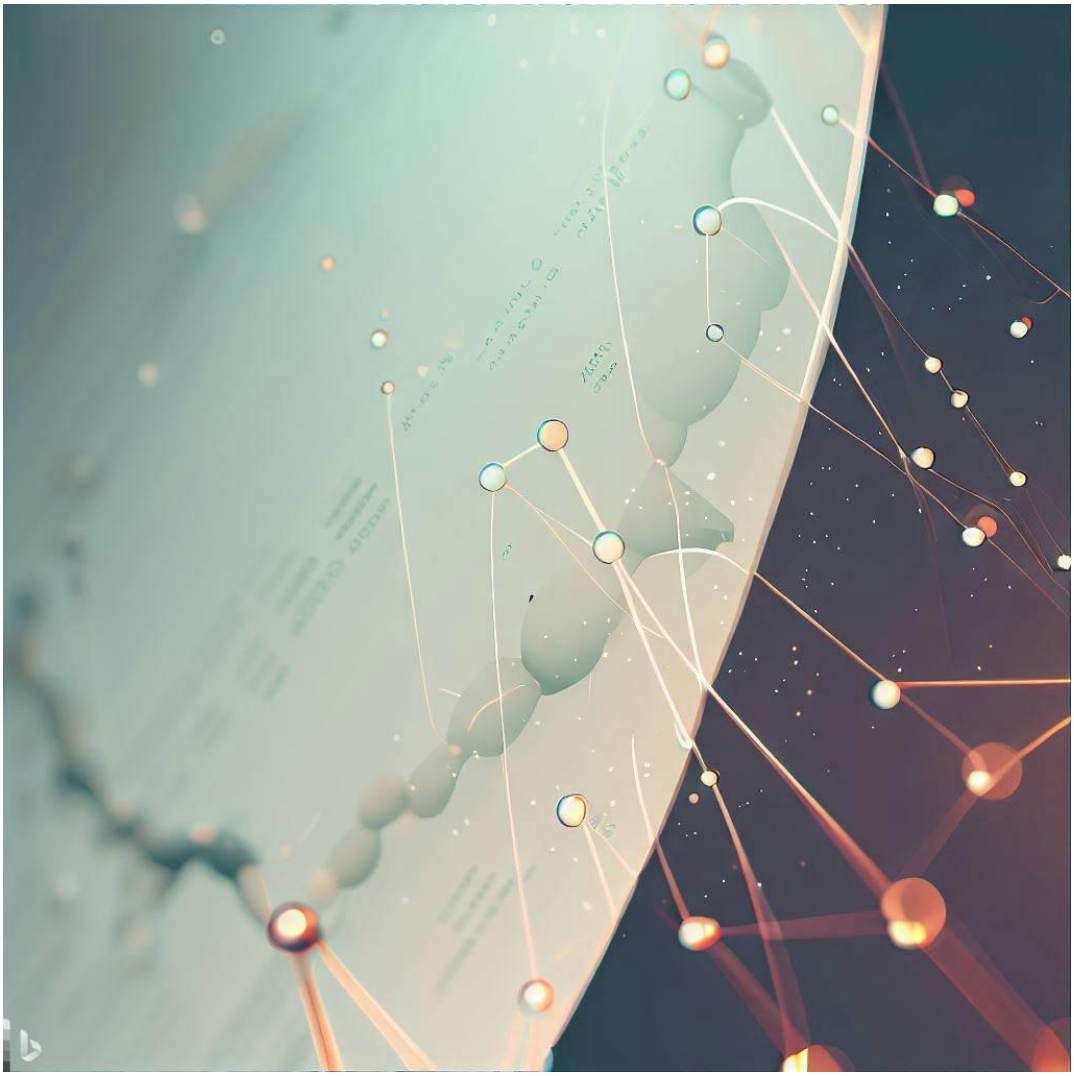
Competentie: Onderzoeken

Onderzoeksrapport en resultaten



Student: Levi Leuwol
Studentnummer: 405538
Datum: 2023-06-30
Onderwerp: Competentie: Analyseren
Opleiding: HBO-ICT
Studiejaar: 3

Hanzehogeschool Groningen
University of Applied Sciences

Inhoudsopgave

1 Abstract

Dit onderzoeksrapport presenteert een vergelijkende analyse van de prestaties tussen het gebruik van een Object Graph Mapper (OGM) en de standaard Neo4j-driver bij het ophalen van alle nodes uit een Neo4j-database. Het doel van dit onderzoek was om inzicht te krijgen in de impact van de OGM-functionaliteit op de verwerkingstijden en variabiliteit van de metingen.

