Análise Comparativa de Desempenho em Drivers para MongoDB em Aplicações Node.js

Leandro Ungari Cayres, Ronaldo Celso Messias Correia

[1]Faculdade de Ciências e Tecnologia – Universidade Estadual Paulista (UNESP)
Presidente Prudente – SP – Brazil

{leandro.ungari,ronaldo.correia}@unesp.br

Abstract. *This meta-paper describes the style to be used in articles and short papers for SBC conferences. For papers in English, you should add just an abstract while for the papers in Portuguese, we also ask for an abstract in Portuguese ("resumo"). In both cases, abstracts should not have more than 10 lines and must be in the first page of the paper.*

Resumo. *Este meta-artigo descreve o estilo a ser usado na confecção de artigos e resumos de artigos para publicação nos anais das conferências organizadas pela SBC. É solicitada a escrita de resumo e abstract apenas para os artigos escritos em português. Artigos em inglês deverão apresentar apenas abstract. Nos dois casos, o autor deve tomar cuidado para que o resumo (e o abstract) não ultrapassem 10 linhas cada, sendo que ambos devem estar na primeira página do artigo.*

1. Introdução

Nos últimos anos, o crescimento no volume de dados mudou a utilização desses por empresas e organizações; tais dados, inicialmente considerados agentes passivos, relacionados às regras de negócio empresarial; tornaram-se potenciais oportunidades de lucro através da análise de informações, e consequentemente, do conhecimento presente no conjunto de dados.

Essa crescente quantidade de dados, chamado de *Big Data*, não somente requer maior espaço de armazenamento, mas uma mudança em sua organização com base em cada contexto, considerando características como volume, variedade, velocidade e valores [Ward and Barker 2013]. A arquitetura dos tradicionais bancos de dados relacionais, baseada no modelo ACID (*atomicity*, *consistency*, *isolation* e *durability*), não se encaixa de modo adequado em ambientes de *big data*, principalmente devido a forte consistência dos dados e a garantia de integridade dos relacionamentos.

Nesse cenário surgem os banco de dados NoSQL ("Not only SQL") provendo maior flexibilidade estrutural, suporte a replicação e consistência eventual consistência seguindo o critério BASE (*basic*, *availability*, *soft-state* e *eventual consistency*) [Han et al. 2011]. Nos últimos anos, a popularidade dos banco de dados não-relacionais tem crescido [Cooper et al. 2010, Edlich 2015] proporcionando diversas soluções conforme características dos dados cada aplicação.

Diferentemente do modelo relacional, os bancos NoSQL não possuem uma linguagem comum entre eles que permita a realização de operações, como o SQL (*Structured*

Query Language), desse modo, cada banco provê uma interface nativa para manipulação dos dados. Contudo, uso de chamadas de sistema para execução de comandos dessas interfaces não é reconhecida como uma boa prática, que pode incorrer em diversos problemas.

De modo a contornar essa situação, para os diferentes ambientes de desenvolvimento e linguagens de programação, *drivers* tem sido desenvolvidos de modo a viabilizar a execução dos comandos no banco de dados. Em muitas situações, a decisão de qual combinação entre banco de dados não-relacional e *driver* a ser empregada pode ser um problema, devido a variedade de possibilidades, assim como o desconhecimento dos pontos positivos e negativos de cada solução.

Neste artigo, por meio de um estudo comparativo, busca-se avaliar as duas principais soluções de *drivers* para o MongoDB [1], respectivamente MongoClient [2] e Mongoose [3], em ambientes de aplicação Node.js. O principal fator considerado para a realização desse análise consiste na definição prévia de esquema para a manipulação dos dados em operações de CRUD (*create-read-update-delete*). Conceitualmente, os banco de dados não-relacionais não requerem esse predefinição, proporcionando flexibilidade, contudo não existe nada que impeça sua utilização, principalmente quanto a respeito do desempenho; possibilitando algum impacto relevante.

A escolha do banco de dados MongoDB ocorreu devido a sua enorme popularidade recente, empregado em diversas aplicações e linhas de pesquisa; o qual consiste na principal opção dentre os bancos de dados que adotam a estratégia de armazenamento orientada a documentos. A respeito do ambiente Node.js, apesar de uma tecnologia recentes alguns trabalhos apontam a sua viabilidade no desenvolvimento de aplicações [Chaniotis et al. 2015]; além disso, com essa escolha, tanto aplicação quanto banco de dados utilizam JavaScript, permitindo a elaboração de um sistema uniforme, em termos de linguagem de programação.

Na análise conduzida neste trabalho, o desempenho do banco de dados MongoDB atrelado a cada um dos *drivers* investigados é analisado sobre perspectivas de tempo de execução, tempo de uso de processamento e consumo de memória sob um conjunto de dados genérico, propiciando a identificação de algumas características.

