Balanceamento de Carga em Sistemas P2P Baseados em Clusters

Trabalho Individual II

CIN-UFPE

Edemberg Rocha da Silva
Fevereiro de 2011

Balanceamento de Carga em Sistemas P2P Baseados em Clusters

1. Introdução

Em sua natureza, os sistemas P2P (*peer-to-peer*) são caracterizados por possuírem um controle descentralizado, robustez, escalabilidade, estabilidade entre outras. Diante dessas características, vem crescendo o interesse de pesquisadores sobre esses sistemas. A natureza dos sistemas P2P oferece uma área vasta para pesquisas, não só no tocante ao compartilhamento de conteúdo, mas também em outros domínios que incluem infra-estrutura, busca, colaboração, redes sociais, roteamento, balanceamento de carga, segurança, tolerância a falhas dentre outros.

Apesar da grande quantidade de usuários que utilizam sistemas P2P, a capacidade desses sistemas proverem serviços com qualidade é questionada (QIAO, 2009). Um balanceamento de carga pode prover serviços com pequena taxa de falha e com um melhor desempenho, com isso a qualidade do serviço poderá ser melhorada. Além disso, sistemas baseados em *clusters* têm sido adotados para os serviços que são tolerantes a falha. Porém, apesar de um sistema de *cluster* ser adotado para melhorar a confiabilidade e robustez de um sistema P2P, o problema do desbalanceamento ainda persiste por causa da heterogeneidade dos *peers* (como larguras de banda, processadores, capacidade de armazenamentos diferentes, entre outros) e do total de suas requisições.

Alguns dos sistemas que trabalham com *clusters* apresentam dois níveis de conexões: intra-*cluster* e inter-*cluster*. E nesses sistemas o desbalanceamento pode ocorrer nesses dois níveis, portanto soluções de balanceamento de carga intra e inter-*cluster* têm sido pesquisadas.

Este trabalhado tem por objetivo apresentar o problema do não balanceamento de carga e algumas soluções/estratégias que foram propostas/adotadas pela comunidade científica. Porém, o foco deste trabalho está nos sistemas P2P baseados em *clusters* e que possuem os dois níveis de conexões citados no parágrafo anterior.

2. *Balanceamento de Carga em Sistemas P2P*

Balanceamento de carga é uma das propriedades dos sistemas P2P. Segundo FIORANO (2003), o balanceamento de carga visa lidar com a distribuição de solicitações entre servidores distintos, a fim de melhorar o desempenho da rede, aliviando aqueles *peers* que possuem *sobrecarga* (*hot peers*). Uma vez que os *hot peers* tornem-se um gargalo, eles tendem a aumentar o tempo de resposta ao usuário e, significativamente, degradam o desempenho do sistema (AYYASAMY et al., 2010).

De acordo com FIORANO (2003), muitas das estratégias de balanceamento de carga são para as topologias híbrida e de super-*peer*. Na primeira, vários servidores podem ser utilizados para melhor distribuir o processamento de consultas. De acordo com PIRES (2007), em se tratando da topologia de super-*peer*, os aspectos a serem considerados para manter um bom balanceamento de carga são:

- Quantidade de super-*peers*

  Um grande número de super-*peers* tendem a provocar um aumento na quantidade de mensagens trocadas na rede, durante o roteamento de consulta, de forma que a topologia venha a ter um comportamento similar a de uma topologia pura não-estruturada. Por outro lado, um número muito reduzido de super-*peers* pode fazer com que eles fiquem sobrecarregados por lidar com uma quantidade excessiva de *peers*. Portanto, a quantidade de super-*peers* deve ser adequada (ZHUANG et al., 2004).

- Métricas para a escolha do super-*peer*

  Métricas bem definidas precisam ser estabelecidas para que a escolha de um super-*peer* possa ser realizada. A escolha deve considerar o espaço para armazenamento, disponibilidade e largura de banda presentes no *peer* candidato a super-*peer* (JOHNSTONE et al., 2005; MONTRESSOR, 2004).

Quando um sistema P2P é arquitetado baseando-se em uma topologia, onde *peers* tendem a assumir alguns papéis, a tolerância a falhas torna-se mais susceptível a acontecer (PIRES, 2007). Considerando uma topologia de super-*peers*, caso um super-*peer* falhe, este não

poderá comprometer os demais *peers*. Neste caso, estratégias de tolerância a falhas devem estar predefinidas nos *peers*, considerando as métricas para a escolha do novo super-*peer* (se for o caso). Algumas pesquisas apontam estratégias para tolerância a falhas. Em (BRITO, 2005; JOHNSTONE et al., 2005) é proposto que a eleição do super-*peer* seja semelhante às métricas definidas no parágrafo anterior. Já JOUNGA et al. (2009) consideram, também, o número mínimo de *links* que o *peer* necessita estar conectado.

