Teste de performance em bancos de dados NoSQL: Apache Cassandra vs. Neo4j

Amanda V. Soares
João Vítor F. Sonego
Leonardo O. Spilere

Introdução

- No dinâmico cenário da tecnologia da informação, "Big Data" surge como divisor de águas em banco de dados, enfrentando a crescente geração diária de dados pela sociedade e empresas. Caracterizado por conjuntos volumosos, complexos e multifornecedores, o Big Data redefine os sistemas tradicionais.
- O conceito central é a eficiente coleta e processamento de grandes volumes de dados, não limitado ao armazenamento, mas incluindo a extração de insights valiosos. Com a rápida evolução tecnológica, os bancos de dados tradicionais enfrentam limitações, impulsionando a busca por soluções mais flexíveis.
- Organizações, visando aproveitar o Big Data, adotam novos paradigmas, como bancos de dados NoSQL, desafiando estruturas rígidas e explorando novas formas de armazenamento e análise em larga escala.

Objetivo

O presente trabalho tem como objetivo comparar a performance do tempo de execução de inclusões, atualizações, remoções e consultas em dois bancos de dados NoSQL: Apache Cassandra e Neo4j.

Banco de dados NoSQL

- Os bancos de dados NoSQL são uma categoria de sistemas de gerenciamento de banco de dados que se destacam por sua abordagem não relacional.
- Os bancos de dados NoSQL utilizam diversos modelos de dados, como: Documentos, Grafos, Colunas e chave-valor.
- Permite armazenar e processar dados de forma mais ágil e escalável, tornando-os ideais para aplicações que envolvem grandes volumes de informações não estruturadas ou semiestruturadas.

Teorema CAP

- Consistência (Consistency) - em que os dados retornados serão os mesmos para todos os usuários quando vistos ao mesmo tempo;
- Disponibilidade (Availability) - em que os usuários sempre receberão um retorno, mesmo que o sistema esteja parcialmente inacessível;
- Tolerante a Falhas/Partição (Partition tolerance) - em que o sistema continua o processamento independentemente de quantas falhas de comunicação ocorram durante o processo.

Cassandra

- Modelo de Dados Orientado a Colunas – desenvolvido para consultas eficientes e de alto desempenho;
- Distribuído e Descentralizado – projetado para ser executado em múltiplas máquinas;
- Escalabilidade Elástica – escolha adequada para sistemas com cargas de trabalho variáveis;
- Alta Disponibilidade e Tolerância a Falhas/Partições – garantindo que os dados estejam acessíveis, mesmo em caso de falhas em nós individuais;
- Consistência Ajustável e Consultas Ricas – permite escolher o nível de consistência dos dados, equilibrando desempenho e consistência.

Neo4j

- Alternativa aos tradicionais bancos de dados relacionais;
- Armazena informações em estruturas de grafo e relacionamentos.;
- Prioriza a Consistência e a Disponibilidade;
- Eficaz quando se lida com dados altamente interconectados;
- Flexibilidade no modo de armazenar informações;
- Maior e mais ativa comunidade dentre os bancos de grafo;
- Possui versão Open-Source.

Metodologia

- API REST em C#
  a. Post – Inserção;
  b. Get – Consulta;
  c. Put – Atualização;
  d. Get by Id – Consulta com identificador;
  e. Delete – Remoção.

- Requisições: Apache JMeter
  a. Único cliente – 1.000 requisições;
  b. Múltiplos clientes (600) – 5 requisições.