O restante desse trabalho está organizado do seguinte modo: na Seção 2 é realiza uma revisão conceitual sobre banco de dados não-relacional, assim como sobre o MongoDB e seus drivers. A Seção 3 é apresentado o estudo comparativo realizado neste trabalho. Seção 3.3 são apresentados os resultados quantitativos obtidos, cuja discussão é elaborada na Seção 4. Em seguida, na Seção 5 e, por fim, na Seção 7 as considerações finais desse trabalho.

[1] https://www.mongodb.com/

[2] https://mongodb.github.io/node-mongodb-native/

[3] https://mongoosejs.com/

2. Banco de Dados Não-Relacional

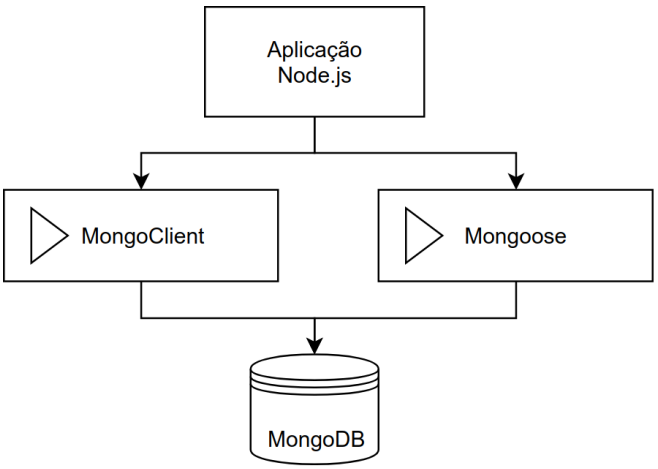
2.1. MongoDB

2.1.1. MongoClient

2.1.2. Mongoose

3. Estudo Comparativo

Dentre as perspectivas de análise desse trabalho consiste na comparação de dois drivers, em que um utiliza definição previamente e outro não, assim



Q1 – *O tamanho médio dos registros impacta de modo relevantes quanto ao uso de memória nas operações de CRUD?*

Q2 – O tamanho médio dos registros pode influenciar no uso de CPU nas operações de CRUD?

Q3 – O tamanho médio dos registros impacta no tempo de execução de cada uma das operações de CRUD?

3.1. Dataset

O presente estudo utilizou um conjunto de dados com cerca 18 mil instâncias de dados, proveniente do seguinte dataset [4]. Originalmente, todos os registros presentes são compostos por 89 atributos, predominante textuais, obtendo um tamanho médio de 1,37KB. A partir do conjunto original, foi construído um conjunto reduzido em número de atributos (6 atributos), com o mesmo total de instância, porém com tamanho médio de 0,13KB.

3.2. Ambiente de Execução

O ambiente de execução para os testes de desempenho consistiu em um computador pessoal com sistema operacional Ubuntu 18.04.2, processador Intel i3 3217U e memória RAM de 5GB.

