Module: Parallelprogrammierung
Aufgabe: 3. Assignment

Prüfer: Prof. Dr. Thomas Umland
Autor: Benjamin Daniel Scheer

Einführung

In den folgenden beiden Abschnitten soll geklärt werden welche Sicherheits- und Lebendigkeitseigenschaften unser implementierter Echo und unser implementierter Election Algorithmus hat und dass diese auch eingehalten wurden.

Aufgabe 3 c)

Sicherheits- und Lebendigkeitseigenschaften des Echo Algortihmus

Race Hazards verhindern

Zur Vermeidung von Race Hazards wurden die Methoden echo und wakeup als synchronized implementiert. Somit sind diese beiden Methoden Atomar. Race Hazards können nun nicht mehr dadurch entstehen, dass die Nachbarn einer Node sich beim Aufruf von dessen wakeup oder echo Methode gegenseitig Unterbrechen. Allerdings kann es immer noch zu RaceHazards kommen wenn die Aufgerufene Node in ihren private und protected Methoden auf die Variablen zugreift, die in wakeup und echo verändert werden. Aus diesen Grund wird auf die Variablen „messageCount“, „internalConnectionData“ und „wokeupBy“ außerhalb von echo und wakeup nur lesend zugegriffen. Auf die Variable „currentState“ wird in „SendWakeups“ sowie in wakeup schreibend zugegriffen. Allerdings wird auf „SendWakeups“ nur zugegriffen wenn die jeweilige Node wach ist und die currentState Variable wird nur dann in „wakeup“ verändert wenn die Node nicht wach ist. Somit ist ausgeschlossen dass es zu einem RaceHazard kommt.

Deadlocks Verhindern

In unserem Algorithmus sind nur die Methoden „echo“ und „wakeup“ Atomar. In diesen Methoden werden keine neuen Ressourcen angefordert. Somit wird diese Notwendige Bedingung für einen Deadlock nie erfüllt, weswegen ein Deadlock auch nie auftritt.

Lebendigkeit

Um zu gewährleisten, dass alle Nodes gleichzeitig anfangen können wird ein CountDownLatch benutzt, sodass die verschiedenen Nodes erst mit dem Algorithmus beginnen können, wenn jede Node bereit ist.

Damit Threads die nichts zu tun haben nicht anderen Threads die Rechenleistung stellen, setzt jeder Thread am Ende eines Schleifendurchlaufs 1 Millisekunde mittels sleep aus. In dieser Zeit besitzt der Thread benötigt der jeweilige Thread keine Rechenleistung, die nun ein anderer Thread bekommen kann.

Module: Parallelprogrammierung
Aufgabe: 3. Assignment

Prüfer: Prof. Dr. Thomas Umland
Autor: Benjamin Daniel Scheer

Aufgabe 3 e)

Sicherheits- und Lebendigkeitseigenschaften des Election Algortihmus

Race Hazards verhindern

Alle Variablen, auf die Mittels „echo“ und „wakeup“ zugegriffen werden kann, werden nur innerhalb von atomaren Methoden verändert. Außerdem werden die Methoden „SendEcho“ und „SendWakeups“ abgebrochen, wenn die jeweilige Node während eines Aufrufs von wait von einer anderen Welle übernommen wird. Dadurch kann es zu keiner Teilung kommen und somit auch zu keinen Race Hazards.

Deadlocks verhindern

In den meisten Methoden kommt es zu keiner inkrementellen Anforderung von Ressourcen. Allerdings geschieht dies in den Methoden „SendEcho“ und „SendWakeups“ aus diesem Grund arbeiten wir in diesen Methoden mit „wait“. Es wird also so lange mittels „wait“ gewartet bis man in den lock für die „echo“ oder die „wakeup“ Methode bekommt. Zusätzlich kommt es nach einer gewissen Wartezeit zu einem Timeout, wo dann die belegten Ressourcen endgültig zurückgegeben werden. Somit ist Sichergestellt dass es immer zu einer Temporären Rückgabe von Ressourcen kommt, weswegen ein Notwendiges Deadlock Kriterium nicht erfüllt ist.

Lebendigkeit

- Durch den CountDownLatch wird sichergestellt dass jede Node die faire Chance hat anzufangen
- Mittels sleep(1) wird sichergestellt dass nichts tuende Threads nicht die Ganze Rechenzeit verbrauchen.
- Alle Nodes werden am Ende durch Leader beendet. Der Ruft eine Rekursiven Methode auf die für den ShutDown der jeweiligen Nodes sorgt.