Metodologia

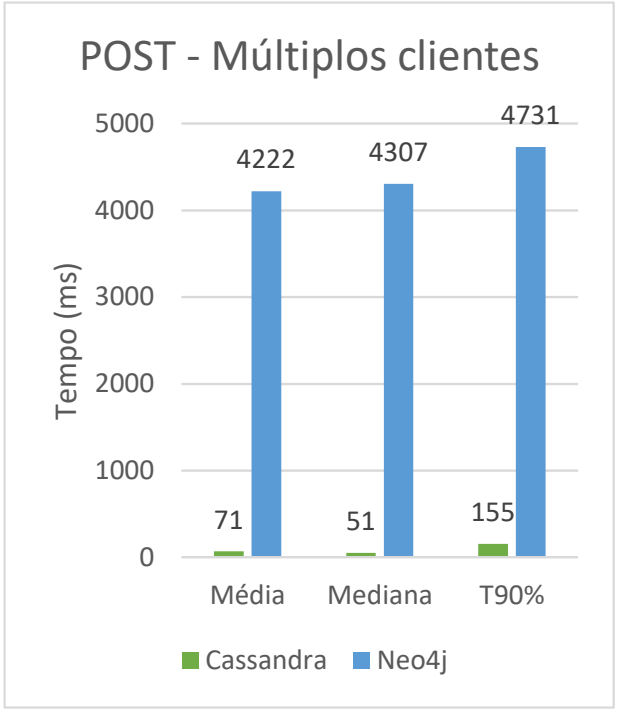
- Requisições com conteúdo aleatório

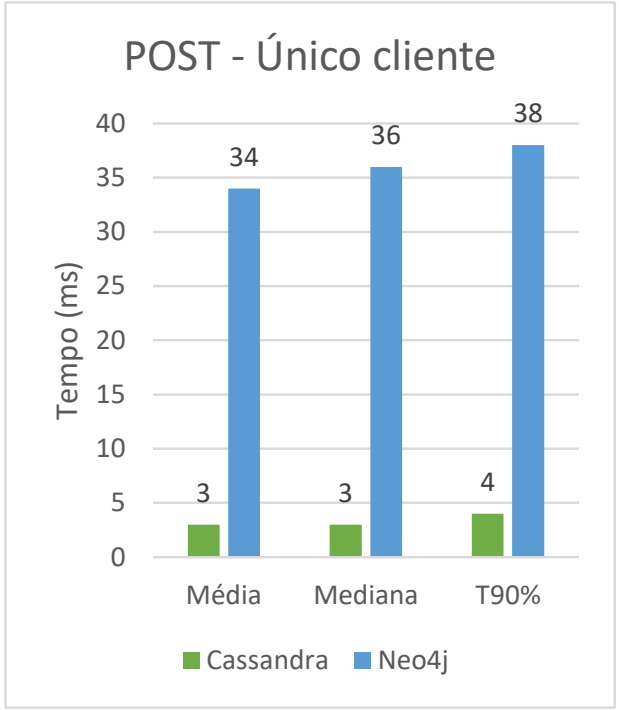
Propriedade	Tipo da propriedade		
	C#	Cassandra	Neo4j
Id	int	int (PK)	integer
Texto	string	text	string
Numero	int	int	integer
Num_Decimal	float	float	float
Data	DateTime	timestamp	Local DateTime

Metodologia

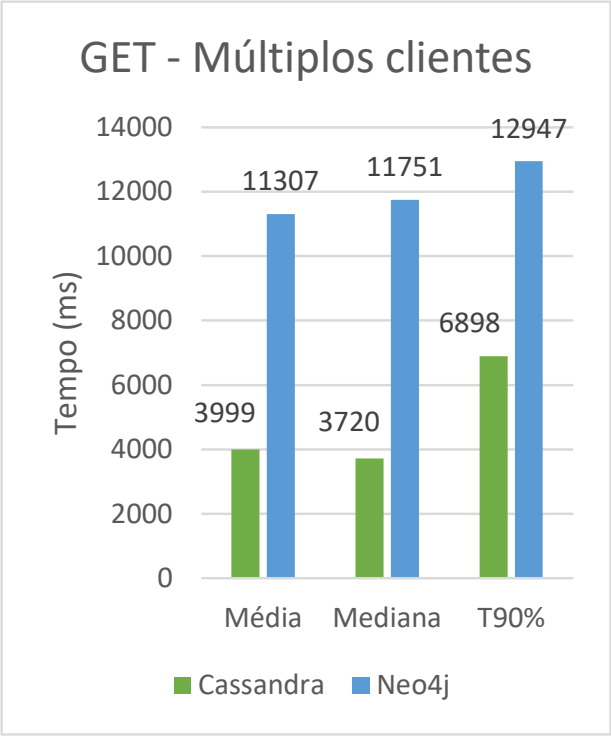
- Requisições com conteúdo aleatório
- Ambiente de testes:
  - CPU: AMD Ryzen 5 3500X
  - RAM: 16GB
  - SSD: 240GB
  - Sistema operacional: Windows 10
- Comparação de performance entre os tempos utilizando:
  a. Média;
  b. Mediana;
  c. T90%;