De dataset bestond uit 2349 nodes, 7678 relaties, 19998 eigenschappen en 2321 labels. De metingen betroffen de tijdsduur in milliseconden om alle nodes op te halen uit de database.

De resultaten toonden aan dat het gebruik van OGM resulteerde in een gemiddelde tijdsduur van 52.1171 ms, terwijl de standaard Neo4j-driver een gemiddelde tijdsduur van 39.065 ms vertoonde. Bovendien vertoonden de metingen met OGM een hogere mate van variabiliteit in vergelijking met de standaard Neo4j-driver, zoals blijkt uit de analyse van de standaarddeviatie en variantie.

Deze bevindingen hebben praktische implicaties voor het ontwerp en de implementatie van systemen die gebruikmaken van een Neo4j-database. Afhankelijk van de vereisten van het systeem, kan het gebruik van de standaard Neo4j-driver de voorkeur hebben vanwege kortere verwerkingstijden en een meer voorspelbaar gedrag. Het is echter belangrijk om rekening te houden met de specifieke context en vereisten van het systeem bij het maken van deze keuze.

Hoewel dit onderzoek enkele beperkingen kent, zoals de beperkte dataset en de focus op het ophalen van alle nodes, draagt het bij aan het begrip van de prestatie-implicaties van het gebruik van OGM in een Neo4j-database. Toekomstig onderzoek kan zich richten op het verkennen van andere aspecten en scenario's, evenals het identificeren van optimalisaties voor OGM om de prestaties verder te verbeteren.

Dit onderzoek biedt waardevolle inzichten voor ontwikkelaars en beheerders die gebruikmaken van Neo4j-databases, en kan dienen als basis voor weloverwogen beslissingen bij het kiezen tussen OGM en de standaard Neo4j-driver.

2 Introductie

2.1 Inleiding

Het onderzoeksrapport presenteert een analyse van metingen met betrekking tot de tijdsduur voor het ophalen van alle nodes uit een Neo4j-database. Het onderzoek is gebaseerd op een dataset bestaande uit 2349 nodes, 7678 relaties, 19998 eigenschappen en 2321 labels.

De efficiënte opslag en verwerking van grote hoeveelheden gegevens zijn essentieel geworden in het moderne tijdperk van gegevensbeheer. Met zijn grafenstructuur biedt Neo4j een krachtig platform voor het beheren en verkennen van complexe relaties tussen entiteiten.

Symfony wordt gebruikt als een robuuste en flexibile API-laag om toegang te bieden tot de database. Een belangrijk onderdeel van de Symfony API is de Object Graph Mapper (OGM) functionaliteit, die de vertaling tussen objectgeoriënteerde modellen en de grafenstructuur van Neo4j mogelijk maakt.

Dit onderzoek is gemotiveerd door de behoefte om de prestaties van de Neo4j-database te verbeteren en de impact van OGM op de prestaties te onderzoeken. De focus van het onderzoek ligt op het ophalen van alle nodes uit de Neo4j-database. De resultaten van het onderzoek kunnen gebruikt worden om een keuze te maken tussen het gebruik van OGM of het gebruik van de standaard Neo4j-driver.

2.2 Onderzoeksvraag

De onderzoeksvraag die in dit rapport wordt behandeld, luidt: “Wat is het verschil in prestaties tussen het gebruik van de Object Graph Mapper (OGM) en de standaard Neo4j-driver bij het ophalen van alle nodes uit een Neo4j-database?”

Deze onderzoeksvraag is geformuleerd om een vergelijking mogelijk te maken tussen de prestaties van een OGM en de standaard Neo4j-driver bij het uitvoeren van een specifieke taak, namelijk het ophalen van alle nodes uit de Neo4j-database. Hierbij wordt gekeken naar de tijdsduur die nodig is om deze taak uit te voeren.

2.3 Doelstellingen

De volgende doelstellingen zijn geformuleerd om de onderzoeksvraag te beantwoorden:

1. Het vergelijken van de prestaties van een OGM en de standaard Neo4j-driver bij het ophalen van alle nodes uit een Neo4j-database.
2. Het bepalen van de invloed van de grootte van de dataset op de prestaties van een OGM en de standaard Neo4j-driver bij het ophalen van alle nodes uit een Neo4j-database.
3. Het identificeren van eventuele prestatieverschillen tussen een OGM en de standaard Neo4j-driver bij het ophalen van alle nodes uit een Neo4j-database.
4. Het bieden van inzicht in de geschiktheid van een OGM voor het ophalen van alle nodes uit een Neo4j-database.

