

Investigação Experimental do Balanceamento de Réplicas no Sistema de Arquivos do Apache Hadoop

Rhauani Weber Aita Fazul, Patricia Pitthan Barcelos

▶ **To cite this version:**

Rhauani Weber Aita Fazul, Patricia Pitthan Barcelos. Investigação Experimental do Balanceamento de Réplicas no Sistema de Arquivos do Apache Hadoop. Anais da XIX Escola Regional de Alto Desempenho da Região Sul (ERAD/RS 2019), SBC, Apr 2019, Três de Maio, RS, Brazil. hal-02414306

HAL Id: hal-02414306
https://hal.science/hal-02414306

Submitted on 16 Dec 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Investigação Experimental do Balanceamento de Réplicas no Sistema de Arquivos do Apache Hadoop

Rhauani Weber Aita Fazul[1], **Patrícia Pitthan Barcelos**[1]

[1]Laboratório de Sistemas de Computação (LSC)
Universidade Federal de Santa Maria (UFSM)
Santa Maria – RS – Brasil

{rwfazul, pitthan}@inf.ufsm.br

Resumo. *O mecanismo de replicação de dados é a base para o funcionamento do HDFS, o sistema de arquivos do Apache Hadoop. Seguindo uma política de posicionamento, as réplicas são dispostas entre os nós do cluster, porém não há garantias de um balanceamento efetivo durante esta distribuição. Este trabalho investiga mudanças de comportamento no HDFS mediante o balanceamento de réplicas, considerando a indução deliberada de falhas em diferentes cenários.*

1. Introdução

A tolerância e capacidade de recuperação perante a falhas é um requisito fundamental para promover alta disponibilidade em cenários de *Big Data*. Ferramentas especializadas, como o Apache Hadoop[1], implementam diversos mecanismos voltados a este fim, dentre eles a tradicional replicação de dados. Embora indispensável para garantir disponibilidade em caso de falhas, a replicação pode afetar diretamente o balanceamento do *cluster*.

No HDFS, o `NameNode` (NN) é o servidor mestre responsável pelo controle de acesso e a distribuição dos arquivos. Assim, quando o processo de replicação de dados é iniciado, cabe ao NN selecionar os `DataNodes` (DNs) - *workers* que realizam o armazenamento efetivo dos dados - para receberem as réplicas. Embora siga uma política de posicionamento de réplicas voltada a tolerância a falhas, o HDFS não garante que estas escolhas sejam realizadas de forma equilibrada entre os múltiplos DNs do *cluster*.

Este trabalho analisa a efetividade da política de posicionamento de réplicas implementada pelo HDFS. Para medir o impacto do desbalanceamento, o comportamento do HDFS foi avaliado durante a execução de operações de E/S, antes e após o balanceamento de réplicas no *cluster*. Adicionalmente, o processo de re-replicação de dados foi disparado a partir da indução deliberada de falhas em DNs específicos.

O artigo está organizado em cinco seções. O processo de replicação de dados e estratégias de balanceamento do Hadoop são apresentados na Seção 2. A Seção 3 descreve a metodologia adotada nos experimentos, enquanto a Seção 4 exibe e discute os resultados. A Seção 5 apresenta as considerações finais e direciona os trabalhos futuros.

2. Replicação de Dados no HDFS

Atuando de forma a garantir a disponibilidade dos dados em caso de falhas, a replicação é a base para o funcionamento do HDFS. Para cada arquivo a ser inserido no sistema, cria-se uma sequência de blocos de tamanho fixo (128MB a partir da versão 2 do Hadoop), que são duplicados com base em um fator de replicação definido *a priori* para cada arquivo.

