Uma análise comparativa entre protocolos de comunicação autonômicos baseados em *gossip*

Eduardo C. B. Julião[1], **Émerson R. Silva**[1], **Patricia T. Endo**[1]

[1]Faculdade de Ciências e Tecnologia de Caruaru - Universidade de Pernambuco (UPE)
Rodovia BR 104, Km 62 - Polo Caruaru - Caruaru - PE – Brasil

{eduardocesarbj, emersonremigio, patriciaendo}@gmail.com

Resumo. *Os protocolos autonômicos e distribuídos baseados em gossip são bastante utilizados em cenários onde há necessidade de uma rapidez na comunicação e de uma fácil escalabilidade. O presente resumo tem como principal objetivo apresentar o comportamento desses protocolos, possibilitando uma análise comparativa acerca dos mesmos. São apresentados alguns exemplos de utilização desses protocolos, discutindo suas correspondentes otimizações.*

Abstract. *Autonomic and distributed protocols based on gossip are widely used in scenarios where is a fast communication and an easy scalability are needed. This article has as main objective presenting these protocols behavior, enabling a comparative analysis between them. We present some usage examples of such protocols, discussing their corresponding optimizations.*

1. Introdução

Desde o advento da computação, os sistemas de computadores estão em constante evolução. Dentre essas evoluções está o avanço do paradigma de sistemas centralizados para os sistemas distribuídos. Os sistemas centralizados funcionam, basicamente, de modo que um servidor concentra todas as informações que serão utilizadas ou requisitadas por seus clientes. Porém, esses sistemas possuem alguns pontos fracos bastante conhecidos, tais quais: único ponto de falha ou gargalo, poder computacional limitado e centralizado. Por outro lado, os sistemas distribuídos, definidos por Coulouris, Dollimore e Kindberg (2007, p. 15) como "*aqueles nos quais os componentes localizados em computadores interligados em rede se comunicam e coordenam suas ações apenas passando mensagens*", minimizam os pontos fracos dos sistemas centralizados citados anteriormente. Agora, pelo fato dos computadores estarem distribuídos fisicamente, mas conectados uns aos outros através de uma rede e poderem trocar informação entre si, o sistema como um todo pode viabilizar uma maior capacidade de processamento, maior robustez e uma maior escalabilidade.

Deste modo, a comunicação realizada entre os computadores de um sistema distribuído se dá por trocas de mensagens, que são padronizadas por protocolos de comunicação. Os protocolos abordados neste trabalho realizam a disseminação das mensagens baseadas em *gossip* (fofoca). Esses tipos de protocolos estão sendo cada vez mais utilizados nos sistemas distribuídos por apresentarem muitas vantagens, como a simplicidade, escalabilidade e a alta velocidade de difusão de informação. Porém,

alguns pontos fracos também podem ser identificados nos mesmos, como por exemplo, a transmissão de mensagens redundantes e a dificuldade de identificar uma condição de parada da propagação de uma informação após o seu início.

Este resumo tem como principal objetivo realizar um levantamento do estado da arte sobre os protocolos de comunicação baseados em *gossip* e apresentar uma análise comparativa entre os mesmos, destacando os pontos fortes e fracos de cada tipo de implementação. Para tanto, o resumo está estruturado da seguinte forma: a Seção 2 apresenta os protocolos baseados em *gossip* e uma classificação para os mesmos; a Seção 3 descreve e discute sobre alguns protocolos existentes; e por fim a Seção 4 apresenta as conclusões obtidas e os trabalhos futuros.

2. Protocolos baseados em *gossip*

Os protocolos baseados em *gossip* têm esse nome particular por realmente parecerem com uma fofoca, pois funcionam de modo que uma informação, no caso a fofoca, se propaga rapidamente em pouco tempo. Por isso, os nós que possuem uma informação fazem a disseminação da mesma entre um número aleatório de vizinhos visando atingir a maior quantidade de nós possível.

Demers et al. (1987) foram os pioneiros ao trabalharem com protocolo baseado em *gossip*, usando-o para manutenção de banco de dados replicados. Isto serviu como ponta pé inicial para que seu uso fosse amplamente estendido para diversas soluções em sistemas distribuídos, tais como comunicação, gerenciamento de recursos até a detecção de falhas.