2.4 Verwachte resultaten

Op basis van de hiervoor genoemde doelstellingen worden de volgende resultaten verwacht:

1. Het verschil in tijdsduur tussen het gebruik van een OGM en de standaard Neo4j-driver bij het ophalen van alle nodes uit een Neo4j-database.
2. Mogelijke prestatievoordelen of -nadelen van een OGM ten opzichte van de standaard Neo4j-driver.
3. Aanbevelingen voor het gebruik van een OGM of de standaard Neo4j-driver.

2.5 Scope van het onderzoek

Het onderzoek richt zich specifiek op het vergelijken van de prestaties van OGM en de standaard Neo4j-driver bij het ophalen van alle nodes uit een Neo4j-database. Andere aspecten van de Neo4j-database of andere taken worden buiten beschouwing gelaten in dit onderzoek.

2.6 Structuur van het rapport

1. **Introductie:** In dit hoofdstuk wordt een introductie gegeven op het onderwerp van het onderzoek.
2. **Methodiek:** In dit hoofdstuk wordt de methodiek van het onderzoek beschreven.
3. **Resultaten:** In dit hoofdstuk worden de resultaten van het onderzoek gepresenteerd.
4. **Alternatieven** In dit hoofdstuk worden alternatieven voor het gebruik van een OGM of de standaard Neo4j-driver besproken.
5. **Conclusie:** In dit hoofdstuk wordt een conclusie gegeven op basis van de resultaten van het onderzoek.

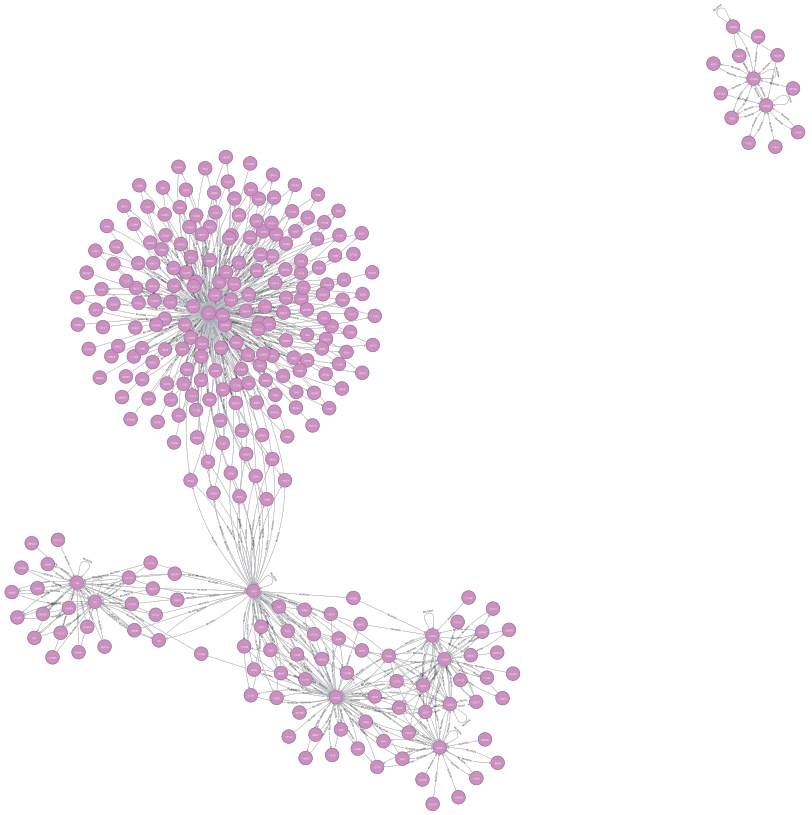
3 Methodiek

3.1 Onderzoeksopzet

Om de onderzoeksvraag te beantwoorden en de impact van OGM op de prestaties te onderzoeken, is een onderzoeksopset gehanteerd. Het opzet wordt beschreven in dit hoofdstuk.

3.2 Dataset

Voor dit onderzoek is gebruikgemaakt van een dataset bestaande uit 2349 nodes, 7678 relaties, 19998 eigenschappen en 2321 labels. Dit is de dataset die beschikbaar is gesteld door de opdrachtgever van het project waar dit onderzoek onderdeel van uitmaakt.