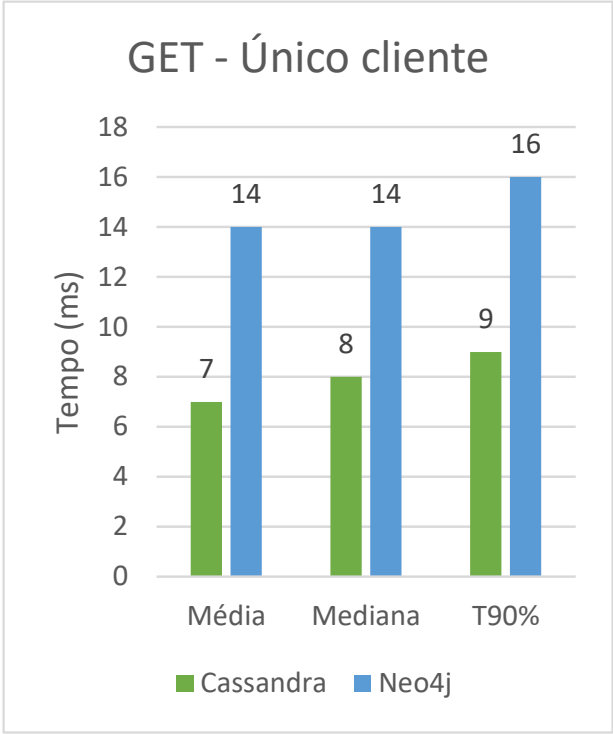
Resultados


POST - Múltiplos clientes
Tempo (ms)
5000
4000
3000
2000
1000
0
4222
4307
4731
71
51
155
Média
Mediana
T90%
Cassandra
Neo4j

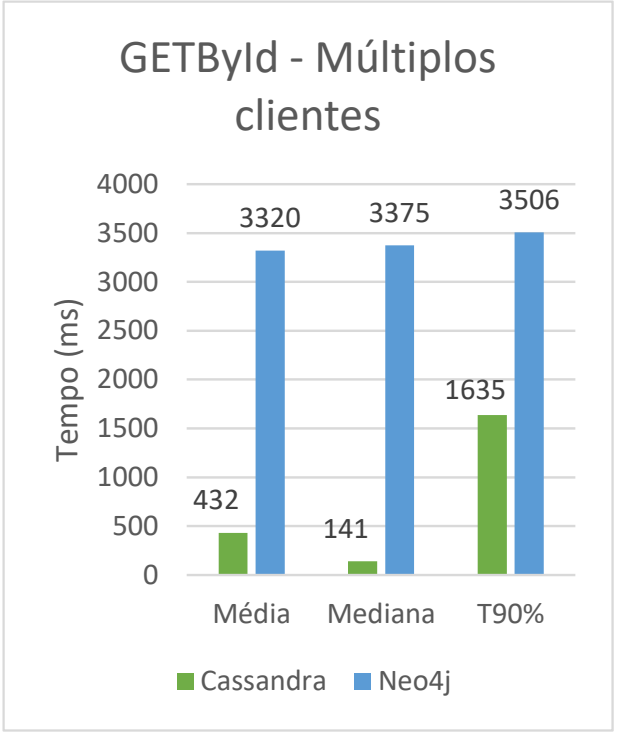

POST - Único cliente
Tempo (ms)
40
35
30
25
20
15
10
5
0
34
36
38
3
3
4
Média
Mediana
T90%
Cassandra
Neo4j

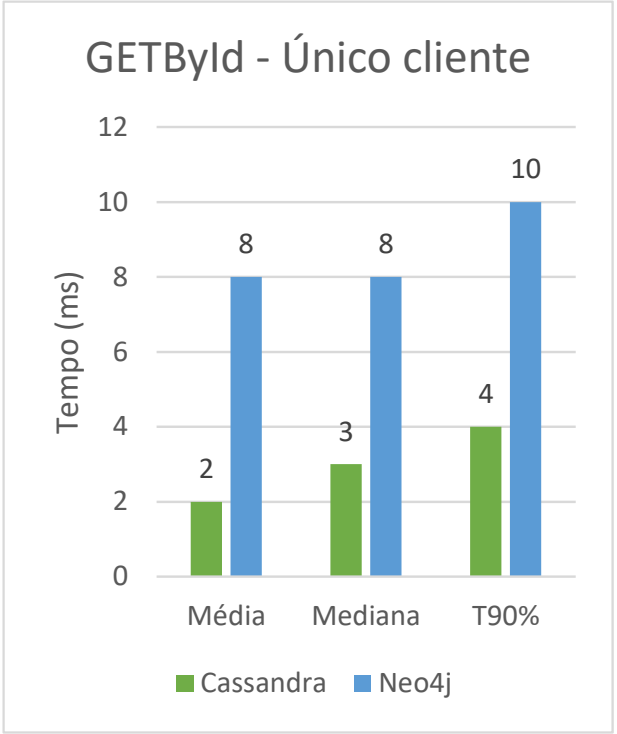
Resultados


GET - Múltiplos clientes
Tempo (ms)
14000
12000
10000
8000
6000
4000
2000
0
3999
11307
3720
11751
6898
12947
Média
Mediana
T90%
Cassandra
Neo4j