[1]`https://hadoop.apache.org/`

Considerando um fator de replicação padrão de 3 réplicas por bloco, segue-se um modelo de distribuição *rack-aware*, otimizado para aumentar a confiabilidade e o desempenho do HDFS de acordo com a arquitetura do *cluster* [White 2015]. A estratégia corrente armazena a primeira réplica no DN local, se o cliente HDFS estiver executando em um nó *cluster*, ou em um DN escolhido arbitrariamente. As duas réplicas seguintes são armazenadas em diferentes DNs de um mesmo *rack* remoto, sendo este diferente do *rack* da primeira réplica. Todo este processo é otimizado a partir de um *pipeline* de escrita e envio de blocos simultâneo, estabelecido entre os três DNs selecionados.

Embora esta política de posicionamento contribua com a tolerância a falhas do HDFS, ela não é capaz, por si só, de assegurar a disponibilidade dos dados em caso de falhas recorrentes. Sendo assim, o processo de re-replicação dos blocos torna-se necessário.

2.1. Re-replicação dos Blocos

O monitoramento ativo é um requisito vital para manter a confiabilidade do HDFS. Para tal, os DNs devem se comunicar periodicamente com o NN através do envio de mensagens *heartbeat*. Falhas são detectadas pelo NN pela ausência destas mensagens por um intervalo de tempo previamente estipulado e, como resultado, o DN é considerado inativo.

A indisponibilidade de um DN faz com que o fator de replicação dos blocos nele armazenados seja decrementado. Quando o número de réplicas estiver abaixo do fator mínimo de replicação especificado, o NN inicia a re-replicação[2] destes blocos usando suas cópias. Assim, a conformidade dos blocos com o fator mínimo de replicação especificado é retomada, preservando a disponibilidade esperada dos dados caso novas falhas ocorram.

2.2. Balanceamento de Réplicas

Embora a política de posicionamento de réplicas favoreça um balanceamento mínimo (réplicas de um mesmo bloco não recaem em um mesmo DN), ela não é suficiente para manter o *cluster* balanceado. Em geral, qualquer DN que satisfaça as restrições impostas pela política é eletivo ao armazenamento da réplica, sendo a decisão final arbitrariamente realizada pelo NN, sem garantias de uma distribuição realmente homogênea dos blocos.

Um *cluster* desbalanceado pode afetar a localidade dos blocos, gerando sobrecarga para os DNs mais utilizados (nós que possuem mais blocos armazenados) e degradando o desempenho na execução de aplicações que realizem E/S. Neste sentido, estratégias para tornar o *cluster* balanceado tornam-se necessárias. Uma solução reativa para o desbalanceamento de réplicas é o Balanceador (HDFS *Balancer*) [Foundation 2018].

O Balanceador é uma ferramenta do Hadoop (modificada na versão 2) responsável pela análise do posicionamento dos blocos presentes no HDFS. Cabe ao Balanceador tomar decisões referentes à redistribuição dos blocos de DNs superutilizados para DNs subutilizados (seguindo a política de posicionamento de réplicas empregada).

Para tal, a execução é controlada por um *threshold* pré-definido (entre 1.0 e 100.0), que especifica a maior diferença permitida entre o uso de disco de um DN (relação do espaço utilizado para a capacidade total de armazenamento do DN) e a ocupação geral do *cluster* (relação do espaço utilizado e a capacidade total de armazenamento do *cluster*) [White 2015]. Um *threshold* menor resulta em um *cluster* mais equilibrado, porém maior será o esforço - em termos de tempo e processamento - necessário para o balanceamento.

[2] A corrupção de um bloco ou o aumento do fator também podem disparar o processo de re-replicação.

3. Metodologia

A metodologia utilizada na experimentação consistiu na submissão de testes de desempenho ao HDFS, considerando execuções sem estratégias para o balanceamento do *cluster* e execuções após o uso do Balanceador. Adicionalmente, cenários com a indução deliberada de falhas (variando de zero a cinco) foram idealizados, de modo a possibilitar a análise do impacto do processo de re-replicação de dados no balanceamento das réplicas.