Ainda se tratando de tolerância a falhas, Li et al. (2004) sugerem a utilização de servidores de *backup*. Neste caso, os metadados armazenados no super-*peer* são periodicamente replicados em um super-*peer* de *backup*. Caso o super-*peer* venha a falhar, o super-*peer* de *backup* assume o papel do super-*peer*. ROUSE et al. (2006) propõem a presença de vários super-*peers* em um mesmo *cluster,* visando melhorias no balanceamento de carga e na tolerância a falhas.

Outras propriedades dos sistemas P2P, além do balanceamento de carga e tolerância a falha, podem ser encontradas em (PIRES, 2007). Porém, neste trabalho nos focaremos apenas no balancemanto de carga.

3. Balanceamento de Clusters

Balanceamento de carga é uma das questões abordadas em redes P2P *overlays* contemporâneas que incluem DHT (GODFREY et al., 2004; KARGER et al., 2004; PITOURA et al., 2006), super-*peers* (MONTRESOR, 2004) e *clusters* (AYYASAMY et al., 2010; JOUNGA et al., 2009; GAROFALAKIS et. al, 2009; MONDAL, 2006).

Em se tratando de sistemas P2P baseados em *cluster*, dois tipos de balanceamentos de carga fazem-se necessários: balanceamento *intra-cluster* e o balanceamento *inter-cluster* (AYYASAMY et al., 2010). A seguir, um levantamento sobre soluções para balanceamento de carga intra e inter-*cluster* serão apresentados e no final desta seção, um quadro resumirá os aspectos mais relevantes de cada solução apresentada a seguir.

3.1 Balanceamento de Carga Intra-*Cluster*

No caso de intra-*cluster*, o balanceamento de carga é realizado dentro de um *cluster* particular. De acordo com TRIANTAFFILOU et al. (2003), um balanceamento intra-*cluster* significa que, para cada *cluster*, todos os seus *peers* devem receber em média (ou aproximadamente) o mesmo número de requisições, do total de requisições que chegam ao

cluster. A seguir, apresentaremos as soluções propostas por diversos autores para a realização do balanceamento de carga intra-*cluster*.

- Popularidade dos *peers*

Em (MICHAIL, 2002) cada *cluster* replica seus documentos entre *peers*, no intuito de garantir o balanceamento de carga. Cada *peer* de um *cluster* é autorizado a criar réplicas para os documentos que ele detém. E cabe ao *peer* decidir onde e quando replicar o documento, baseado no conhecimento sobre a carga de tarefas dos membros do *cluster*. Esse conhecimento é obtido através do mecanismo do relato de popularidade de cada *peer*. A popularidade de cada *peer* é o percentual de sua carga de trabalho em relação à carga de trabalho do sistema. Além disso, para saber se houve um desbalanceamento de carga, periodicamente os *peers* verificam o índice de equilíbrio dos *clusters* (*fairness* - métrica utilizada para avaliar o estado de desbalanceamento do sistema) que é um valor compreendido entre 0 e 1. A medida que índice de equilíbrio aproxima-se do valor 1, o *cluster* necessita ser balanceado. Mas detalhes deste índice pode ser encontrado em (MICHAIL, 2002), (TRIANTAFILLOU, 2003), (JAIN et al., 1984).

A replicação de documentos e redirecionamento de requisições são as principais ações do balanceamento de carga intra-*cluster* apresentado por MICHAIL (2002). Todas as requisições de consultas que chegam a um *cluster* através de um *peer i* são redirecionadas ou não para outros *peers* de acordo com a capacidade de processamento dos mesmos. Todos os *peers* em um *cluster* podem processar consultas, devido ao repositório compartilhado de metadados que é mantido entre os *peers*. A probabilidade que uma consulta seja respondida por um *peer i* é dada pela fórmula apresentada na Figura 1.

$$p(i) = \frac{pc_i}{\sum_{j=1}^{N} pc_j}$$

Figura 1-Fórmula para o cálculo da probabilidade de um *peer* responder uma consulta (MICHAIL, 2002)

Onde pc_i é a capacidade de processamento do *peer i* e *N* é o número de *peers* no *clusters*. Se $\{P_1, ..., P_n\}$ é o conjunto de *peers* os quais contêm uma cópia dos documentos requisitados em uma consulta, então cada documento é recuperado de um *peer i*, com a probabilidade dada pela fórmula da Figura 1.

- Seleção aleatória de *peers*

Quando uma consulta chega a qualquer *peer* de um *cluster*, ela será respondida localmente a partir do *peer* consultado. Porém, caso a consulta não possa ser respondida localmente, um *peer* será selecionado aleatoriamente, considerando alguns metadados, para respondê-la. A seleção aleatória de *peers* pode garantir que os *peers* do *cluster* obtenham um compartilhamento igual de sua carga de trabalho sobre seu *cluster*. Assim, se a consulta for encaminhada para um *peer,* escolhido aleatoriamente, então a carga será balanceada entre os *peers* do *cluster*. Logo, o balanceamento de carga intra-*cluster* será alcançado (TRIANTAFFILOU et al., 2003).

