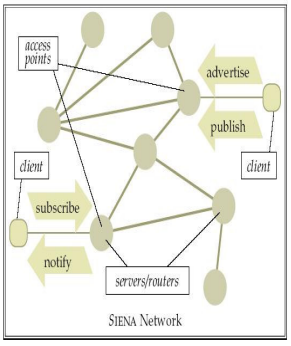
Amadeu Dias
amadeu@di.fc.ul.pt

Estrutura

- Siena
- Redes Baseadas em Conteúdo
- CBCB
  - Descrição
  - Encaminhamento
  - Protocolos
  - Optimizações
- Análise

Siena

- Scalable Internet Event Notification Architecures
- Objectivo:
  - Construir um serviço de notificação publish/subscriber escalável



Revisão

- Context Based Publish-Subscribe
  - Publishers / Senders
    - Publicam conteúdo
  - Subscriber / Receivers
    - Subscrevem conteúdos
    - Baseado em propriedades da mensagem

Redes Baseadas em Conteúdo

- Propriedades
  - Não existe endereçamento directo
  - Cada subscriber indica as propriedades que as mensagens que pretendem receber exibem
- Objectivo
  - Entregar as mensagens que exibem essas propriedades a todos os subscribers que as desejam receber

Redes Baseadas em Conteúdo

- Elementos
  - Mensagem
    - Atributos
      - Nome
      - Tipo { integer, double, boolean,string}
      - Valor
    - Exemplo: [accao='PT',preco=100,variacao=false]
  - Predicados
    - Disjunção de conjunções ("ou"'s de "e"'s) de condições lógicas
    - Exemplo (accao ='PT' AND preco<100)
  - Operações
    - Send_message(m)
    - Set_predicate(p)

O que é uma CBCB?

- Combined Broadcast and Content Based
- 2 camadas
  - Broadcast
    - Estabelece uma árvore de broadcast
  - Content-Based
    - Poda a árvore de broadcast
- Podemos pensar numa CBCB como uma rede em que árvore de broadcast é alterada dinamicamente

CBCB

- Broadcast
  - Concretiza uma maneira de enviar uma mensagem desde o publisher até todos os potenciais subscribers de uma rede
- Content-based
  - Difunde as condições de subscrição dos subscribers para os publishers
  - Faz o encaminhamento das mensagens dos publishers para os subscribers

CBCB

- Estratégia geral
  - “Espalhar” as condições anunciadas desde os subscribers até aos publishers

CBCB

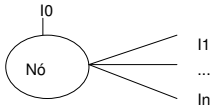
- Broadcast
  - Princípios
    - Uma função de broadcast que para cada nó e dada a origem da mensagem, retorna quais os interfaces para os quais enviar a mensagem
    - B(s,i)
      - S → nó fonte da mensagem
      - i → interface input
      - Resultado → conjunto de interfaces para os quais é possível enviar mensagens ao longo da árvore
  - Propriedades
    - Definir uma àrvore de broadcast
    - All-pairs path simmetry

CBCB - Encaminhamento

- Conceitos
  - Endereços
    - O endereço de um nó é o conjunto de condições que define as mensagens que cada nó está apto a receber (i.e. o predicado)
  - Relação de cobertura
    - P(m) → avaliação do predicado p sobre uma mensagem m
    - Selection(p) → conjunto de mensagens que p seleciona
      - Diz-se que p1 cobre p2 se selection(p2) é um subconjunto de selection(p1)

CBCB - Encaminhamento

- Componentes
  - Tabela de encaminhamento BC



Interface	Endereço
I0	P0
I1	P1
..	...
In	Pn

CBCB - Encaminhamento

- Função de encaminhamento (FC)
  - Indica quais os interfaces a usar para enviar as mensagens face ao conteúdo das mesmas

(B(source(m) , incoming_if(m)) U {I0})

∩

FC(m)

CBCB - Protocolos

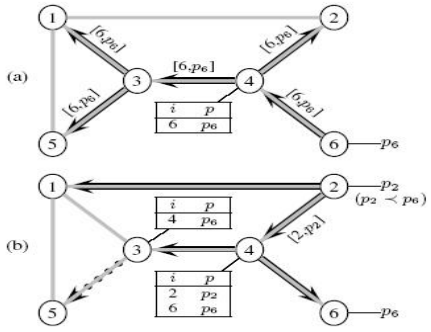
- Protocolos de encaminhamento
  - RA (Receiver Advertisements)
    - Mensagens dos nós subscriber
    - Difundem as condições das mensagens que estão interessados em receber para os potenciais publishers
    - Servem para estabelecer informação de encaminhamento ao longo dos nós da rede
    - São enviados periodicamente e/ou quando a condição de subscrição é alterada

issuer
predicate
...