A aplicação utilizada foi o TestDFSIO, um *benchmark* proprietário do Hadoop, capaz de medir o desempenho do HDFS em situações de leitura e escrita de dados em disco. O *benchmark* foi usado para escrita (e posterior leitura) de 10 arquivos de 10GB cada. Os parâmetros de configuração padrão do HDFS referentes ao fator de replicação e ao tamanho dos blocos de cada arquivo foram mantidos em 3 e 128MB, respectivamente. Para forçar a re-replicação após uma falha de DN, o fator mínimo de replicação também foi fixado em 3. Por padrão, o tempo limite necessário para o NN considerar um DN inativo é longo, o que evita disparar a re-replicação de blocos em caso de oscilações na comunicação. Em função do tempo de execução do *benchmark* utilizado, este intervalo foi reduzido para aproximadamente 20 segundos sem o recebimento de mensagens *heartbeat*.

Os experimentos foram realizadas na plataforma Grid'5000[3], onde foram configurados 10 nodos no *cluster* `suno` (rede Gigabit Ethernet) pertencente ao sítio Sophia-Antipolis, cada um com dois processadores Intel Xeon E5520 (quatro cores por CPU) com frequência de 2.27GHz, 32GB de memória RAM e 557GB de disco rígido, utilizando uma distribuição Debian 9.5 (*Stretch*). Cada nó foi responsável pela execução de um DN, enquanto o NN foi configurado para executar dentro do nó do primeiro DN.

4. Experimentação e Resultados

Os experimentos foram conduzidos com a versão 2.9.2 do Hadoop. A Tabela 1 apresenta o estado de ocupação do HDFS antes e após o uso do Balanceador com um valor de *threshold* de 5%. Verticalmente, para cada uma das configurações de teste (caracterizadas pelo número de falhas), exibe-se a porcentagem referente a quantidade de dados armazenada no disco de cada DN em relação ao total escrito durante a execução do *benchmark*.

Tabela 1. Ocupação no HDFS (%) antes e após o balanceamento.

Falhas	**0**		**1**		**2**		**3**		**4**		**5**	
Balanc.	**sem**	**com**	**sem**	**com**	**sem**	**com**	**sem**	**com**	**sem**	**com**	**sem**	**com**
DN01	8,24	8,24	-	-	-	-	-	-	-	-	-	-
DN02	15,47	13,37	8	10,28	-	-	-	-	-	-	-	-
DN03	9,88	9,72	18,08	14,19	13,74	13,7	-	-	-	-	-	-
DN04	7,45	7,34	18,21	9,17	9,37	9,35	21,88	18,13	-	-	-	-
DN05	7,54	9,31	7,75	10,92	19,85	16,61	12,77	11,64	10,82	15,46	-	-
DN06	21,19	13,12	9,12	15,09	9,13	9,1	9,8	14,59	26,17	20,44	28,12	21,47
DN07	6,87	10,32	9,17	10,11	8,97	11,43	12,42	12,17	10,9	13,94	16,1	21,72
DN08	8,24	8,12	13,37	8,53	15,39	15,35	12,27	12,42	10,47	14,9	8,36	19,35
DN09	6,7	11,31	8,63	12,1	8,74	9,69	18,55	18,26	11,15	15,33	10,17	16,69
DN10	8,42	9,15	7,67	9,61	14,81	14,77	12,31	12,79	30,49	19,93	37,25	20,77
Ideal	10		11,11		12,5		14,28		16,67		20	
DP	4,68	2,05	4,34	2,26	4,09	2,98	4,28	2,82	9,14	2,78	12,36	2,07

No HDFS sem balanceamento percebe-se um desiquilíbrio de carga acentuado no sistema, com DNs subutilizados e superutilizados. Com o uso do Balanceador, é possível

[3]Grid'5000 é uma plataforma para experimentos apoiada por um grupo de interesses científicos hospedado por Inria e incluindo CNRS, RENATER e diversas Universidades, bem como outras organizações (mais datalhes em `https://www.grid5000.fr`).

aproximar a quantidade de dados armazenada em cada DN ao valor ideal esperado para um *cluster* balanceado, dado pela razão do volume total de dados armazenados no HDFS pela quantidade de DNs ativos ('Ideal' na Tabela 1). Mesmo que uma variação controlada nos valores seja permitida em função do *threshold* configurado, é possível observar a efetividade do balanceamento pela redução no desvio padrão (DP) obtido em cada cenário após o uso do Balanceador, indicando que o uso dos DNs ficou próximo do valor ideal.