Figuur 1: Grafiek van de dataset

De dataset bestaat uit een verzameling van nodes die een bepaalde entiteit representeren. Deze nodes zijn onderling verbonden door relaties. De relaties representeren de relaties tussen de entiteiten. De nodes en relaties kunnen eigenschappen bevatten die de entiteiten en relaties beschrijven.


```
{
  "identity": 0,
  "labels": [
    "Node"
  ],
  "properties": {
    "externalid": "100",
    "type": "HGNC"
  },
  "elementId": "0"
}
```


Figuur 2: Voorbeeld van een node uit de dataset

Zoals te zien is in het bovenstaande voorbeeld, bestaat een node uit een identiteit, een verzameling labels en een verzameling eigenschappen. De identiteit is een unieke identificatie van de node binnen de dataset. De labels zijn een verzameling van labels die de node categoriseren. De eigenschappen zijn een verzameling van eigenschappen die de node beschrijven.

3.3 Meetmethode

Om de prestaties van een OGM en de standaard Neo4j-driver te vergelijken, is de tijdsduur gemeten die nodig is om alle nodes uit de dataset op te halen. De tijdsduur is gemeten met behulp van de PHP-functie `microtime()`. Deze functie geeft de huidige tijd in seconden en microseconden terug. Door het verschil te nemen tussen de tijd voor en na het uitvoeren van de taak, kan de tijdsduur worden bepaald.

```php
public function saveQueryMeasurements(Neo4jClient $client): void
{
    $measurements = [];
    for ($i = 0; $i < 100; $i++) {
        $startTime = microtime(true);

        // Haal alle nodes uit de Neo4j-database,
        // via de standaard Neo4j-driver
        $response = $client->getAllEntities();

        $endTime = microtime(true);
        $executionTime = $endTime - $startTime;

        // Voeg de executionTime toe aan de lijst met metingen.
        // Vermenigvuldig de executionTime met 1000 om de tijd
        // in milliseconden te krijgen.
        $measurements[] = $executionTime * 1000;
    }

    // Sla de metingen op in een JSON-bestand
    $jsonMeasurements = json_encode($measurements);
    file_put_contents('measurements.json', $jsonMeasurements);
}
```

Figuur 3: Voorbeeld van het meten van de tijdsduur, geschreven in PHP.

In het bovenstaande voorbeeld is te zien hoe de tijdsduur is gemeten. De metingen zijn uitgevoerd op een lokale ontwikkelomgeving. De metingen zijn uitgevoerd in een docker container met de volgende specificaties:

```
root@3d18886dc705
-----------------
OS: Debian GNU/Linux 12 (bookworm) x86_64
Host: B450M DS3H
Kernel: 6.4.2-zen1-1-zen
Uptime: 3 hours, 42 mins
Packages: 308 (dpkg)
Shell: bash 5.2.15
Resolution: 2560x1440
CPU: AMD Ryzen 5 3600 (12) @ 3.600GHz
GPU: AMD ATI Radeon RX 6600/6600 XT/6600M
Memory: 10688MiB / 15912MiB
```

Figuur 4: Specificaties van de docker container waarin de metingen zijn uitgevoerd.

De resultaten van de metingen zijn opgeslagen in een JSON-bestand. Met deze resultaten kunnen de prestaties van een OGM en de standaard Neo4j-driver worden vergeleken.

3.4 Analysemethode

De verzamelde metingen worden geanalyseerd om inzicht te krijgen in de prestatieverschillen tussen het gebruik van een OGM en de standaard Neo4j-driver. Hierbij worden statistsche technieken toegepast om de resultaten te vergelijken en te interpreteren.

Voor de analyse van de resultaten is een script[1] ontwikkeld. Dit script is geschreven in Python en maakt gebruik van matplotlib, een Python-bibliotheek voor het maken van grafieken. Het script leest de metingen uit het JSON-bestand en maakt een grafiek van de metingen. De grafiek wordt opgeslagen als een PNG-bestand. Ook maakt het script gebruik van de Python-bibliotheek statistics om de standaarddeviatie en variantie van de metingen te berekenen.

Een tweede script[2] is ontwikkeld voor het creeëren van tabellen met de beschrijvende statistieken van de metingen. Dit script is geschreven in Python en maakt gebruik van tabulate, een Python-bibliotheek voor het maken van tabellen. Het script leest de metingen uit het JSON-bestand en berekent de beschrijvende statistieken van de metingen. De statistieken worden weergegeven in een tabel. De tabel wordt opgeslagen als een PNG-bestand.