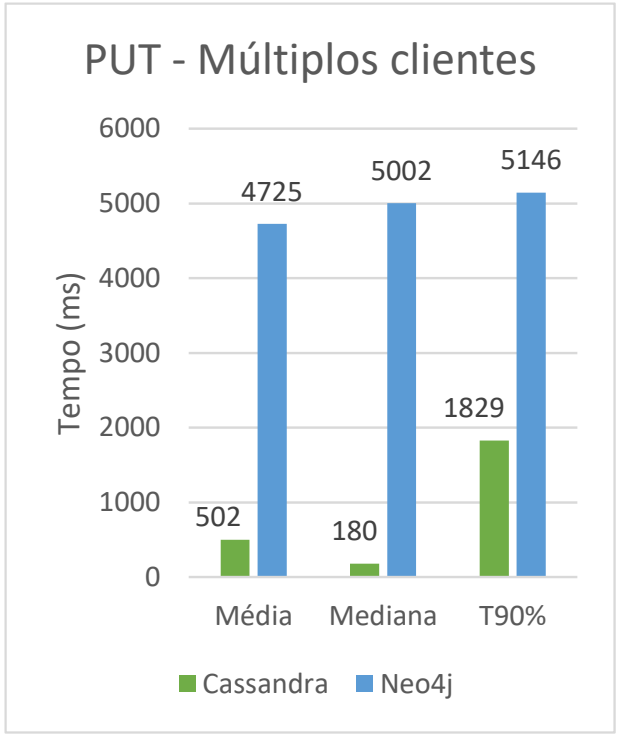

GET - Único cliente
Tempo (ms)
18
16
14
12
10
8
6
4
2
0
7
14
8
14
9
16
Média
Mediana
T90%
Cassandra
Neo4j

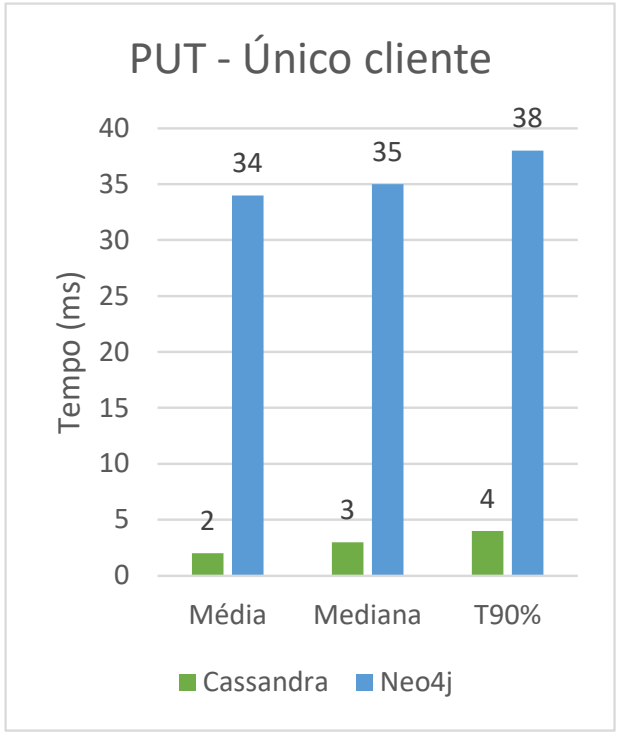
Resultados


GETById - Múltiplos clientes
Tempo (ms)
4000
3500
3000
2500
2000
1500
1000
500
0
3320
3375
3506
1635
432
141
Média
Mediana
T90%
Cassandra
Neo4j