- Tomada de decisão centralizada

A decisão de quando disparar o mecanismo de balanceamento intra-*cluster*, detectar *peers* sobrecarregados e a quantidade de dados a serem replicados ou migrados são tarefas críticas para o desempenho do sistema. E para resolver essas tarefas, MONDAL et al. (2003) adotaram uma abordagem chamada "tomada de decisão centralizada" para realizar o balanceamento intra-*cluster*. Nessa abordagem, cada *peer* envia, periodicamente, mensagens contendo estatísticas de suas tarefas executadas para o *líder* do *cluster*. O líder do *cluster* é um *peer* responsável por coordenar as atividades dos demais *peers* do *cluster*, além de gerenciar as informações sobre o conjunto de categorias de seu *cluster*, assim como as dos seus *clusters* vizinhos.

Em (MONDAL et al., 2003), um líder de *cluster* C_i recebe periódicamente informações sobre as cargas L_i e dispobibilidade de espaço em disco D_i dos *peers*. O líder cria uma lista ordenada pelas cargas L_i dos *peers*, onde o primeiro elemento da lista é o *peer* mais sobrecarregado. Considerando a lista com *n* elementos, e dentre os últimos *n/2 peers* da lista cujos respectivos valores de D_i forem menores que um limiar estabelecido pelo sistema, esses *peers* serão removidos da lista. Logo, o balanceamento de carga se dará pela migração ou remoção dos dados mais procurados do primeiro *peer* da lista para o último lugar , do segundo para o penúltimo e assim sucessívamente. E os dados só serão movidos (migrados ou replicados) se a diferença de carga entre os *peers* excederem o limiar pré-estabelecido no sistema.

MONDAL et al. (2003) acreditam que se a detecção da necessidade de um balanceamento de carga fosse realizada cada vez que um *peer* entrasse/saísse do sistema, isso iria resultar em condições indesejáveis para se trabalhar (*peers* se preocupando mais com o

balanceamento de carga do que com seu trabalho útil). Por isso, a verificação é realizada em determinados intervalos de tempo.

- Seleção de *peers*

GAROFALAKIS et al. (2009) analisam dois pontos que para eles são as principais razões para um desbalanceamento em uma distribuição de carga. O primeiro está relacionado com a preferência do usuário para *downloading* sobre *peers* com alta largura de banda. A segunda razão está relacionada com *peers* que atuam como *free-rides,* ou de uma forma geral *peers* que compartilham um pequeno número de documentos ou documentos com baixa popularidade.

Para tratar a primeiro problema citado anteriormente, um super-*peer* é utilizado para direcionar a consulta de um usuário para um *peer* selecionado pelo próprio sistema. A seleção desse *peer* é baseada nas informações sobre a largura de banda de conexão dos *peers* ou pela carga recebida até então por eles, lidando com estratégias de respostas que podem ser encontradas com mais detalhes em (GAROFALAKIS et al., 2009).

Com relação ao segundo problema, a estratégia de replicação é adotada. Neste caso, os documentos mais populares são replicados para um *cache* local de cada *peer*. O algoritmo utilizado por GAROFALAKIS et al. (2009) toma decisões baseadas nas seguintes questões: o que replicar? Quantos documentos serão replicados? Quais *peers* receberão uma réplica? Quantas réplicas por *peer*? De onde as réplicas são transferidas?

Além das questões anteiores, métricas são definidas para auxiliar o sistema a identificar situações de desbalanceamento e no tocante a qualidade do serviço. As métricas definidas em (GAROFALAKIS et al., 2009) são:

- Índice de Equilibrio *(Fairness Index*-FI): métrica que mostra a distribuição de carga entre *peers* em um *cluster*. Esse índice é baseado no que foi definido em (JAIN et al., 1984). Quanto mais o índice se aproximar do valor inteiro um, mais o balanceamento torna-se necessário.
- Qualidade de disponibilidade *(Quality of Availability)*: fração das requisições aceitas para a transferência de dados sobre o total de pedidos de transferência de dados.
- *Throughput*: número total de documentos transferidos, sem considerar os documentos que tiveram suas transferências iniciadas pelo processo de replicação.
- *Tamanho dos dados replicados (Size of Replicated Data)*: além do custo da transferência dos dados que foram replicados, ainda existe o custo para armazená-los. O tamanho total de dados replicados pode ser comparado ao tamanho total de documentos disponíveis

para determinar o peso desse custo do ponto de vista do sistema. Segundo GAROFALAKIS et al. (2009), a avaliação média dos documentos replicados por *peers* é importante também pois ela mostra a distribuição por cada *peer*.

- *Sugestões de cache local (Local Cache Hits)*: número de documentos satisfatoriamente requisitados pelos documentos que ainda têm de ser transferidos para o *cache* local durante a replicação.

- Listagem de *peers* segundo sua carga

De forma similar a MODAL et al. (2003), AYYSAMY et al. (2010) também levantam questões de quando detectar e a quantidade de dados a serem replicados. E a necessidade de um balanceamento é detectada pelo líder do *cluster*, assim como em (MODAL et al., 2003). A cada intervalo de tempo os *peers* enviam informações sobre seus espaços em disco e estatísticas sobre os seus trabalhos de carga aos líderes dos seus *clusters*. Baseada nas cargas desses *peers*, o líder do *peer* cria uma lista cujo primeiro elemento é o *peer* que possui a maior carga. E considerando N o tamanho dessa lista, serão removidos os ultimos N/2 elementos da lista, cujo espaço de armazenamento seja inferior ao limiar estabelecido pelo sistema. Esses passos encontram-se sumarizados no algoritmo exibido pela Figura 2.