Het script voert de volgende stappen uit om de metingen te verwerken en de resultaten te genereren:

1. Inlezen van de gegevens: De metingen worden ingelezen uit het JSON-bestand.
2. Berekenen van de aspecten van de metingen: De gemiddelde, minimale en maximale tijdsduur worden berekend. Ook worden de standaarddeviatie en variantie berekend.
3. Berekenen van het verschil in tijdsduur: Het verschil in tijdsduur tussen het gebruik van een OGM en de standaard Neo4j-driver wordt berekend.
4. Genereren van een visuele plot: De metingen worden geplot in een grafiek. De grafiek wordt opgeslagen als een PNG-bestand.

Door het gebruik van dit script kunnen de verzamelde gegevens op een gestructureerde en efficiënte manier worden geanalyseerd. Het script biedt de mogelijkheid om inzicht te krijgen in verschillende aspecten van de metingen, zoals de gemiddelde tijdsduur, standaarddeviatie en het verschil in tijdsduur. Dit inzicht kan worden gebruikt om de resultaten te interpreteren en conclusies te trekken.

3.5 Validatie

Om de validiteit van de resultaten te waarborgen, is gebruikgemaakt van een aantal validatietechnieken. Deze technieken zijn toegepast om de betrouwbaarheid van de resultaten te verhogen. De validatietechnieken die zijn toegepast, worden in dit hoofdstuk beschreven.

- **Betrouwbaarheid van de metingen:** Om de betrouwbaarheid van de metingen te verhogen, zijn de metingen 100 keer uitgevoerd. De metingen zijn uitgevoerd op een lokale ontwikkelomgeving, in een docker container. De metingen zijn uitgevoerd op een rustig moment, om de invloed van andere processen te minimaliseren.
- **Steekproefgrootte en representativiteit:** De steekproefgrootte is een dataset van 2349 nodes. Dit is de dataset die beschikbaar is gesteld door de opdrachtgever van het project waar dit

[1] Script is te vinden in ./bijlagen/onderszoeksrapport/main.py
[2] Script is te vinden in ./bijlagen/onderzoeksrapport/tables.py

onderzoek onderdeel van uitmaakt. De grootte van de dataset is representatief voor de dataset die gebruikt wordt in het project.

- **Interne validiteit:** De standaard Neo4j-driver dient als controlegroep voor de metingen. De metingen met de standaard Neo4j-driver worden gebruikt als referentiepunt voor de metingen met een OGM. De metingen met de standaard Neo4j-driver worden uitgevoerd op dezelfde dataset als de metingen met een OGM. Hierdoor is de interne validiteit van de resultaten gewaarborgd.
- **Kritische reflectie:** Een beperking van dit onderzoek is dat grootte van de dataset relatief klein is. De dataset bestaat uit 2349 nodes. Dit is een relatief kleine dataset. De resultaten van dit onderzoek zijn mogelijk niet representatief voor grotere datasets. Een aanbeveling voor vervolgonderzoek is om de resultaten te valideren met een grotere dataset. Ook is er geen variatie in de grootte van de dataset. De grootte van de dataset is constant. Een aanbeveling voor vervolgonderzoek is om de resultaten te valideren met datasets van verschillende groottes.

4 Resultaten

Dit hoofdstuk presenteert de resultaten van het onderzoek naar de impact van de OGM op de prestaties van het ophalen van alle nodes uit de database. De resultaten worden gepresenteerd aan de hand van de analyses van de verzamelde metingen.

4.1 Beschrijvende statistieken

De beschrijvende statistieken van de metingen zijn berekend met behulp van het script dat is beschreven in het hoofdstuk Methodiek. Er wordt gekeken naar de gemiddelde, minimale en maximale tijdsduur van de metingen. Ook wordt er gekeken naar de standaarddeviatie en variantie van de metingen.

De gemiddelde, minimale en maximale tijdsduur bieden inzicht in de prestaties van zowel de OGM als de standaard Neo4j-driver. Met deze statistieken kunnen er vergelijkingen worden gesteld tussen de prestaties van beide benaderingen. Daarnaast geven de gemiddelde, minimale en maximale tijdsduur ook inzicht in de variabiliteit van de metingen, terwijl de standaarddeviatie en variantie informatie verschaffen over de mate van spreiding van de metingen.

