Fachbereich Informatik
Wesley Terpstra
Dr.-Ing. Ilia Petrov
Prof. Alejandro Buchmann, Ph. D.
Sommersemester 2009

TECHNISCHE UNIVERSITÄT DARMSTADT

Grundlagen der Informatik II

4. Praktikumsaufgabe

Um alle geplante Missionen umsetzen zu können, musste die Dienststelle viele neue Agenten einstellen. Unglücklicherweise scheinen einige Doppelagenten durch die Sicherheitsscreenings geschlüpft zu sein. Bei diesen Doppelagenten kann es sich um Saboteure von fremden Mächten, Industriespionen oder Terroristen handeln. Sie gefährden nicht nur den Erfolg der Missionen an denen sie beteiligt sind, sondern auch die Vertraulichkeit unserer Arbeit. Sie müssen eliminiert werden.

Es konnten bereits einige Aktionen der Doppelagenten eingegrenzt werden. Da der Zeitrahmen dieser Aktionen festgestellt werden konnte, können Agenten entlastet werden, die im entsprechenden Zeitraum keinen Zugriff auf die kompromittierten Systeme hatten. Daher hat die Dienststelle die vollständige Überwachung aller Beschäftigten eingeleitet. Wann immer der Aufenthaltsort und die Tätigkeit eines Agenten bekannt sind, werden sie zentral protokolliert.

Sie werden eine Datenbank zusammenstellen, welche für jeden Agenten die Zeiträume enthält in denen für den Agenten eine unkritische Tätigkeit protokolliert werden konnte. Agenten werden solange als schuldig betrachtet bis ihre Unschuld nachgewiesen werden konnte. Ihre Aufgabe ist es die Agenten zu identifizieren, die aufgrund der Überwachungsprotokolle als unschuldig eingestuft werden können. Agenten ohne Alibi werden vom Sicherheitsdienst behandelt. Da die Aufdeckung von Sicherheitsverstößen kontinuierlich fortgesetzt wird, muss eine Abfrage der Datenbank jederzeit möglich sein. Ohne aktuellen Datenbestand könnten unschuldige Agenten einer unnötigen „Überprüfung“ ausgesetzt werden. Die Datenbank muss also Abfragen (bei der Entdeckung eines Vorfalls) und Einfügeoperationen (bei der Aktualisierung von Überwachungsprotokollen) abwechselnd zulassen.

Es ist Ihnen klar, dass eine passende Indexstruktur für eine effiziente Umsetzung nötig ist. Diese Datenstruktur speichert alle Intervalle in denen ein Agent ein Alibi hat. Es muss mit der Datenstruktur möglich sein, alle Agenten zu finden, die während des Zeitraums eines Vorfalls ein Alibi vorweisen können. Ihre Erfahrung sagt Ihnen, dass Sie einen AVL-Baum so erweitern können, dass er anstatt Schlüsselwerten überlappende Intervalle finden kann.

Zu implementieren: CounterIntelligence.java um unschuldige Agenten zu identifizieren!

Die Lösung ist bis zum **21. Juni 2009 23:59** einzureichen!

Achtung: Diese Anweisungen sind mit besonderer Sorgfalt zu behandeln und werden sich automatisch selbst zerstören!

Schritt 0: Laden Sie die Sourcecode-Vorgaben von `https://www.dvs.tu-darmstadt.de/teaching/inf2/2009-en/Problem4.zip` herunter.

Schritt 1: Implementieren Sie eine einfache Binbärbaum-Einfügeoperation in .add().

Hinweis: Verwenden Sie den Beginn des Intervalls als primären Sortierschlüssel und den Agentennamen als sekundären Sortierschlüssel.

Hinweis: Schreiben Sie die Funktion in rekursiver Weise um es später leichter zu haben.

Schritt 2a: Erweitern Sie add() um die Höhe zu aktualisieren.

Hinweis: Benutzen Sie die Hilfsfunktion updateHeight().

Schritt 2b: Implementieren Sie die AVL Links- und Rechtsdrehungen.

Hinweis: Step2bTest verwendet die add() Methode *nicht*. Nur rotateLeft() und rotateRight() werden überprüft. Sie können diese unabhängig implementieren.

Schritt 2c: Stellen Sie die Beibehaltung der Baumbalance sicher.

Hinweis: Verwenden Sie Links- und Rechtsdrehungen um den Baum zu rebalancieren, wenn balance() ≥ 2 oder $\leq$ -2 ist.

Schritt 3: Erweitern Sie die add()- und Rotationsfunktionen um biggest_end zu aktualisieren.

Hinweis: Eine Hilfsfunktion wie updateBiggest() könnte den Code vereinfachen.

Schritt 4: Implementieren Sie remove() um falsch eingetragene Agenten zu entfernen.

Hinweis: Sie werden dazu removeSmallest() benötigen.

Schritt 5: Implementieren Sie die Abfragefunktion um Agenten zu entlasten.

Hinweis: Verwenden Sie biggest_end um zu entscheiden, ob Sie in den linken Unterbaum absteigen müssen.

Zusätzliche Informationen zu AVL-Bäumen finden Sie unter:

a) `http://en.wikipedia.org/w/index.php?title=AVL_tree&oldid=292056402`
b) Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest. *Introduction to Algorithms*, 2nd edition. Chapter 13.5 (Problem 13-3).

Informationen über Intervallbäume und -suche finden Sie unter:

a) `http://en.wikipedia.org/w/index.php?title=Interval_tree&oldid=284625704`
b) Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest. *Introduction to Algorithms*, 2nd edition. Chapter 14.3.