O balanceamento é obtido através de uma replicação de dados na lista da mesma forma que ocorre em (MODAL et al., 2003), assim como a checagem da existência de um desbalanceamento.

Algorithm –Intra Cluster Load Balancing

1. For each $\{CL_i\}^{k}_{i=1}$
2. For each member $\{Pj\}^{n}_{j=1}$ of CLi
3. $P_{j,i}$ send Wj, i and $S_{j,i}$ to CL_i
4. CL_i add $P_{j,i}$ to the list $\{l_i\}$
5. End For
6. CL_i sort l_i such that $W_{j,i} > W_{j-1,i} > W_{j-2,i}$.....
7. For each $\{l_j\}^{n}_{j=n/2}$
8. If $S_{j,i} < S_{th}$ then
9. Delete the element $P_{j,i}$
10. End if
11. End For
12. If $Wa - W_b > \beta$ for any a, b < n, then
13. Move H1 (N_1) into N_n.
H2 (N_2) into N_{n-1} and so on.
14. End if
15. End For
16. End

Figura 2-Algoritmo de balanceamento de carga intra-cluster (AYYASAMY et al., 2010)

3.2 Balanceamento de Carga Inter-*Cluster*

No inter-*cluster*, o balanceamento de carga é realizado entre os *clusters* presentes no sistema. Uma definição mais formal do problema do balanceamento de carga inter-*cluster* pode ser encontrado em (TRIANTAFFILOU et al., 2003).

A seguir, apresentaremos as soluções propostas por diversos autores para a realização do balanceamento de carga inter-*cluster*.

- Migração de *peers*

Mecanimos de monitoramento e de relatos de popularidades, como vistos anteriormente quando apresentamos o balanceamento de carga intra-*cluster* de MICHAIL(2002), provêem aos peers do sistema uma forma de alerta sobre o balanceamento de carga do sistema. Cada *peer* verifica, periodicamente, se uma ação de balanceamento de carga inter-*cluster* deverá ser desempenhada. Se a necessidade de um balanceamento existir, *peers* poderão ser movidos para outros *clusters* através de uma operação chamada “migração de *peer*”.

Uma alternativa para balancear a carga é o reparticionamento de *peers* dentro de novos *clusters*. Contudo, com essa alternativa é difícil garantir que o novo particionamento

terá proximidade com a partição original (MICHAIL, 2002). No caso da nova partição derivar de uma velha, o custo de grande transferência de dados torna-se inaceitável. Uma estratégia alternativa para modificar a distribuição de carga é migrar a carga dos *peers,* distribuindo-as entre os *clusters*.

A migração do *peer* necessita apenas dos metadados dos *peers* para serem transmitidos e pode ser considerada uma operação de baixo custo, em relação ao montante de dados que é movido durante uma operação normal do sistema. Cabe ao líder de cada *cluster* calcular a carga que deverá ser movida para outros *clusters*. O tamanho atual do sistema, isto é o número de de *clusters*, é verificado antes da escolha do algoritmo de distribuição dos *peers*.

Finalmente, cada líder de *cluster* segue o resultado do algoritmo de distribuição e tenta selecionar os *peers* para migrá-los para outros *clusters*. Os *peers* selecionados são informados por uma mensagem de "migração" para que seja executado o procedimento de migração.

- Atribuição de categorias de documentos populares

TRIANTAFFILOU et al. (2003) afirmam que para garantir o alto desempenho do sistema abordado em seu trabalho, existe a necessidade de evitar gargalos e balancear a carga em todos os *peers* do sistema. Para TRIANTAFFILOU et al. (2003), carga é o número de requisições que chegam a um *peer* pertencente ao sistema. O balanceamento pode ser parcialmente obtido pela associação das categorias de documentos aos *clusters* de *peers*, de modo a garantir uma justa distribuição de categorias de documentos populares para os *clusters*. Baseado nisso, TRIANTAFFILOU et al. (2003) realizam o balanceamento de carga inter-*cluster*.

O objetivo do balanceamento de carga formalizado em (TRIANTAFFILOU et al., 2003) é criar *k clusters* de *peers* que terão igual popularidade e, consequentemente, igual carga. A popularidade de um documento é dada pela probabilidade que um usuário deseja recuperá-lo. Porém, nem todos os *cluster* são do mesmo tamanho. Logo, a popularidade deverá ser normalizada com relação ao número de *peers* nos *clusters* de modo que a carga média enfrentada por cada *peer* seja igual.