4.2 Vergelijking van de prestaties

Dataset	Gemiddelde	Standaarddeviatie	Variantie	Minimum	Maximum
Met OGM	52.1171	13.789	190.138	47.3928	187.796
Zonder OGM	39.065	2.2187	4.92264	35.6419	53.942

Tabel 1: Vergelijking van de prestaties van een OGM en de standaard Neo4j-driver.

Dit tabel toont de beschrijvende statistieken van de metingen. Alle statistieken zijn in milliseconden. Dit tabel[3] is gebaseerd op het tabel dat is gegenereerd door het script dat is beschreven in het hoofdstuk Methodiek.

Er kunnen enkele observaties worden gemaakt op basis van dit tabel:

1. **Gemiddelde tijdsduur:** De gemiddelde tijdsduur voor het ophalen van alle nodes uit de database met behulp van de OGM is 52.1171 milliseconden, terwijl de gemiddelde tijdsduur zonder het gebruik van de OGM 39.065 milliseconden bedraagt. Hieruit valt mogelijk te concluderen dat het gebruik van de OGM een negatieve invloed heeft op de prestaties van het ophalen van alle nodes uit de database.
2. **Variabiliteit van metingen:** De standaarddeviatie van de metingen met de OGM is 13.789 milliseconden, terwijl deze slechts 2.2187 milliseconden is zonder de OGM. Dit wijst op een grotere spreiding in de metingen met de OGM, wat betekent dat de prestaites van de OGM mogelijk meer variëren dan de prestaties zonder de OGM.
3. **Spreiding van metingen:** De minimale en maximale tijdsduur geven inzicht in het bereik van de metingen. Met de OGM varieert de tijdsduur tussen de 47.3928 milliseconden en 187.769 milliseconden, terwijl de tijdsduur zonder de OGM varieert tussen de 35.6419 milliseconden en 53.942 milliseconden. Ook dit geeft mogelijk aan dat de tijdsduur van het ophalen van alle nodes meer varieert met de OGM.

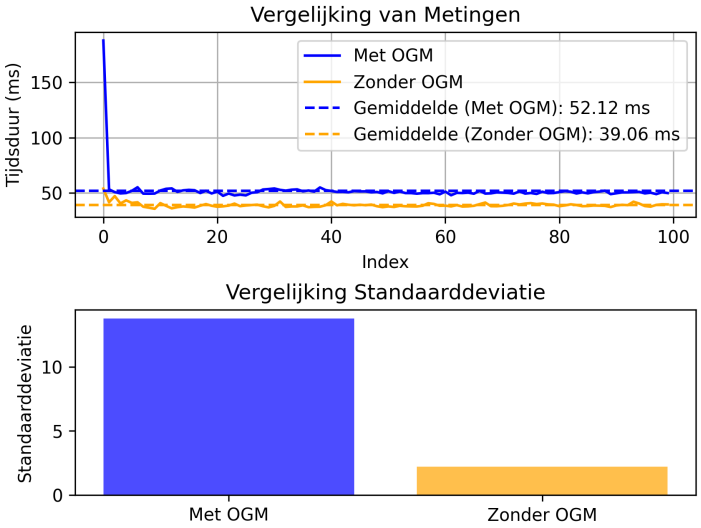
Deze observaties suggereren dat het gebruik van een OGM de prestaties van het ophalen van alle nodes uit de database kan beïnvloeden. De prestaties van de OGM lijken meer te variëren dan de prestaties zonder de OGM.

[3] Oorspronkelijke generatie van dit tabel is te vinden in ./bijlagen/onderzoeksrapport/tabel.txt

