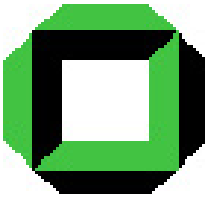

UNIVERSITÄT KARLSRUHE (TH)
Fakultät für Informatik
System Architecture Group
Frank Bellosa, Gerd Liefländer, Philipp Kupferschmied
Dominik Bruhn, Atanas Dimitrov,
Jonathan Dimond, Johannes Weiß

Basispraktikum Systemarchitektur - WS 2008/2009

Das Affenfelsenproblem

1 Thematik

Das Affenfelsenproblem ist eine spezielle Koordinationsaufgabe, welche eine intelligente Zusammenarbeit nebenläufiger bzw. paralleler Threads beansprucht. Gegeben seien zwei Affenfelsen, ein nördlicher und ein südlicher, die durch ein über eine tiefe Schlucht führendes Seil miteinander verbunden sind. Auf dem nördlichen Felsen gebe es einen großen Affenbrotbaum, auf dem südlichen gebe es eine Wasserstelle, so dass die Bewohner der Affenfelsen von Zeit zu Zeit gezwungen sind, den anderen Felsen über das Seil zu erreichen. Im Laufe der Evolution haben diese Affen ein erstaunliches Verhalten an den Tag gelegt, um möglichst schnell und bequem auf den jeweils anderen Felsen über das Seil zu gelangen. Sie haben erkannt, dass gleichzeitig bis zu k Affen auf dem Seil die Schlucht überqueren können, alle jedoch in der gleichen Richtung, da ein Ausweichen auf dem Seil zu riskant ist. Es gibt nun unter den Affen welche, die schneller durstig bzw. schneller hungrig werden als ihre Mitbewohner, während alle gleich schnell das Seil überklettern können.

Entwerfen Sie zwei verschiedene Protokolle, gemäß denen die Affen überleben könnten.

Frage 1: Gibt es Ausgangssituationen, die ein Überleben der Affen erschweren?

2 Grundlagen

Das Seil stellt offensichtlich einen kritischen Abschnitt dar, der nach Betreten in Δt Zeiteinheiten wieder verlassen wird. Gleichzeitig können bis zu k Threads in den kritischen Abschnitt, aber jeweils nur solche, die entweder alle nördliche oder alle südliche Affen modellieren.

Frage 2: Kennen Sie ein in der Betriebssystemliteratur sehr bekannt gewordenes Problem, das ebenfalls zwei verschiedene Sorten von Konkurrenten um gemeinsame Daten kennt?

2.1 Modellüberlegungen

Sie sollten initial auf beiden Affenfelsen einstellbar viele Affen platzieren, die jeweils auf den entsprechenden Felsen folgende Phasen durchlaufen:

- Essen bzw. Trinken (felsenabhängig)
- Miteinander Spielen (felsenunabhängig)
- Wird durstig bzw. hungrig, muss also das Seil überklettern

Es genügt zunächst mal, jeden Affen nur einmal die Schlucht zu überqueren. In der endgültigen Lösung, sollte jedoch das Affenleben unbegrenzt sein, d.h. ein ehemals nördlicher Affe wird zum südlichen, der auch wieder auf den nördliche Felsen will usw.

2.2 Java Monitore

Ein historisch relativ früh entwickeltes Sprachkonzept zur Zwangsserialisierung von kritischen Abschnitten sind Hoare's-Monitore. Ein Monitor entspricht einem Objekt, dessen Schnittstellenfunktionen (*member functions*) unter gegenseitigem Ausschluss stehen, d.h. so beschaffen sind, dass höchstens ein Thread eine dieser Monitorschnittstellenfunktionen ausführen kann. Mittels des Konzepts der synchronisierten Methoden (*synchronized*) kann man in Java ein Monitorobjekt modellieren. Ein Thread kann mittels der Funktion *wait()* an einer anonymen Bedingungsvariablen (*condition variable*) anhalten und das Monitorobjekt verlassen, bis durch ein *notify()* bzw. *notifyAll()* in einem anderen Thread der angehaltene Thread wieder fortsetzungfähig wird. Java garantiert nicht, dass der am längsten angehaltene Thread durch ein *notify()* bzw. *notifyAll()* aufgeweckt wird.

Eine typische Programmstelle innerhalb einer synchronisierten Methode eines Monitorobjekts sieht wie folgt aus:

while (condition) try {wait();} catch (Interruptet Exception e} {}

Man beachte, dass an obiger Programmstelle der aufrufende Thread angehalten wird, was gleichbedeutend mit einer Threadumschaltung ist.

3 Experimente

Lassen Sie sich was einfallen, wie sie diese Szene aus Affenfelsen, Schlucht, Seil, Wasser- bzw. den Affenbrotbaum und vor allem die Affen einigermaßen lebendig visualisieren können. Insbesondere das hungrig bzw. durstig werden, sollten Sie analog zur Visualisierung des Threadfortschritts in früheren Versuchen darzustellen. Sollten Ihnen Ihre Affen gar verhungern bzw. verdursten, sollte dies ebenfalls dargestellt werden. Sie können sich für das Visualisieren Zeit lassen, wichtiger ist zunächst mal die Lösung des Koordinationsaufgabenteils.

3.1 Einmaliges Überqueren der Schlucht

Platzieren Sie zunächst bis zu 30 Affen auf den nördlichen bzw. südlichen Felsen. Halten Sie dabei den Zeitpunkt fest, wann ein nördlicher Affe durstig bzw. ein südlicher Affe hungrig wird. Definieren Sie, wann ein Affe verhungert bzw. verdurstet sein soll, und eliminieren Sie diesen Affen aus Ihrer Modellierung.

3.2 Mehrmaliges Überqueren der Schlucht

Erweitern Sie Ihr obiges Testprogramm, so dass es am Beobachter liegt, wann er die Simulation beenden will.