Para o banceamento de carga inter-*cluster* um algoritmo chamado MaxFair (TRIANTAFFILOU et al., 2003) é executado. Este considera cada categoria por vez e a atribui a um *cluster*. O critério adotado é o maior índice de equilíbrio (TRIANTAFFILOU et al., 2003) entre os *clusters* com popularidades normalizadas, a médida que eles surgem depois dessa atribuição. Quando uma nova categoria for atribuída a um dos *clusters*, todas as possíveis

atribuições são testadas e finalmente ela é atribuída ao *cluster,* considerando o *maximum fairness*, entre todas as possíveis atribuições.

- Colaboaração entre líderes de *clusters*

O balanceamento de carga inter-*cluster* é realizado através de um trabalho colaborativo entre os líderes dos *clusters* (MONDAL et al., 2003). A movimentação de dados de um *cluster* para um outro pode incorrer em alta sobrecarga de comunicação para justificar o movimento.

Periodicamente, os líderes de *clusters* trocam informações apenas com os líderes vizinhos. Se um líder α percebe que sua carga excede a carga média do conjunto β de seus líderes vizinhos em mais de 10% da média de carga, esse líder primeiro verifica os dados mais procurados que devem ser movidos. Em seguida, esse líder envia uma mensagem informando o espaço mínimo necessário para armazenar cada dado mais acessado nele, para cada líder vizinho em β, a fim de aliviar uma parte de sua carga entre eles. Os líderes β verificam em cada *peers* de seus *clusters* a disponibilidade de espaço em disco. Se algum desses *peers* satisfizer a condição de espaço em disco, os líderes informarão ao líder α sobre a disponibilidade de espaço disponível e a média de carga recebida pelos *peers*. α classifica os *peers,* indicados pelos líderes β, em uma lista $List_1$ onde seu primeiro elemento é o *peer* menos sobrecarregado (MONDAL et al., 2003).

Considere que os itens mais acessados são o h_1, h_2 h_3,... (onde h_1 é o elemento mais acessado), e considere *b* o número de *peers* contidos em β e *h* o número de dados mais acessados. Se $b < h$, h_1 é atribuído ao primeiro da lista de $List_1$, h_2 é atribuido ao segundo elemento e assim sucessivamente. Mas se $b \geq h$, a atribuição dos itens mais acessados é feita da mesma forma, porém alguns elementos de $List_1$ não receberão alguns desses itens. Mais detalhes poderão ser encontrados em (MONDAL et al., 2003).

Podemos observar, pelo texto contido em (AYYASAMY et al. , 2010), que seus autores adotaram para o balanceamento inter-*cluster*, o mesmo mecanismo utilizado no trabalho de (MONDAL et al., 2003). Mecanismo esse que já foi descrito anteriormente.

A Figura 3 sumariza os passos que realizam o balanceamento de carga inter-*cluster*.

Algorithm- Inter Cluster Load Balancing

1. Consider a cluster leader $CL_{k.}$
2. CL_k exchanges $\{Wi\}$ with $\{CL_i\}^{n}_{i=1}$
3. If $(Wk - W_{avg}) > (W_{avg} * 0.10)$ Then
   (W_{avg} is the average load of $\{CL_i\}^{n}_{i=1}$ and
   Wk is the load of CL_k)
4. For each member $\{Pj\}^{n}_{j=1}$ of CL_k
5. CL_k send S_j to $\{CL_i\}^{n}_{i=1}$
6. End For
7. For each $\{CL_i\}^{n}_{i=1}$
8. For each member $\{P_v\}^{n}_{v=1}$ of CLi
9. If $S_{v,i} > \text{Min}(\{S_j\}^{n}_{j=1})$ Then
10. Send $S_{v,i}$ and $W_{v,i}$ to CL_i
11. End if
12 End For
13. CL_i sends $\sum W_{v,i}$ and $\sum S_{v,i}$ to CL_k
14. CL_k add CL_i to the list $\{l_k\}$
15. End For
16. CL_k sort l_k such that $\sum W_{v,i} < \sum W_{v+1,i} < \sum W_{v+2,i} \ldots$
17. If $r < m$ Then
18. For each Hi of CL_k
19 Move H1 into CL1,
H2 into CL2 and so on.
20 End For
21 End if
22 Apply Intra-Cluster load balancing to $\{CL_i\}^{n}_{i=1}$
23 End if
24 End.

Figura 3- Algoritmo de Balanceamento Inter-Cluster (AYYASAMY et al., 2010)

- Monitoramento do número de *peers*

Apesar de não mostrar resultados, GAROFALAKIS et al. (2009) informam que o balanceamento de carga inter-*cluster* pode ser implementado por um mecanismo de monitoramento do número de peers em cada cluster. Esse mecanismo decidiria pela divisão dos *clusters* ou pela migração dos *peers* para outros *clusters*, garantindo o número igual de *peers* (ou valor aceitável estabelecido por alguma medida) entre os *clusters*. Parte desse mecanismo pode ser feito pelo processo de migração de *peers* para que a tranferência de dados possa ser feita em outros *clusters*.

