Virtuelle Threads und strukturierte Nebenläufigkeitssteuerung in Java

Gernot Hofbauer

BACHELORARBEIT (EXPOSÉ)

eingereicht am

Fachhochschul-Bachelorstudiengang

Software Engineering

in Hagenberg

im Juni 2024

Betreuung (Vorschlag):

DI Johann Heinzelreiter

Inhaltsverzeichnis

Exposé

1 Einleitung

Seit einigen jahren Arbeitet Oracle an Möglichkeiten zur Verbesserung der Skalierbarkeit von Java-Anwendungen im Projekt „Loom“. Der Hauptansatz dieses Vorhabens ist die Einführung von virtuellen Threads, die von der JVM verwaltet werden. Diese Threads sind leichtgewichtiger als klassische Plattform Threads und können in größerer Anzahl erzeugt werden. In Kombination mit „Continuations“ [3], die es ermöglichen sollen, die Ausführung von Threads zu pausieren und fortzusetzen, können so bestimmte Anwendungen mit hoher Nebenläufigkeit zukünftig effizienter gestaltet werden. Diese Bachelorarbeit beleuchtet diese Technologien und stellt die Neuerungen dem bereits bekanntem gegenüber.

2 Theoretischer Hintergrund und Stand der Forschung

Die parallele Ausführung von Code bringt immer Schwierigkeiten mit sich. Eine davon ist die Bindung von Threads im Betriebssystem an die erstellten Threads auch wenn dieser nur wartet. Bei einer hohen Anzahl an Threads kann dies schnell zu einem Problem werden. Virtual Threads [2] sollen dieses Problem lösen, indem sie von der JVM verwaltet werden und nicht an Threads im Betriebssystem gebunden sind. Wartet ein Virtual Thread, wird dieser vom OS-Thread gelöst und dieser kann derzeit Code eines anderen Virtual Threads ausführen. Dies führt dazu dass bei einer sehr hohen Anzahl an Virtual Threads wesentlich weniger Threads im Betriebssystem erstellt werden müssen und somit die Anwendung vor allem im Server-Clients Bereich viel skalierbarer wird. Diese Technologie ist auch die Grundlage für „Structured Concurrency“ [5]. Diese führt StructuredTaskscopes ein, die die Ausführung von asynchronen Aufgaben erleichtern und verbessern sollen. Durch die Möglichkeit von den Basisklassen abzuleiten und einzelne Methoden zu überschreiben, können so stark individualiesierte Ausführungsdiesnte entstehen [4]. Damit sollen Entwickler ihren Code in kleinere, unabhängige Aufgaben oder Aufgabengruppen strukturieren können oder eine Verwendung ohne manuelle Erstellung und Verwaltung der einzelenen Threads ermöglichen. Die Technologien soll auch die Erstellung von asynchronen Methoden vereinfachen und die Lesbarkeit des Codes verbessern. Eine weiters geplante Neuerung sind ScopedValues [1], die es ermöglichen sollen, unveränderbare Werte in einem bestimmten Scope zu speichern und abzurufen. Dies gilt auch für alle Child-threads und auch Verschachtelungen von Scopes sind mög-

lich. Ziel dabei ist eine Steigerung der Robustheit und Verständlichkeit des Codes im Vergleich zu dem bereits bekannten ThreadLocal.

3 Forschungsfrage

Aus diesen Ansätzen ergibt sich die folgende Forschungsfrage für diese Bachelorarbeit:

> Wie und wozu können Virtual Threads und anderere Neuerungen in Projekt Loom verwendet werden und wie unterscheiden sie sich von den bisherigen Technologien?

4 Methodik

Um diese Frage zu beantworten, soll die Bachelorarbeit als eine Kombination von Literaturarbeit und praktischer bzw. prototypischer Umsetzung realisiert werden.

Zu Beginn sollen die grundlegenden Eigenschaften und Limitierungen des bestehenden Thread-Konzepts untersucht werden um die Motivationen für die Neuerungen zu verstehen. Danach soll ein grober Überblick über Projekt Loom und die damit verbundenen Technologien gegeben werden. Daraufhin sollen die Technologien im Detail erläutert und mit den bisherigen Technologien verglichen werden wobei der Schwerpunkt auf Virtual Threads liegt. Dabei soll auch gezeigt werden wie und in welchen Fällen diese Neuerungen in der Praxis angewendet werden können und welche neuen Möglichkeiten sich dadurch ergeben. Es soll auch verdeutlicht werden welche bestehende Probleme der parallelen Ausführung durch die neuen Technologien nicht gelöst werden können. Die Benchmarks sollen Laufzeit und Speicherverbrauch unter verschiedenen Umständen beinhalten. Aus diesen Ergebnissen sollen Schlüsse in welchen Fällen die neuen Technologien verwendet werden sollten und in welchen nicht.

5 Erwartete Ergebnisse

Als konkretes Ergebnis wird eine Sammlung Kleinerer Programme erstellt die die neuen Technologien in Projekt Loom demonstrieren. Es sollen verschiedene Anwendungsfälle jeweils mit den neuen Technologien und den bisherigen Technologien umgesetzt werden sodass die Unterschiede klar erkennbar sind. Zusätzlich werden die neuen Möglichkeiten erforscht und nach möglichen Problemen untersucht. Benchmarks mit verschiedenen Rahmenbediengungen sollen die Unterschiede in Laufzeit und Speicherverbrauch möglichst representativ zu zeigen. Die Implementierung eines großen zusammenhängenden Programms ist nicht geplant, da dies wenig zielführend wäre.

Quellenverzeichnis

Literatur

[1] Andrew Haley. *JEP 481 Scoped Values (Third Preview)*. URL: https://openjdk.org/jeps/481 (siehe S. 1).

[2] Oracle. *Java Core Libraries Developer Guide/Virtual Threads*. URL: https://docs.oracle.com/en/java/javase/21/core/virtual-threads.html#GUID-DC4306FC-D6C1-4BCC-AECE-48C32C1A8DAA (besucht am 17. 09. 2024) (siehe S. 1).

Medien

[3] Oracle. *Continuations - Under the Covers #JVMLS*. URL: https://youtu.be/6nRS6UiN7X0 (besucht am 26. 08. 2023) (siehe S. 1).

[4] Oracle. *Java Asynchronous Programming Full Tutorial with Loom and Structured Concurrency - JEP Café #13*. URL: https://youtu.be/2nOj8MKHvmw (besucht am 02. 08. 2022) (siehe S. 1).

[5] Oracle. *Project Loom - Structured Concurrency*. URL: https://youtu.be/0mXGfsy7_Qo (besucht am 12. 07. 2024) (siehe S. 1).