Além de promover uma distribuição mais homogênea dos blocos no sistema de arquivos, o balanceamento do *cluster* proporciona melhorias em seu uso. Alguns aspectos de desempenho, correspondentes às médias aritméticas recuperadas a partir dos *logs* de 20 execuções distintas com o TestDFSIO em modo leitura, são exibidos na Tabela 2. Através do balanceamento reduz-se a localidade dos blocos armazenados no HDFS. Assim, a capacidade do sistema em processar conjuntos de tarefas paralelas e independentes consegue ser melhor explorada, possibilitando o aumento observado do *throughput* e da taxa média de transferência de dados durante a execução da aplicação. Com o balanceamento de réplicas, nota-se também um ganho de desempenho significativo no tempo total necessário para a leitura dos dados armazenados no HDFS.

Tabela 2. Desempenho obtido pelo balanceamento do HDFS.

Falhas	**0**		**1**		**2**		**3**		**4**		**5**	
Balanceamento	**sem**	**com**	**sem**	**com**	**sem**	**com**	**sem**	**com**	**sem**	**com**	**sem**	**com**
***Throughput* (MB/s)**	44,65	51,52	52,43	61,73	51,58	57,22	53,68	61,52	44,41	51,74	51,23	57,68
Aumento (%)	-	13,33	-	15,07	-	9,86	-	12,74	-	14,17	-	11,18
Taxa de E/S (MB/s)	55,29	62,45	53,57	65,37	63,79	69,37	73,09	85,47	49,69	57,64	58,02	65,02
Aumento (%)	-	11,47	-	18,05	-	8,04	-	14,48	-	13,79	-	10,77
Tempo de leitura (sec)	302,3	272,58	309,19	274,01	312,28	283,22	318,45	291,54	325,14	295,21	348,27	319,18
Redução (%)	-	9,83	-	11,38	-	9,31	-	8,45	-	9,21	-	8,35

5. Considerações Finais

Este trabalho conduziu uma investigação experimental do balanceamento de réplicas no HDFS. Os experimentos ressaltaram o desiquilíbrio de carga ocasionado pela atual política de posicionamento de blocos, demonstrando que a localidade dos dados é afetada pelos processos de replicação e re-replicação. Diferentes benefícios de desempenho foram alcançados a partir do balanceamento do *cluster* efetivado com o HDFS *Balancer*.

Além do Balanceador, soluções que otimizem a distribuição das réplicas podem ser empregadas ao HDFS [Abad et al. 2011, Lin and Lin 2015]. Com o uso de uma estratégia que melhor explore a localidade dos blocos, nota-se potencial em promover expressivas melhorias no desempenho do sistema. Assim, trabalhos futuros envolvem o estudo e criação de algoritmos voltados a uma distribuição balanceada de réplicas no HDFS.

Referências

Abad, C. L., Lu, Y., and Campbell, R. H. (2011). Dare: Adaptive data replication for efficient cluster scheduling. In *Int. Conf. on Cluster Computing*, pages 159–168. IEEE.

Foundation, A. (2018). Hdfs users guide - balancer. `https://hadoop.apache.org/docs/r2.9.2/hadoop-project-dist/hadoop-hdfs/HdfsUserGuide.html#Balancer`. Dezembro.

Lin, C.-Y. and Lin, Y.-C. (2015). A load-balancing algorithm for hadoop distributed file system. In *Int. Conf. on Network-Based Information Systems*, pages 173–179. IEEE.

White, T. (2015). *Hadoop: The Definitive Guide, 4th Edition*. "O'Reilly Media, Inc.".