[4] https://www.kaggle.com/karangadiya/fifa19

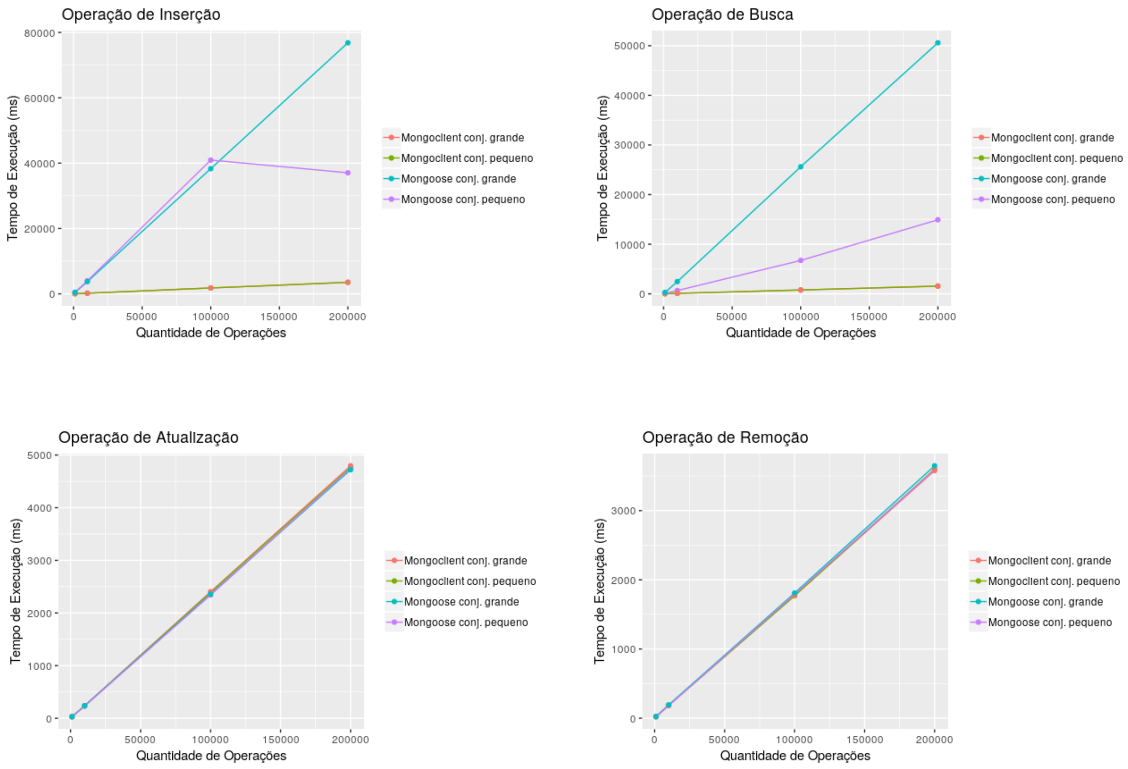


Figura 1. The average and standard deviation of critical parameters

Durante a execução dos testes, o ambiente de execução da aplicação Node.js foi definido o uso do *heap* de memória com limite máximo de 3GB, o que restringiu no limite superior do número de operações executadas em cada cenário de teste.

Em cada cenário de execução, foram extraídos dados relativos ao tempo de execução, tempo de uso de CPU e uso de memória RAM. Foram analisados cenários com diferentes quantidades de operações CRUD realizadas, as quais variaram de 1000, 10000, 100000 e 200000; cada qual foi repetido 10 vezes.

3.3. Resultados

Esses são os resultados de tempo

Esses são os resultados de uso de cpu

Esses são os resultados de uso de memoria

4. Discussão dos Resultados

4.1. Discussão da Q1

4.2. Discussão da Q2

4.3. Discussão da Q3

5. Ameaças à Validação

6. Trabalhos Relacionados

The first page must display the paper title, the name and address of the authors, the abstract in English and “resumo” in Portuguese (“resumos” are required only for papers written