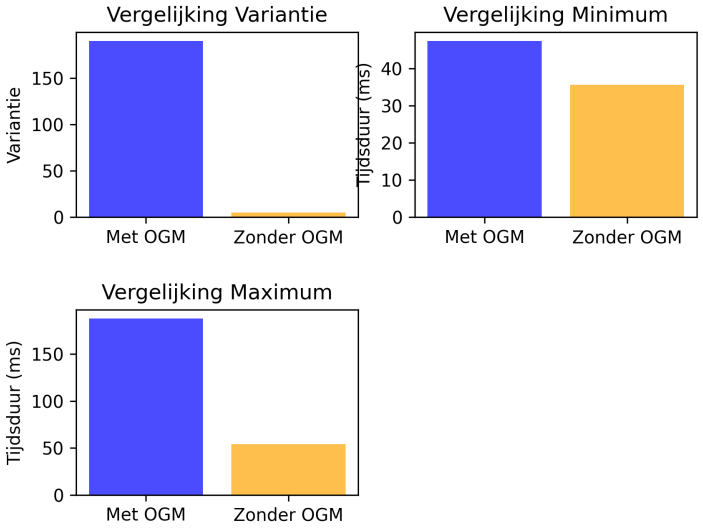
4.3 Visuele weergave

De resultaten van de metingen worden gepresenteerd in een grafiek die de tijdsduur van de metingen weergeeft. De grafiek toont twee kleuren: één voor de metingen met OGM en één voor de metingen zonder OGM. De grafiek illustreert de verschillen in tijdsduur tussen de twee benaderingen en biedt een visuele weergave van de prestaties. Daarnaast kunnen individuele datapunten worden weergegeven om de variabiliteit en spreiding van de metingen te tonen.

Tenslotte zijn er subplots toegevoegd. Deze subplots dienen als extra visuele weergave van de prestaties.



Figuur 5: Grafiek van de metingen 1



Figuur 6: Grafiek van de metingen 2

Zoals voorheen al is geobserveerd, lijkt het gebruik van de OGM een negatieve invloed te hebben op de prestaties van het ophalen van alle nodes uit de database. De grafiek toont aan dat de tijdsduur van de metingen met de OGM hoger is dan de tijdsduur van de metingen zonder de OGM. Ook is te zien dat de tijdsduur van de metingen met de OGM meer varieert dan de tijdsduur van de metingen zonder de OGM.

4.4 Discussie

In deze sectie worden de resultaten van het onderzoek besproken en geïnterpreteerd. Het doel is om inzicht te bieden in de bevindingen en de implicaties ervan te analyseren. De resultaten worden geanalyseerd aan de hand van de onderzoeksvraag en de doelstellingen van het onderzoek.

4.4.1 Prestatieverschillen tussen de OGM en de standaard Neo4j-driver

Uit de resultaten blijkt dat de gemiddelde tijdsduur voor het ophalen van alle nodes met OGM (52.1171 ms) hoger is dan zonder OGM (39.065 ms). Dit suggereert dat het gebruik van OGM de prestaties kan beïnvloeden en leiden tot langere verwerkingstijden. Bovendien vertoont de OGM-gebaseerde benadering een grotere standaarddeviatie (13.789 ms) in vergelijking met de standaard Neo4j-driver (2.2187 ms), wat wijst op een hogere mate van variabiliteit in de metingen met de OGM.

4.4.2 Mogelijke verklaringen

Er zijn verschillende factoren die kunnen bijdragen aan de waargenomen prestatieverschillen. Ten eerste voegt de OGM een extra laag toe aan de interactie tussen het objectgeoriënteerde model en de grafenstructuur van neo4j. Deze extra complexiteit kan leiden tot vertragingen en hogere

verwerkingstijden. Ten tweede kan de implementatie van de OGM en de configuratie ervan invloed hebben op deze prestaties. Het is mogelijk dat bepaalde instellingen of optimalisaties de efficiëntie van de OGM beïnvloeden.

4.4.3 Praktische implicaties

Op basis van de bevindingen kunnen praktische implicaties worden afgeleid. Indien de snelheid van het ophalen van alle nodes een cruciale factor is in het systeem, kan overwogen worden om de standaard Neo4j-driver te gebruiken in plaats van de OGM implementatie. Dit kan de prestaties van het systeem verbeteren en de efficiëntie van het ophalen van alle nodes verhogen.

Het is echter belangrijk om op te merken dat de keuze afhankelijk is van de context van het systeem. Andere aspecten, zoals complexiteit van het model en de behoefte aan grafenmanipulaties, kunnen ook een rol spelen bij de keuze tussen de OGM en de standaard Neo4j-driver.

Tenslotte is het ook belangrijk om de voordelen van de OGM te benadrukken. De OGM biedt een objectgeoriënteerde benadering van de grafenstructuur van Neo4j. Dit kan de ontwikkeling van het systeem vereenvoudigen en de complexiteit van de interactie met de Neo4j-database verminderen.

4.4.4 Beperkingen en aanbevelingen

Bij het interpreteren van de resultaten is het belangrijk om rekening te houden met enkele beperkingen van dit onderzoek. Ten eerste is de dataset beperkt tot een specifieke grootte. Het is mogelijk dat de prestatieverschillen tussen OGM en de standaard Neo4j-driver variëren bij gebruik van grotere datasets. Daarnaast is de focus van dit onderzoek beperkt tot het ophalen van alle nodes. Het is mogelijk dat de prestatieverschillen in andere contexten of bij het uitvoeren van verschillende soorten operaties op de database anders zijn.