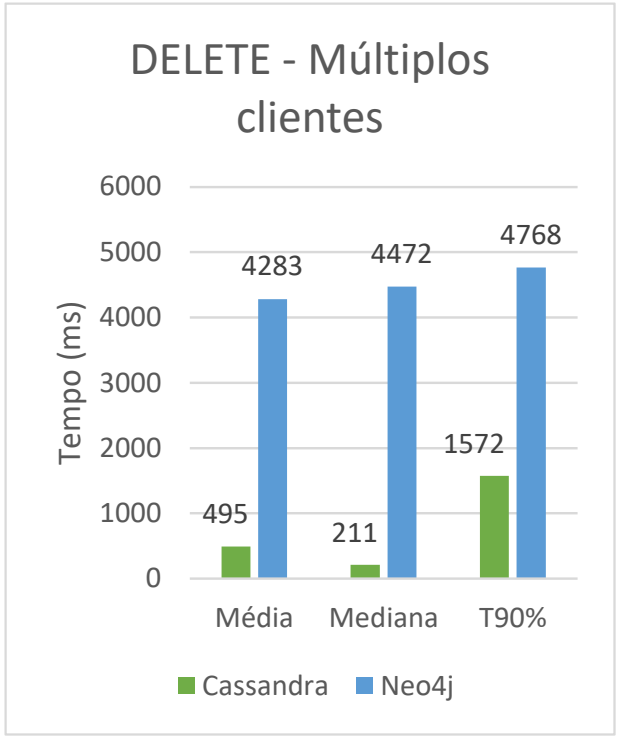

GETById - Único cliente
Tempo (ms)
12
10
8
6
4
2
0
8
8
10
2
3
4
Média
Mediana
T90%
Cassandra
Neo4j

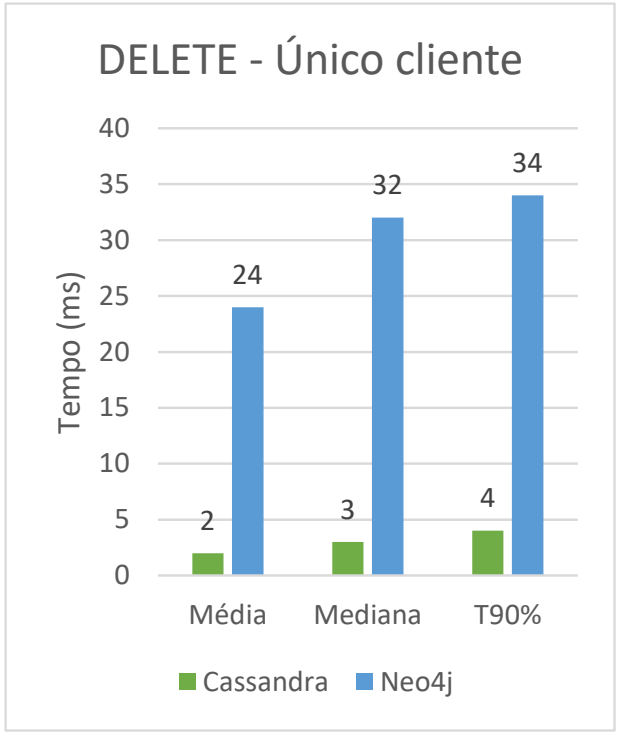
Resultados


PUT - Múltiplos clientes
Tempo (ms)
6000
5000
4000
3000
2000
1000
0
4725
5002
5146
1829
502
180
Média
Mediana
T90%
Cassandra
Neo4j


PUT - Único cliente
Tempo (ms)
40
35
30
25
20
15
10
5
0
34
35
38
2
3
4
Média
Mediana
T90%
Cassandra
Neo4j

Resultados


DELETE - Múltiplos clientes
Tempo (ms)
6000
5000
4000
3000
2000
1000
0
4283
4472
4768
495
211
1572
Média
Mediana
T90%
Cassandra
Neo4j


DELETE - Único cliente
Tempo (ms)
40
35
30
25
20
15
10
5
0
24
32
34
2
3
4
Média
Mediana
T90%
Cassandra
Neo4j

Conclusão

- Único cliente possui melhor desempenho do que múltiplos clientes;
- Cassandra possui melhor desempenho do que Neo4j;
  - Caso Genérico vs. Caso específico;
  - Qual a influência dos drivers utilizados?

- Implementar testes em máquina propícia para servidor;
- Realizar testes com maior número de dados para avaliar escalabilidade.

Teste de performance em bancos de dados NoSQL: Apache Cassandra vs. Neo4j

Amanda V. Soares
João Vítor F. Sonego
Leonardo O. Spilere