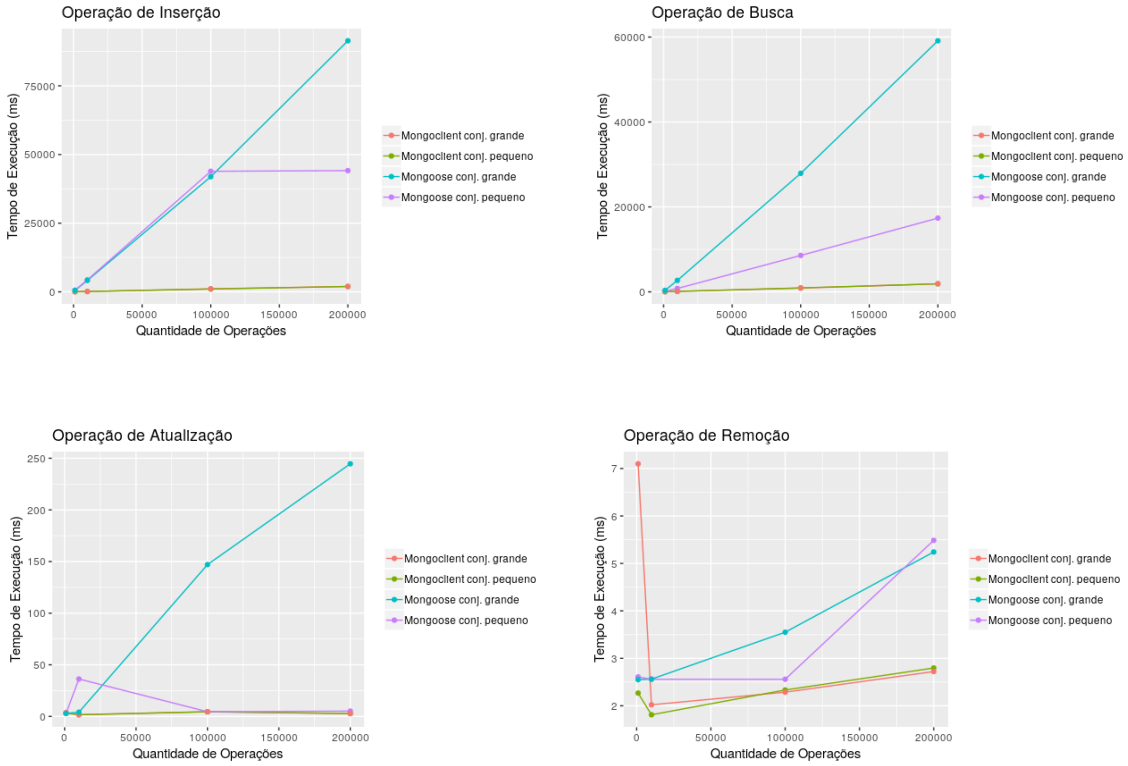


Figura 2. The average and standard deviation of critical parameters

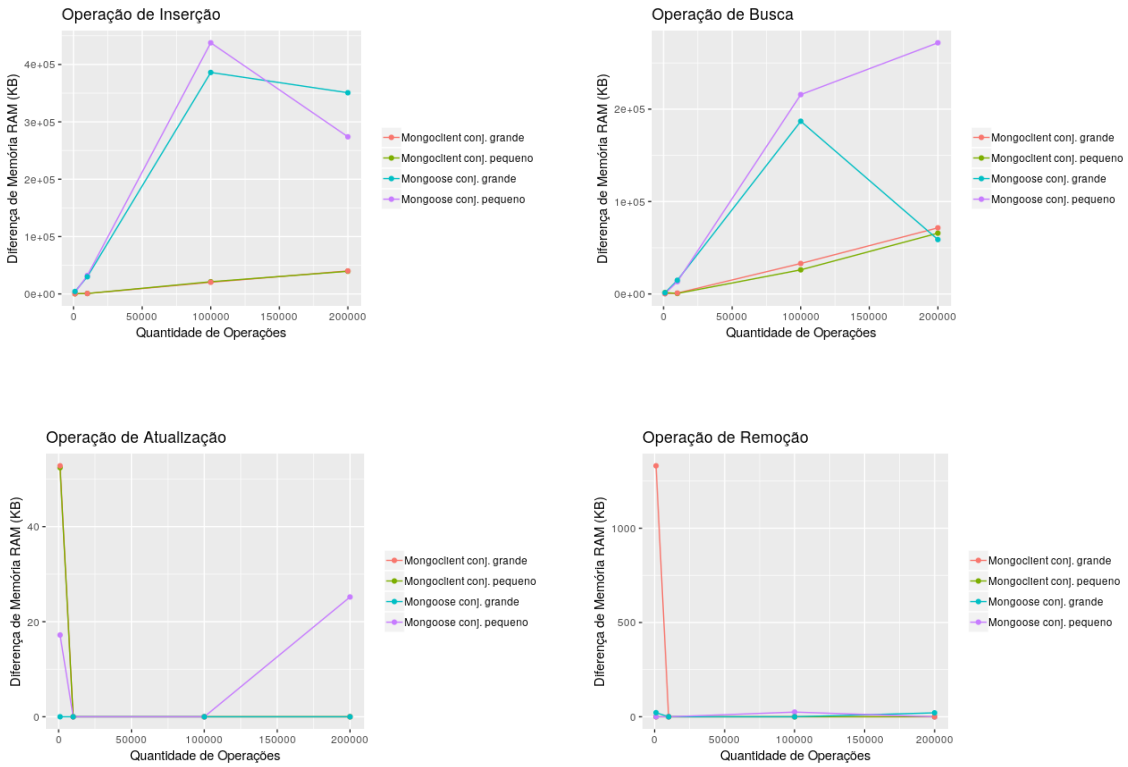


Figura 3. The average and standard deviation of critical parameters

in Portuguese). The title must be centered over the whole page, in 16 point boldface font and with 12 points of space before itself. Author names must be centered in 12 point font, bold, all of them disposed in the same line, separated by commas and with 12 points of space after the title. Addresses must be centered in 12 point font, also with 12 points of space after the authors' names. E-mail addresses should be written using font Courier New, 10 point nominal size, with 6 points of space before and 6 points of space after.

The abstract and "resumo" (if is the case) must be in 12 point Times font, indented 0.8cm on both sides. The word **Abstract** and **Resumo**, should be written in boldface and must precede the text.

7. Considerações Finais

8. Referências Bibliográficas

Referências

Chaniotis, I. K., Kyriakou, K.-I. D., and Tselikas, N. D. (2015). Is node. js a viable option for building modern web applications? a performance evaluation study. *Computing*, 97(10):1023–1044.

Cooper, B. F., Silberstein, A., Tam, E., Ramakrishnan, R., and Sears, R. (2010). Benchmarking cloud serving systems with ycsb. In *Proceedings of the 1st ACM symposium on Cloud computing*, pages 143–154. ACM.

Edlich, S. (2015). Nosql-your ultimate guideto the non-relational universe! nosql.

Han, J., Haihong, E., Le, G., and Du, J. (2011). Survey on nosql database. In *2011 6th international conference on pervasive computing and applications*, pages 363–366. IEEE.

Ward, J. S. and Barker, A. (2013). Undefined by data: a survey of big data definitions. *arXiv preprint arXiv:1309.5821*.