2.1. *Gossip* genérico

O esquema genérico dos protocolos baseados em *gossip* é baseado em duas *threads*, sendo uma *thread* ativa e a outra passiva, como mostra a Figura **1**. Cada nó possui as duas *threads* implementadas, sendo a ativa utilizada no momento em que um determinado nó deseja iniciar uma comunicação, ou seja, basicamente um nó que deseja transmitir uma mensagem escolhe alguns nós vizinhos randomicamente para então difundir essa mensagem; e a passiva executada sempre em um nó, pois é a responsável por receber e aceitar requisições provenientes dos vizinhos.

***Thread* ativa**

```
1: loop
2:     espera(tempo)
3:     p = selecionaNos( )
4:     s = preparaMensagem( )
5:     envia (REQUISICAO, s) para p
6: fim do loop
```

***Thread* passiva**

```
1: loop
2:     receba (t, sp) de todos os nós
3:     se t = REQUISICAO então
4:         s = preparaMensagem( )
5:         envia (RESPOSTA, s) para t
6:     fim do se
7:     atualiza(sp)
8: fim do loop
```

Figura 1. O esquema genérico do *gossip* (Traduzido de MONTRESOR, 2008)

A *thread* ativa seleciona primeiramente, a cada determinado espaço de tempo e através do método *selecionaNos()*, uma quantidade aleatória de nós vizinhos que é atribuída à variável *p*. Posteriormente, é com o método *preparaMensagem()* que torna-

se possível armazenar informações sobre estado local do nó na variável *s*, para que então essa seja enviada, através do método *envia(REQUISIÇÃO,s)*, para os nós selecionados.

Já a thread passiva espera mensagens vindas dos seus nós vizinhos por tempo indefinido. Também faz uso do método *preparaMensagem()*, caso receba uma requisição de um vizinho, fazendo com que seja enviado ao remetente as informações sobre seu estado local. Por fim, o método *atualiza(s_p)* é responsável por realizar a atualização do estado local do nó pelo que foi recebido por seu vizinho.

Montressor (2008) define o esquema mostrado na Figura 1 de forma genérica. Devido a isso, ele lista uma série de características que diferenciam os protocolos baseados em *gossip* dos que não são:

- A seleção dos nós vizinhos deve ser randômica;
- Apenas informações locais estão disponíveis em todos os nós;
- A comunicação é periódica;
- A transmissão e a capacidade de processamento são limitadas;
- Todos os nós executam o mesmo protocolo.

2.2. Variações do protocolo *gossip* genérico

Existem dois parâmetros fundamentais na configuração dos protocolos *gossip* segundo Leitão (2007), que podem influenciar no desempenho dos mesmos: *fanout* e *maximum rounds.* Ambos estão representados na Figura 2, onde *fanout* representa o número de nós selecionados pelo protocolo para cada mensagem recebida pela primeira vez; e o segundo parâmetro, o *maximum rounds*, descreve o número máximo de vezes que uma determinada mensagem será retransmitida pelos nós.

Dessa forma, o autor apresenta dois modos nos quais os protocolos baseados em *gossip* podem atuar, sendo o Modo Ilimitado, onde não é estabelecido o parâmetro *maximum rounds*, e o Modo Limitado, o qual determina que o valor do parâmetro *maximum rounds* seja maior que 0, limitando então a retransmissão de cada mensagem.

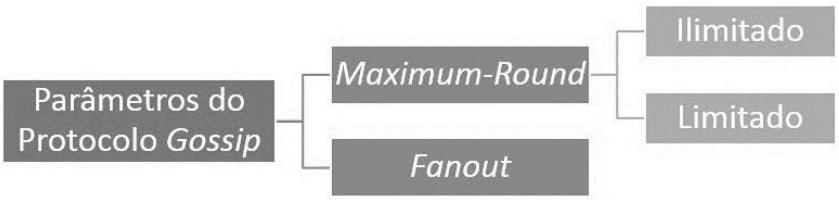

Figura 2. Parâmetros de configuração dos protocolos *gossip* (baseado em Leitão, 2007)

Através da Figura 3, torna-se possível a visualização do comportamento do protocolo baseado em *gossip*, onde no primeiro momento o círculo cinza escuro, que representa a fonte da mensagem que deve ser transmitida para todos os outros círculos do sistema, inicia a difusão da mensagem para um número aleatório de vizinhos. Após isso, cada um dos círculos que recebeu a mensagem repete o procedimento, atingindo uma outra quantidade de vizinhos aleatórios que darão continuidade ao processo. Com isso, é bem provável que, após alguns rounds, os círculos que ainda não tinham recebido

a mensagem recebam. A propagação da mensagem, então, torna-se extremamente rápida por difundila com crescimento exponencial entre os participantes de um sistema.