- Movimento de *peers*

Diferente dos demais trabalhos, QIAO et al. (2009) adoratam o movimento de *peers* ao invés do movimento de dados para o processo de balanceamento de carga. Segundo eles, sem servidores virtuais, o balanceamento através do movimento de objetos só pode ser realizado entre *clusters* consecutivos em um sistema P2P baseado em *cluster*. Dessa maneira, o balanceamento de carga iria convergir lentamente. E com o balanceamento através do movimento de *peers,* são evitados: a sobrecarga de gerenciar a consistência dos dados, considerando o grande volume de *peers,* em diferentes *clusters* a serem movimentados; e as atualizações das tabelas de roteamento da rede.

Assim como os outros trabalhos, é necessário mensurar quando um balanceamento deverá ocorrer. Pensando nisso, QIAO et al. (2009) adoram a capacidade disponível como índice indicador de balanceamento de carga inter-*cluster*. Esse índice indicador é o que os autores chamam de "índice de carga" (*load index*). Esse índice foi proposto nos trabalhos (ZHU et al., 1998; BHASKARAM et al., 2003). A fórmula apresentada na Figura 4 denota a equação que calcula a capacidade disponível de um *peer*.

$$\text{AvailableCapacity} = \text{MaximumCapacity} * (1 - \text{utlization})$$

Figura 4-Fórmula para calcular a capacidade disponível de um *peer* (QIAO et al., 2009)

Utilizando capacidade disponível como o índice de carga, cada *cluster* interativamente executa um procedimento de balanceamento difuso que indentifica o estado de seu próprio *cluster* com relação a seus vizinhos. QIAO et al. (2009) consideram três esquemas: *directory-initiated*, *sender-initiated* e *receiver-initiated*. O *sender-initiated* é o *peer* que irá transferir sua carga, e o *receiver-initiated* é o *peer* que receberá a carga de um *sender*. Quando um *cluster* executa o procedimento de balanceamento, o *directory-initiated* localiza os que enviam (*senders*) e os que recebem (*receivers*) entre seus *clusters* vizinhos.

A execução do procedimento de balanceamento é disparada através de evento de *timeout* ou por uma mudança do estado em um *cluster* (passou a ser um *sender* ou *receiver*). O *cluster* determina o *status* de sua carga, assim como a de seus vizinhos, através de troca de mensagens entre eles. Nessas mensagens, informações sobre os índices de carga dos *clusters* vizinhos são fornecidas. Um parâmetro chamado *bound* é utilizado para determinar se um *cluster* está sobrecarregado ou não. A carga média de todos os *clusters* vizinhos é calculada. O limite mínino e máximo de carga são calculados pela fórmula = *average* – *load index* * (1 +/- *bound*). Onde o *bound* é dado em porcentagem da carga média (*average*). Logo, um *cluster* é um candidato a *receiver*(*sender*) de uma carga se seu índice de carga for maior(menor) do que o limite inferior obtido pela fórmula anterior. Todo esse cálculo visa auxiliar na tomada de

decisão de qual *cluster* deverá assumir o papel de *receiver* ou de *sender*, e enviar um pedido de transferência de carga para o receptor de cada *peer*, incluindo parâmetros como o ID do rementente e a quantidade de carga necessária que o *peer* necessita para atingir a carga média.

Após o *receiver cluster* receber do sistema a instrução para movimentação de *peer*, ele excluirá um dos seus membros e os enviará para o *sender cluster*. Enquanto que o *sender cluster* enviará o *peer* que se unirá ao *receiver cluster*. Um fato a ser a considerado é que com a movimentação do *peer* para o *receiver cluster*, este não poderá mudar seu estado de não sobrecarregado para sobrecarregado.

3.3 Comparativo das pesquisas sobre balanceamento

A tabela a seguir sumariza as estratégias adotadas pelas pesquisas observadas nesta seção. A Tabela 1 leva em consideração a estratégia de balanceamento tanto no nível intra quanto inter-*cluster* utilizada pelos autores. Além disso, métricas adotadas para a tarefa de balanceamento também são citadas.

Tabela 1-Comparativo das pesquisas sobre balanceamento de carga em sistemas P2P

Artigos	Balanceamento Inter-Cluster	Balanceamento Intra-Cluster	Outras características
AYYASAMY et al. (2010)	Migração de dados	Replicação de dados	• *Cluster* líder. • Replicação é feita em tempo de execução. • Envio de estatísticas de volume de carga aos *clusters* líder.
GAROFALAKIS et al. (2009)	Migração de *peers*	Replicação de dados	• Considera as seguintes características dos *peers*: largura de banda, capacidade de processamento, tempo *online.* • Conceito de *leaf, superNode* e *Candidate super-node.* • Métricas: *fairness índex, quality of availability, throughput, size of replicated data* e *local cache hits*.
MONDAL et al. (2003)	Migração de dados	Replicação/Migração de dados	• Realiza o cálculo da carga de um *peer*. • Os *peers* guardam estatísticas de acesso. • Em tempo de execução, decide qual estratégia de balanceamento de carga será utilizada. • *Cluster* líder • Algoritmo Orientado a Qualidade. • Abordagem *tomada de decisão centralizada.*