Voor verder onderzoek wordt aanbevolen om de prestatieverschillen tussen OGM en de standaard Neo4j-driver verder te onderzoeken in verschillende scenario's en met diverse datasets. Daarnaast kan het interessant zijn om te kijken naar optimalisaties en configuratie-instellingen die de prestaties van OGM kunnen verbeteren.

5 Conclusie

Dit onderzoek richtte zich op het vergelijken van de prestaties tussen het gebruik van een Object Graph Mapper (OGM) en de standaard Neo4j-driver bij het ophalen van alle nodes uit een Neo4j-database. De resultaten van het onderzoek bieden inzicht in de impact van de OGM-functionaliteit op de verwerkingstijden en variabiliteit van de metingen.

Uit de bevindingen blijkt dat het gebruik van OGM resulteert in een gemiddelde tijdsduur van 52.1171 ms, terwijl de standaard Neo4j-driver een gemiddelde tijdsduur van 39.065 ms vertoont. Dit duidt op een prestatieverschil tussen beide benaderingen, waarbij de OGM-gebaseerde benadering over het algemeen langere verwerkingstijden laat zien. Bovendien blijkt uit de analyse van de standaarddeviatie en variantie dat de metingen met OGM een hogere mate van variabiliteit vertonen in vergelijking met de standaard Neo4j-driver.

Deze bevindingen hebben praktische implicaties voor het ontwerp en de implementatie van systemen die gebruikmaken van een Neo4j-database. Indien de snelheid van het ophalen van alle nodes een cruciale factor is in het systeem, kan overwogen worden om de standaard Neo4j-driver te verkiezen boven de OGM-functionaliteit. Door de standaard driver te gebruiken, kunnen kortere verwerkingstijden worden gerealiseerd en kan een meer voorspelbaar gedrag worden verkregen.

Het is echter belangrijk om op te merken dat de keuze tussen OGM en de standaard Neo4j-driver afhankelijk is van de specifieke vereisten en context van het systeem. Andere aspecten, zoals de complexiteit van het objectgeoriënteerde model en de behoefte aan geavanceerde grafenmanipulaties, kunnen doorslaggevend zijn bij de keuze voor OGM.

Hoewel dit onderzoek inzicht heeft geboden in de prestatieverschillen tussen OGM en de standaard Neo4j-driver, zijn er enkele beperkingen die moeten worden overwogen. De dataset is beperkt tot een specifieke omvang en kenmerken, en de focus ligt op het ophalen van alle nodes. Het is mogelijk dat de prestatieverschillen variëren bij gebruik van grotere datasets of andere soorten operaties op de database.

Voor toekomstig onderzoek wordt aanbevolen om de prestatieverschillen tussen OGM en de standaard Neo4j-driver verder te onderzoeken in verschillende scenario's en met diverse datasets. Daarnaast kunnen optimalisaties en configuratie-instellingen voor OGM worden onderzocht om de prestaties te verbeteren en de variabiliteit te verminderen.

Dit onderzoek draagt bij aan het begrip van de prestatie-implicaties van het gebruik van OGM bij het ophalen van alle nodes uit een Neo4j-database. De bevindingen kunnen worden toegepast bij het maken van weloverwogen keuzes tussen OGM en de standaard Neo4j-driver, afhankelijk van de specifieke vereisten van het systeem en de gewenste prestaties.

6 Literatuurlijst

7 Bronnen

Bijlagen

Hanzehogeschool Groningen logo	Hanzehogeschool Groningen	https://freebiesupply.com/logos/hanzehogeschool-groningen-logo/
Titelpagina figuur	DALL-E-2, OpenAI	bijlagen -> onderzoeksrapport -> OIG.jpg
Figuur 1	L. J. J. Leuwol	bijlagen -> onderzoeksrapport -> graph.png
Figuur 2	L. J. J. Leuwol	Neo4j Browser
Figuur 3	L. J. J. Leuwol	broncode -> backend -> src -> Controller -> QueryController.php.png
Figuur 4	L. J. J. Leuwol	Neo4j Container
Tabel 1	L. J. J. Leuwol	bijlagen -> onderzoeksrapport -> tabel.txt
Figuur 5	L. J. J. Leuwol	bijlagen -> onderzoeksrapport -> plot_1.png
Figuur 6	L. J. J. Leuwol	bijlagen -> onderzoeksrapport -> plot_2.png