2.3. Classificação

Os algoritmos baseados em *gossip* podem ser divididos, segundo Renesse et al. (2008), em duas categorias, sendo elas: *Anti-Entropy* e *Rumor Mongering*. A primeira categoria propaga a informação desejada até o momento em que ela seja desatualizada por uma nova informação, já a segunda espalha a informação por um período de tempo definido previamente, como descrevem, sucintamente, os autores. A Figura 4 tem a finalidade de resumir o comportamento desses protocolos, bem como os relacionar com um dos parâmetros vistos na Seção anterior. Onde esse parâmetro, o *maximum-round*, é responsável por limitar ou não o número de vezes que uma mensagem será transmitida.

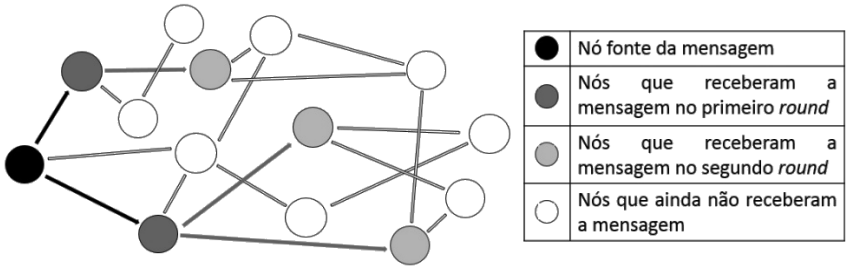

Figura 3. O comportamento do algoritmo baseado em *gossip*

2.3.1. Anti-Entropy

Revière e Voulgaris (2011, p. 257) detalham o comportamento desse algoritmo: "*No AntiEntropy, cada nó, fofoca periodicamente, ou seja, periodicamente ele escolhe randomicamente um nó entre todos os outros*". Nesse sentindo, pode-se concluir que essa categoria age semelhante ao algoritmo genérico, pois, ele somente preocupa-se na propagação da mensagem, diferente do que acontece no *Rumor Mongering*, que será visto no tópico à seguir. A categoria atua, basicamente, de forma que a cada espaço de tempo os nós que possuem uma mensagem escolhem de maneira aleatória algum vizinho para enviá-la.

2.3.2. Rumor Mongering

Esse protocolo é caracterizado por atribuir um determinado número à mensagem que se pretende transmitir, número que, por sua vez, pode ser chamado de tempo de vida da mensagem, é responsável por representar a permanência da mensagem na rede e é decrescido a cada vez em que a mensagem é transmitida.

Então, ele funciona de modo que um nó inicia a transmissão de uma mensagem escolhendo um número aleatório de vizinhos, que por sua vez, dão continuidade ao processo de propagação da mensagem. Essa propagação só é realizada uma única vez para cada mensagem recebida até que o tempo de vida da mensagem seja igual a zero, ou seja, quando um nó recebe uma mensagem, ele escolhe alguns de seus vizinhos randomicamente para enviar a mensagem, e, então, para o processo de propagação até que uma nova mensagem seja recebida.