			• Verifica se está desbalanceado em determinados intervalos de tempo.
MICHAIL (2002)	Migração de *peers* (difusão)	Replicação de dados/Redirecionamento de requisições	• Métrica: índice de equilíbrio. • Categorização de documentos. • Monitoramento de popularidades. • Fases para balanceamento inter-*cluster*: *load evaluation, flow vector calculation, load selection* e *load migration*. • Gerenciamento de réplicas.
TRIANTAFILLOU et al. (2003)	Replicação de dados	Replicação de dados	• Utiliza *fairness index* para balanceamento de carga inter-*cluster*. • Considera a popularidade dos documentos. • Algoritmo" MaxFair" para balanceamento inter-*cluster*. • Realiza a normalização de popularidade.
QIAO et al. (2009)	Migração de *peers* (difusão)	- (não abordado)	• Balanceamento feito em determinado intervalo de tempo ou por algum evento ocorrido no *cluster*. • Propõem balanceamento difuso. • *Load index* é baseado na capacidade de armazenamento. • Capacidade de disponibilidade de um *peer* calculada considerando sua utilização. • Fases do procedimento para balanceamento: *LB triggering*, *load*

			determination, *decision* e *load transfer*.

4 Outros trabalhos

Alguns trabalhos aparecem na comunidade científica propondo balanceamento de carga em sistemas P2P, sem que seja utilizada a estrutura de *cluster*. Alguns desses trabalhos encontram-se em (PITOURA et al., 2006; LI et al., 2005; GODFREY et al., 2004; STOICA et al., 2003; NOVÁK, 2006; ABERER et al., 2003; JAGADISH et al., 2005; ASPNES et al., 2004; GANESAN et al., 2004; KARGER et al., 2004; RISSON et al., 2006).

5 Considerações Finais

O tamanho dos sistemas P2P e sua natureza dinâmica tornam o gerenciamento dos dados um desafio. Consultas enviadas pelos usuários necessitam de um mecanismo de balanceamento de carga que venha a diminuir o tempo de resposta para o usuário. Estratégias de formação de *clusters* tendem a diminuir esse tempo de resposta por agruparem os dados de forma categorizada. Porém, à medida que os clusters são construídos, estes não estão livres de sobrecarga. Logo, soluções para balanceamento de carga em sistemas P2P se fazem necessárias tanto no nível intra-*cluster* quanto no nível inter-*cluster*.

Neste trabalho foram apresentadas noções de balanceamento de carga em sistemas P2P, assim como a sua importância nesses tipos de sistemas. Foram apresentadas as diversas soluções propostas na comunidade cientifica, para os sistemas P2P baseados em *cluster*. No final, uma tabela resumindo as estratégias adotadas para o balanceamento de carga intra-*cluster* e inter-*cluster* foi exibida. No entanto, outros trabalhos sobre balanceamento de carga em sistemas P2P, sem o uso de *clusters* foram citados.

O próximo passo será verificar soluções para balanceamento de carga em sistemas gerenciadores de dados em P2P (PDMS- *Peer-to-peer Data Management System*).

6 Referências Bibliográficas

ABERER, K., MAUROUX, P. C., DATTA, A., DESPOTOVIC, Z., HAUSWIRTH, M., PUNCEVA, M., et al. (2003). P-Grid: A Self-organizing Structured P2P System. *ACM SIGMOD Record, vol. 32* , pp. 29-33.

ASPNES, J., KIRSCH, J., & KRISHNAMURTHY, A. (2004). Load balancing and locality in range-queriable data structures. *Proceedings of the twenty-third annual ACM symposium on Principles of distributed computing*, pp. 115-124. Newfoundland - Canada.

AYYASAMY, S., & SIVANANDAM, S. (2010). A Cluster Based Replication Architecture for Load Balancing in Peer-to-Peer Content Distribution. *International Journal of Computer Networks & Communications (IJCNC), vol.2*, pp. 158-172.

BHASKARAM, R., & KATZ, R. (2003). Load balancing and stability issues in algorithms for service compositon. *Proceedings of the Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies*, pp. 1477-1487. San Francisco - USA.

BRITO, G. A. (2005). *INTEGRAÇÃO DE OBJETOS DE APRENDIZAGEM NO SISTEMA ROSA - P2P.* Dissertação de Mestrado. Instituto Militar de Engenharia do Rio de Janeiro.

FIORANO. (2003). Super-Peer Architectures for Distributed Computing. White Paper, Fiorano Software, Inc.

GANESAN, P., BAWA, M., & GARCIA-MOLINA, H. (2004). Online balancing of range-partitioned data with applications to peer-to-peer systems. *Proceedings of the Thirtieth international conference on Very large data bases, vol. 30*, pp. 444-455. Toronto - Canada.

GAROFALAKIS, J., & MICHAIL, T.-A. (2009). Load Balancing in a Cluster-Based P2P System. *Proceedings of Fourth Balkan Conference in Informatics*, pp. 133-138.Thessaloniki - Greece.

GODFREY, B., KARP, R., LAKSHMINARAYANAN, K., SURANA, S., & STOICA, I. (2004). Load balancing in dynamic structured P2P systems. *Proceedings of INFOCOM.* Hong Kong - China.