CBCB – Protocolos - RA

- Protocolo
  - Se o Pi cobrir Pra → RA é descartado
  - Se o Pi não cobrir o Pra:
    - O RA é difundido para os restantes nós na árvore de broadcast
    - Na tabela de encaminhamento é criado um novo predicado para aquele interface
      - Pi = Pi U Pra

CBCB – Protocolos - RA

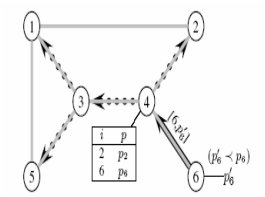


CBCB – Protocolos - RA

- Consequências do protocolo
  - Inflação dos endereços/predicados:
  - As condições mais específicas são sempre cobertas →
  - Os RA's dessas condições não são difundidos →
  - Os nós passam a receber mensagens não desejadas

CBCB – Protocolos - RA

- Exemplo:
  - p6=preco<100
  - p6'=preco<50
  - Nó 5 quer enviar mensagem com preco=70
  - Devido à regra a ùltima condição não é propagada para os restantes nós
  - Nó 5 não tem essa alteração na sua tabela e envia a mensagem na mesma pela arvore de broadcast
  - Nó 6 recebe uma mensagem indesejada

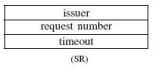


CBCB – Protocolos - RA

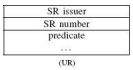
- Solução:
  - Protocolo SR
    - Estratégia:
      - "Estancar" a inflação de endereços
      - Passa por ir buscar juntos dos subscribers as condições mais específicas e recentes
      - Essas condições são armazenadas na tabela do publisher para serem utilizadas "à saída" da mensagem da origem
  - Como?

CBCB – Protocolos – SR/UR

- SR (Sender Request)
  - São criados pelos publishers
  - São usados para estabelecer as tabelas de encaminhamento do publisher ao enviar



- UR (Update Reply)
  - São respostas a SR que são enviados

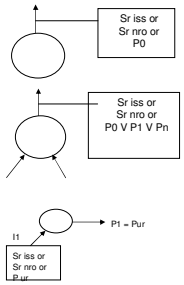


CBCB – Protocolos – SR/UR

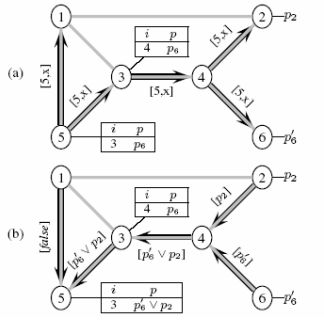
- Objectivos
  - Limitar a inflação de endereços → Limitar o número de mensagens enviadas erroneamente
  - Compensar possíveis perdas de difusão de RA's
- Protocolo
  - Difusão de SR
    - Feita periodicamente
    - Todos os nós (excepto os terminais) enviam os SR's para baixo na árvore de broadcast mantendo a identificação de quem enviou

CBCB – Protocolos – SR/UR

- Processamento de SR/UR
  - Se fôr um nó terminal:
    - Envia um UR com o seu endereço local "para cima"
  - Se for um nó intermedio:
    - Envia um UR cujo predicado é a combinação do seu predicado local com os que estão nos UR recebidos
  - Se for o nó que enviou o SR correspondente:
    - Actualiza a tabela de routing para o interface no qual recebeu a mensagem com Pur



CBCB – Protocolos – SR/UR



CBCB – Protocolos – SR/UR

- Timeouts
  - Quando um SR é recebido é logo propagado para baixo
  - Um UR é enviado para cima:
    - Nós intermédios
      - Após receber todos os UR's de baixo
      - Após timeout td= t – tc – tp
    - Nós terminais: Imediatamente

CBCB - Optimizações

- Problemas
  - Grande quantidade de mensagens de controlo:
    - O protocolo SR/UR foi desenhado para limitar o número de mensagens enviadas, mas...
    - Também gera um número consideravel de mensagens
  - O que para grandes redes pode afectar a escalabilidade
- Como contornar???