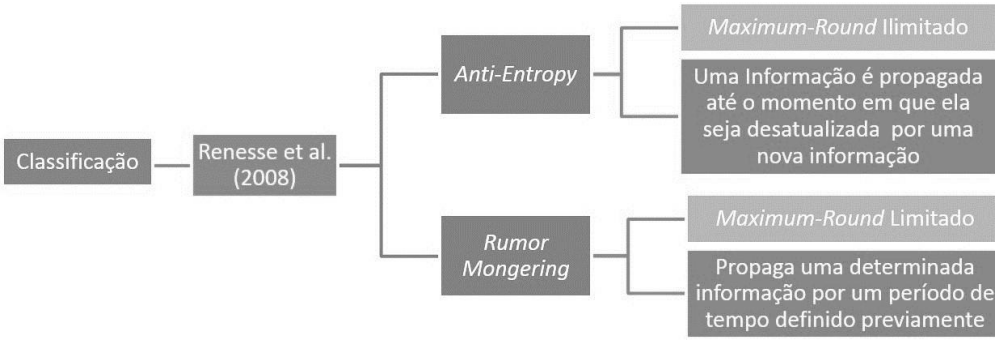

Figura 4. Classificação do *Gossip*

3. Implementações existentes

Atualmente a utilização dos protocolos baseados em *gossip* abrangem uma extensa área da computação, sendo utilizados tanto para detecção de falhas quanto para manutenção de banco de dados replicados. Nas sub-seções a seguir serão abordados alguns protocolos e suas respectivas utilidades.

3.1. NeEM

O Network Friendly Epidemic Multicast é um exemplo de protocolo baseado em *gossip* e fora apresentado por Pereira et al. (2003). O protocolo proposto tem como característica crucial ser *network-friendly*, o que faz com que a comunicação entre os nós dependa de um protocolo da camada de transporte orientado a conexão.

O principal objetivo do protocolo é garantir que os nós não congestionem ainda mais a rede durante um período de sobrecarga. Para que esse objetivo fosse atingido, os autores desenvolveram dois mecanismos que o diferenciam de um protocolo baseado em *gossip* padrão. Sendo o primeiro mecanismo o uso do protocolo TCP, propiciando assim, o uso seguro da largura de banda disponível e a diminuição de perdas de mensagens motivadas pelo congestionamento da rede. E o segundo uma técnica de gerenciamento de *buffer*, que tem a finalidade de descartar mensagens em excesso e manter as informações relevantes.

Portanto, foi realizado um experimento com o jogo *Microsoft Flight Simulator 2002* através da execução de uma simulação com um número fixo de 500 nós. O que tornou possível demonstrar que os resultados do desempenho do protocolo foram satisfatórios dentre eles, tanto quanto a atomicidade, pois, aumentou o número de jogadores simultâneos de 5 para 20, quanto latência, devido ao aumento da mesma em decorrência da interrupção, quando possível, da transmissão de mensagens obsoletas.

3.2. HEAP

Frey et al. (2009) propõem em seu trabalho o protocolo HEAP (*Heterogeneity-Aware Gossip Protocol*), e o designam para distribuição de conteúdo colaborativo em ambientes heterogêneos. Os autores descrevem-no sobre duas perspectivas, sendo em primeiro lugar os resultados matemáticos confirmando a eficiência da disseminação de mensagens dos protocolos baseados em *gossip*, e, em segundo, a possibilidade da implementação de algum protocolo de agregação, que faz com que seja fornecido

continuamente a cada nó uma aproximação de sua capacidade de largura de banda relativa.

Dessa forma, são relatadas avaliações que permitem observar algumas melhorias desse protocolo quando comparado ao protocolo baseado em *gossip* tradicional, que tem funcionamento ideal em cenários homogêneos e em redes sem restrições, mas, se mostrou ineficiente em cenários heterogêneos. O HEAP, por sua vez, otimizou o uso da largura de banda, pois ele maximiza a utilização da banda de nós com grande capacidade, da mesma forma que minimiza os que possuem menor capacidade, melhorando assim, a utilização de largura de banda total e fazendo com que haja uma contribuição mais eficiente da largura de banda entre os nós.

Foram realizados experimentos com o número aproximado de 270 nós *PlanetLab* em uma aplicação de *streaming* de vídeo, comparando a adaptação proposta com o protocolo clássico. Então, ficou comprovado que o HEAP adapta a carga de cada nó de acordo com a sua largura de banda, melhora a qualidade de *streaming* de todos os nós, progride o *lag* de *stream* de 40% para 60% em relação ao *gossip* padrão, e, por fim, resiste bem a cenários de falha, diferente do que acontece no *gossip* genérico.