JAGADISH, H. V., OOI, B. C., & VU, Q. (2005). BATON: A balanced tree structure for peer-to-peer networks. *Proceedings of the 31st international conference on Very large data bases* , pp. 661-672. Trondheim - Noruega.

JAIN, R. K., CHIU, D. W., & HAWE, W. R. (1984). A quantitative measure of fairness and discrimination for resource allocation in shared computer systems. *Techinical Report DEC-TR-301, Digital Computer Corporation.*

JOHNSTONE, S., SAGE, P., & MILLIGAN, P. (2005). iXChange – A Self-Organising Super Peer Network Model. *Proceedings of the 10th IEEE Symposium on Computers and Communications (ISCC'05), vol. 00*, pp. 164-169. Cartagena - Spain.

JOUNGA, Y.-J., & CHUANGB, F.-Y. (2009). OntoZilla: An ontology-based, semi-structured, and evolutionary peer-to-peer network for information systems and services. *Future Generation Computer Systems, vol. 25*, pp. 53–63.

KARGER, D. R., & RUHL, M. (2004). Simple efficient load balancing algorithms for peer-to- peer systems. *Proceedings of SPAA*, pp. 36–43. Barcelona - Spain.

LI, J., & VUONG, S. (2005). Ontology-Based Clustering and Routing in Peer-to-Peer Networks. *Proceedings of the 6th International Conference on Parallel and Distributed Computing, Applications and Technologies*, pp. 791 - 795. Dalian - China.

LI, Z. J., & LIAO, M. H. (2005). Modeling Load Balancing in Heterogeneous Unstructured P2P Systems. *Journal of Computer Science* , pp. 323-331.

MICHAIL, D. V. (2002). Lobster- A load balanced P2P content sharing network. *Masters thesis.* Technical University of Crete.

MONDAL, A., GODA, K., & KITSUREGAWA, M. (2003). Effective load-balancing of peer-to-peer systems. *Proceedings of the 6th IEEE International Symposium on Cluster Computing and the Grid*, pp.81-88. Área Central de Singapura - Singapura.

MONTRESOR, A. (2004). A robust protocol for building superpeer overlay topologies. *Proceedings of the 4th International Conference on Peer-to-Peer Computing*, pp. 202–209. Zurich - Switzerland.

MONTRESSOR, A. (2004). A Robust Protocol for Building Superpeer Overlay Networks. *Proceedings of the 4th International Conference on Peer-to-Peer Computing (P2P'04)*, pp. 202-209. Zurich - Switzerland.

NOVÁK, D. (2006). Load Balancing in Peer-to-Peer Data Networks. *Procedings of 2nd Doctoral Workshop on Mathematical and Engineering Methods in Computer Science*, pp. 1-7. Mikulov - Austrian.

PIRES, C. (2007). *Um Sistema P2P de Gerenciamento de Dados com Conectividade Baseada em Semântica.* Exame de Qualificação e Proposta de Tese. Universidade Federal de Pernambuco.

PITOURA, T., NTARMOS, N., & TRIANTAFILLOU, P. (2006). Replication , Load Balancing and Efficient Range Query Processing in DHTs. *Proceedings of the 10th International Conference on Extending Database Technology (EDBT).* Munich - Germany.

QIAO, Y., & BOCHMANN, G. (2009). Applying a diffusive load balancing in a clustered P2P system. *Proceedings of 9th Internacional Conference on New Technologies of Distributed Systems (NOTERE)*, pp. 189-199. Montreal - Canada.

RISSON, J., & MOORS, T. (2006). Survey of Research towards Robust Peer-to-Peer Networks. *The International Journal of Computer and Telecommunications Networking archive, vol. 50*, pp. 1-36. New York - USA.

ROUSSE, C., & BERMAN, S. (2006). A Scalable P2P Database System with Semi-Automated Schema Matching. *Proceedings of the 26th IEEE International Conference Workshops on Distributed Computing Systems (ICDCSW'06)*, pp. 78. Lisboa - Portugal.

STOICA, I., RAO, A., LAKSHMINARAYANAN, K., SURANA, S., & KARP, R. (2003). Load Balancing in Structured P2P Systems. *Proceedings of the IPTPS.* Berkeley - USA.

TRIANTAFFILOU, P., XIRUHAKI, C., KOUBARAKIS, M., & NTARMOS, N. (2003). Towards high performance peer-to-peer content and resource sharing systems. *Proccedings of CIDR.* Asilomar - USA.

ZHU, H., YANG, T., ZHENG, Q., WATSON, D., IBARRA, O. H., & SMITH, T. (1998). Adaptive Load Sharing for Clustered Digital Library Servers. *Proceedings of the 7th IEEE international Symposium on High Performance Distributed Computing.* Chicago - USA.

ZHUANG, Z., LIU, Y., & XIAO, L. (2004). Dynamic Layer Management in Super-Peer Architectures. *Proceedings of the International Conference on Parallel Processing (ICPP'04), vol. 00*, pp. 29-36. Montreal - Canada.