CBCB - Optimizações

- Caching e reutilização de UR's
  - Estratégias:
    - Não propagação de SR's para baixo na árvore de broadcast
    - Actualização imediata das tabelas de um nó intermédio
  - Em princípio, apenas o emissor do SR pode actualizar as suas tabelas com a informação do UR
  - Motivo: cada árvore de broadcast é específica para cada emissor de SR

CBCB - Optimizações

- Caching e reutilização de UR's (cont)
  - Regra:
    - Um nó só pode re-usar e fazer cache de um UR se:
      - Um dos interfaces (links) que sai desse nó ligar duas partes da rede que estariam desconexas, isto é, esse link é uma bridge
  - Este caching de UR's diminui a propagação de SR's diminuindo o tráfego na rede

CBCB - Optimizações

- Controlo dos SR's
  - Em vez de enviar SR's periodicamente (gerando tráfego "desnecessário") controlar o envio de SR's
  - Sempre que as mensagens enviadas através de um link/interface excedem um determinado número (configurável) é lançado um novo SR para restabelecer as novas condições na tabela de encaminhamento

CBCB - Optimizações

- Simplificação de endereços
  - Consiste em simplificar a lógica dos endereços
  - Convém (!) manter a mesma semântica
  - Explo:
    - Nó anuncia: (preco>50 AND preco<200)
    - Novo predicado obriga a actualização: (preco <100)
    - Nova condição:
      (preco>50 AND preco<200) OR (preco<100)
    - Sem alteração da semântica podemos simplificar para:
      preco<200

CBCB - Optimizações

- Simplificar demais?
  - Exemplo:
    (preco>50 AND preco < 200) OR (preco=250) OR (preco > 500)
    para
    (preco>50)
  - Consequências:
    - Vão ser entregues mensagens não desejadas!

CBCB - Optimizações

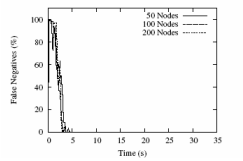
- Simplificar demais?
  - Positivo:
    - Não é necessário tanto tempo para simplificar uma expressão lógica
    - As tabelas de encaminhamento são mais pequenas
  - Negativo
    - Leva a que haja um maior tráfego de mensagens
    - Entrega de mensagens não desejadas

Avaliação

- Segundo 3 vectores
  - Funcionalidade: Se entrega as mensagens a quem está interessado
  - Filtragem: Se previne a entrega de mensagens não desejadas
  - Escalabilidade: Se o tráfego gerado na rede face ao número de nós é aceitável
- Rede Usada
  - Simulação:
    - Todos os nós são routers
    - 75% são receivers e 20% são senders
    - As aplicações só estão interessadas numa fracção das mensagens

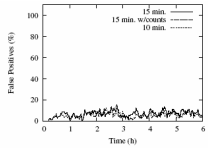
Avaliação

- Funcionalidade
  - Existe um período de 5 segundos em que é feito o set-up do routing
  - Durante esse período existem nós que não recebem as mensagens a eles destinados

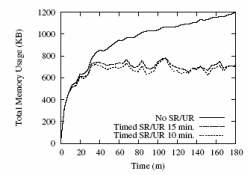


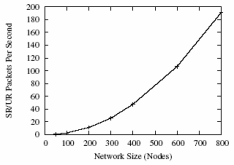
Avaliação

- Filtragem
  - Em média 10% do tráfego de rede são falsos positivos
    - A variante de SR's que apresenta melhor performance é a que envia um SR cada 10 minutos



Escalabilidade





Referências

- "A Routing Scheme for Context-Based Networking", Antonio Carzaniga, Matthew Rutherford, Alexander Wolf
- http://serl.cs.colorado.edu/~carzanig/siena/siena_flyer.pdf