3.3. CREW

O CREW (*Concurrent Random Expanding Walkers*) foi desenvolvido por Deshpande et al. (2006) e tem como meta principal manter a escalabilidade e a resistência às falhas dos protocolos baseados em *gossip* continuando com a rápida transmissão de mensagens em redes heterogêneas. O tipo de disseminação utilizada pelo protocolo é a Disseminação *Flash*, a qual consiste em uma rápida propagação de informações para uma grande quantidade de nós em um período curto de tempo e tem como principais características: imprevisibilidade, escalabilidade e heterogeneidade da rede e de conteúdo.

A implementação do CREW se fez a partir do zero e foi feita através de uma plataforma *middleware* escalável, cujas conexões entre os nós usam o TCP, para que seja estimada a largura de banda disponível. Onde o Protocolo *Bounce*, que funciona como um serviço básico de *membership* passou a ser utilizado, e otimizações foram realizadas, o que tornou possível a diminuição do envio de mensagens redundantes, a redução da sobrecarga na rede e a sua adaptação em redes heterogêneas, fazendo com que a transmissão de dados seja continuamente rápida e eficiente.

O protocolo foi testado por meio da ferramenta *Modelnet,* que permite a moldagem do tráfego de rede em tempo real e a configuração de várias topologias de redes, e tomou como base alguns parâmetros, sendo uns deles: rapidez da disseminação de uma mensagem para nós espalhados em uma rede ampla, sobrecarga dos dados e adaptação em redes heterogêneas. Foi comparado com os sistemas de disseminação: *BitTorrent*, *Bullet*, *SplitStream* e *Asynchronous TCP Gossip*, o que permitiu enxergar que o CREW os demais sistemas tanto quanto ao desempenho, como também nos vários aspectos observados.

3.4. Análise e discussão

Os protocolos baseados em *gossip* estão sendo utilizados e modificados constantemente desde sua criação. Assim, a Tabela 1 apresenta um resumo dos protocolos abordados no

tópico anterior, onde os autores de todos os protocolos estudados propuseram otimizações quanto ao *gossip* genérico que se mostraram eficientes nos experimentos realizados.

Como pôde ser observado, os protocolos foram utilizados com objetivos e áreas de atuação diferentes. No entanto, todos mantiveram como foco principal a rápida propagação de informação, que é a finalidade primordial do protocolo de comunicação baseado em *gossip*. Além de realizarem otimizações que visam solucionar alguns pontos fracos do *gossip*, como a transmissão de mensagens redundantes e da sobrecarga na rede. Como mencionado na Seção 2, as mensagens redundantes e a sobrecarga na rede são consequência da forma de difusão das mensagens: os nós que precisam enviar uma mensagem, escolhem seus vizinhos randomicamente e, portanto, alguns vizinhos podem receber a mesma mensagem diversas vezes. Por outro lado, deve-se considerar que essa redundância auxilia na questão da confiabilidade do protocolo, pois, caso haja alguma falha de comunicação na rede, um nó poderá receber uma mensagem (considerada perdida) em um outro momento, caso seja escolhido por seu vizinho.

No NeEM uma otimização bastante significativa se deu pelo uso do TCP, que proporcionou uma confiável disseminação de mensagens mesmo durante congestionamentos na rede. E também o emprego de uma técnica de gerenciamento de buffer, que tem como principal função a eliminação de mensagens em excesso.

O HEAP considera que o *gossip* genérico não apresenta a mesma eficiência em redes heterogêneas como possui em redes homogêneas. Os autores otimizaram o protocolo, tendo como principal aprimoramento a maximização do uso da banda, onde há uma adaptação de carga de cada nó, a fim de evitar o congestionamento daqueles que possuem uma baixa capacidade de carga e de aumentar a contribuição dos nós que detém uma maior largura de banda.

No CREW os autores também se preocuparam com a perda de eficiência do *gossip* genérico em redes heterogêneas, otimizando o mesmo, com base do emprego de duas técnicas: redução de mensagens redundantes e diminuição da sobrecarga na concorrência entre os nós, que se tornaram possível através da utilização dos módulos *Bandwidth Manager*, que calcula e otimiza a largura de banda, e *Random Walker*, que garante a conexão entre os nós.

