NP – Projekt – Meilenstein Omega

Fabian Hupperich, Martine Marx

Grundlegendes Prinzip:

Zu Beginn der Ausführung lesen wir die benötigten Informationen aus dem Problem aus. Damit initialisieren wir dann eine „Fahrtdienstleitung“. Sie verwaltet unsere eigenen Monitore für die Stationen und Gleise. Bei Ihr können die Züge anfragen um ein einzelnes Gleis/Parkplatz zu reservieren oder frei zu geben (! Die Fahrtdienstleitung ist keine Monitor mehr).

Danach erzeugen wir für jede Trainschedule aus dem Problem eine Instanz unserer eigenen Klasse „Zug“ (implements Runnable) und starten dann anschließend jeweils einen Thread mit einer der Zug-Instanzen als Runnable.

Nun führen die Züge den beschriebenen Algorithmus aus der Projektbeschreibung Schritt für Schritt (siehe unten) aus und melden bei der Fahrtdienstleitung an wenn sie terminiert sind.

Wir warten dann mit „thread.join“ bis alle Zug-Threads terminiert sind und überprüfen zum Schluss bei der Fahrtdienstleitung ob sich alle Threads regelkonform abgemeldet haben. Ist dem nicht so, also etwas ist schief gegangen, geben wir false zurück, ansonsten true. Kurz davor geben wir noch mit dem Recorder „done“ aus.

Zug-Algorithmus und Nebenläufiges Reservieren:

Die Züge benutzen die vorgegebene Funktion „map.route“ um sich eine Liste an Connections der kürzesten Route zu holen. Anschließend versuchen sie diese Strecke für sich zu Reservieren. Sie versuchen jetzt jedes Gleis nacheinander (!) für sich zu reservieren. Das heißt, jeder Zug-Thread hällt in der Fahrtdienstleitung nicht ein Lock bis die ganze Strecke reserviert ist. Gelingt es dem Zug, jedes Gleis (und Parkplätze) der Strecke zu reservieren dürfen sie fahren.

Wie garantieren wir, dass sich die Züge die Gleise nicht gegenseitig wegschnappen (und verhindern somit ein DeadLock):

Wir teilen jedem Gleis eine eindeutige ID zu. Dies erlaubt uns eine totale Ordnung auf den Gleisen festzulegen. Wenn ein Zug anfängt seine Strecke zu locken, so muss er mit dem Gleis beginnen dass die kleinste ID hat und dann in aufsteigender Ordnung weiter.

Gibt es nun auf den Routen zweier Züge ein oder mehrere Gleise die sich decken, so wird nur einer der beiden Züge das Gleis unter denen die sich decken mit der kleinsten Id zuerst bekommen und kann dann alle Gleise die sich überschneiden für sich gewinnen. Der andere Zug stellt dann auch zum frühestmöglichen Zeitpunkt fest dass ein Gleis schon belegt ist.

Stellt ein Zug bei uns fest, dass eines seiner Gleise schon reserviert ist, so gibt er mit der Funktion „reverseReservation“ alle Gleise und Stellplätze wieder frei die er während seinem gescheiterten Reservierungsprozess bereits für sich genommen hatte.

Somit schließen wir aus, dass ein Zug der das „Gleis-Duell“ mit einem ersten Zug gewonnen hat dann die Strecke für diesen und andere Züge weiterhin blockiert.

Somit kann es nie zu einem Deadlock kommen, da alle Gleise nach der totalen Ordnung reserviert werden müssen, und sollte der Fall eintreten, dass ein Zug ein Gleis oder Stellplatz nicht mehr für sich beanspruchen kann, gibt er seine eigenen beanspruchten Plätze wieder frei um nichts zu blockieren.

Sollte beim ersten Versuch die Strecke zu reservieren dies nicht gelingen, so gibt die tryReserve-Funktion eine Liste mit den Gleisen zurück die bereits belegt sind.

Diese wird dann im zweiten Teil des Algorithmus benutzt um eine neue Route zu finden mit avoid-Liste.

3. Teil des Algorithmus – Auf die Fahrtdienstleitung warten

Sollten die beiden oberen Schritte aus dem beschrieben Algorithmus scheitern, so kommt es zu dem Schritt wo wir uns einen nächsten freien Parkplatz reservieren. Dazu suchen wir uns immer das nächste anliegende Gleis aus der Route und versuchen wenn die angrenzende Location ein Bahnhof oder Knotenpunkt ist dort einen Platz zu reservieren.

Dies geht im Notfall so lange, bis wir am Zielbahnhof ankommen, dort wird immer eine Parkmöglichkeit frei sein.

Nun versuchen wir die Strecke zu reservieren, sollte dies wieder Scheitern, so holen wir uns aus der Fahrtdienstleitung ein dort bereits vorhandenes Lock und eine dazugehörige Condition. Wir gehen dann mit der Condition in .await(). Die Fahrtdienstleitung signalisiert sobald ein Gleis unlocked wurde dann mit signalAll() dass die Züge die darauf warten, sich dieses nun Reservieren können.

Sobald es gelungen ist zu einer Parkmöglichkeit zu fahren, fängt der Algorithmus wieder von vorne an.

Das Ganze geht so lange bis der Zug an seiner Destination angekommen ist, oder durch eine Exception der Thread beendet wird.