Tabela 1. Otimização do *gossip*

	NeEM	HEAP	CREW
Atuação	Congestionamento de redes	Distribuição de conteúdo colaborativo em ambientes heterogêneos	Disseminação Flash
Objetivo	Garantir que os nós não congestione ainda mais a rede durante um período de sobrecarga	Adaptar a contribuição de cada nó de acordo com sua largura de banda	Manter a escalabilidade, resistência a falhas e a rápida transmissão de mensagens em redes heterogêneas
Otimização	Uso do TCP e técnica de gerenciamento de buffer	Maximização do uso da banda	Utilização de conexões TCP, do Protocolo Bounce, entre outras
Experimento	Jogo Microsoft Flight Simulator 2002 com 500 nós	Aplicação de streaming de vídeo cerca de 270 nós	Ferramenta Modelnet
Resultado	Aumento do número de jogadores simultâneos e aumento da latência	Melhora na utilização da largura de banda	Redução do envio de mensagens redundantes, diminuição da sobrecarga na rede e adaptação em redes heterogêneas
Referência	Pereira et al. (2003)	Frey et al. (2009)	Deshpande et al. (2006)

4. Conclusões e trabalhos futuros

Este resumo apresentou um estudo sobre protocolos autonômicos e distribuídos baseados em *gossip*. Onde foi apresentado o *gossip* genérico com alguns parâmetros importantes e uma classificação bastante relevante. Por fim, foram exibidos exemplos otimizados de protocolos baseados em *gossip*, comprovando assim o uso desse tipo de protocolo em diversos âmbitos e a sua rapidez na propagação da informação.

Como contribuição, o presente resumo apresenta uma base para estudos mais aprofundados sobre os protocolos estudados e também para a implementação de otimizações aplicáveis aos mesmos. E como trabalhos futuros, pretende-se modelar e simular diferentes implementações de protocolos baseados em *gossip* com o objetivo de realizar uma avaliação quantitativa entre os mesmos.

Referências

COULOURIS, George; DOLLIMORE, Jean; KINDBERG, Tim. **Sistemas Distribuídos:** conceitos e projeto. 4ª ed. Bookman, 2007.

DEMERS, Alan; GREENE, Dan; HAUSER, Carl; IRISH, Wes; LARSON, John; SHENKER, Scott; STURGIS, Howard; SWINEHART, Dan; TERRY, Doug. **Epidemic algorithms for replicated database maintenance**. In: 6th ACM Symposium on Principles of distributed computing. Nova Iorque, 1987.

DESHPANDE, M.; XING, B.; LAZARDIS, I.; HORE, B.;

VENKATASUBRAMANIAM, N.; MEHROTRA, S. **Crew**: A Gossip-based Flash-

Dissemination System. In: 26th International Conference on Distributed Computing Systems. Washington, DC, EUA, 2006.

FREY, Davide; GUERRAOUIL, Rachid; KERMARREC, Anne-Marie; KOLDEHOFE, Boris; MOGENSEN, Martin; MONOD, Maxime; QUÉMA, Vivien. **Heterogeneous Gossip**. In: International Conference on Middleware. Nova Iorque, 2009.

HOLLERUNG, Tim Daniel; BLECKMANN, Peter. **Epidemic Algorithms**. Universitat Paderborn, 2004.

KUROSE, James F.; ROSS, Keith W. **Redes de Computadores e a Internet:** uma abordagem top-down. 5ª ed. Pearson, 2010.

LEITÃO, João Carlos Antunes. **Gossip-based broadcast protocols**. 2007. 92 f. Dissertação (Mestrado em Engenharia Informática) – Universidade de Lisboa. Lisboa, 2007.

MONTRESOR, Alberto. **Intelligent Gossip**. In: Intelligent Distributed Computing, Systems and Applications. Springer Berlin Heidelberg, 2008.

PEREIRA, J.; RODRIGUES, L.; MONTEIRO, M. J.; OLIVEIRA, R.; KERMARREC, A. M. **NeEM**: Network-friendly epidemic multicast. In: 22th Symposium on Reliable Distributed Systems. Florência, Itália, 2003.

RIVIÈRE, Etienne; VOULGARIS, Spyros. **Gossip-based networking for internetscale distributed systems**. In: E-Technologies: Transformation in a Connected World. p. 253-284. Springer Berlin Heidelberg, 2011.

TANENBAUM, Andrew S.; VAN STEEN, Maarten. **Sistemas Distribuídos:** princípios e paradigmas. 2ª ed. Pearson, 2008.