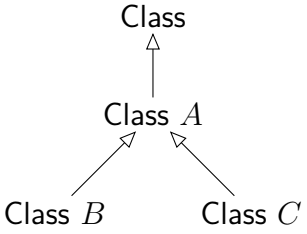
Universität Duisburg-Essen
Ingenieurwissenschaften / Informatik
Dozent: Prof. Dr. Barbara König
Übungsleitung:

WS 2018/19
6. Dezember 2018
Übungsblatt 5

Modellierung, Analyse, Verifikation (Programmanalyse)

Aufgabe 17 *Verifikation von Java-Bytecode*

Analysieren Sie folgendes Programm mit Hilfe des Worklist-Algorithmus. Verwenden Sie dazu nachfolgende Klassenhierarchie



und den initialen Wert $\iota = ([], [\mathsf{Class}\ B, \mathsf{Integer}, \mathit{Undef}, \ldots, \mathit{Undef}])$. Nehmen Sie außerdem an, dass die Klasse Class A eine Methode m mit einem Parameter vom Typ Integer und einem Rückgabewert vom Typ Integer besitzt. Die Methode des Bytecodes hat folgendes Aussehen und der Rückgabewert soll vom Typ Integer sein. Ist der Bytecode korrekt?

1: Load 0
2: Load 1
3: Invoke A m Integer Integer
4: Store 0
5: Load 0
6: IConst 10
7: CmpEq 1
8: Load 0
9: Return

Aufgabe 18 *Monotonie*

Zeigen Sie, dass die Transferfunktionen f_ℓ des Java Bytecode Verifier monoton sind. Tun Sie dies der Einfachheit halber nur an Hand des Invoke-Befehls. Überlegen Sie sich dazu, wann der Befehl ausgeführt werden kann und wie er sich auswirkt.

Aufgabe 19 *Neue Befehle für den Java Bytecode Verifier*

Im folgenden sind weitere Java Bytecode Befehle aufgeführt, die nicht in der Vorlesung definiert wurden. Definieren Sie die Typregel und die Transferfunktion für jeden der Befehle, so dass der Bytecode Verifier Code verifizieren kann, der diese Befehle enthält.

(a) Der Befehl Pop entfernt das oberste Element des Stacks.

(b) Der Befehl Dup erstellt eine Kopie des obersten Stackelements und legt dieses oben auf den Stack.

(c) Der Befehl Swap vertauscht die obersten beiden Elemente des Stacks.

(d) Der Befehl Checkcast *cname* kann angewandt werden, wenn das oberste Stackelement eine Referenz ist. Bei Anwendung wird der Typ des obersten Stackelements auf Class *cname* geändert. Die Korrektheit der Cast-Operation eines Objekts in die Klasse Class *cname* kann nur zur Laufzeit geprüft werden und wird daher vom Verifier nicht überprüft!

Aufgabe 20 Galois-Einbettung

Bei vielen praktischen Beispielen für Galois-Verbindungen stellt man fest, dass tatsächlich $\alpha(\gamma(m)) = m$ für alle $m \in M$ gilt. Das ist zumeist auch die natürlichere Definition, da es ansonsten mehrere Elemente in M geben kann, die das gleiche Element in L repräsentieren. Diese Aufgabe beschäftigt sich näher mit diesem Phänomen.

Seien $(L, \sqsubseteq)$ und $(M, \sqsubseteq)$ zwei vollständige Verbände. Ein Paar $\langle \alpha, \gamma \rangle$ von monotonen Funktionen $\alpha\colon L \to M$, $\gamma\colon M \to L$ heißt *Galois-Einbettung*, falls folgende Bedingungen erfüllt sind:

$$\forall l \in L\colon l \sqsubseteq \gamma(\alpha(l)) \tag{1}$$

$$\forall m \in M\colon \alpha(\gamma(m)) = m \tag{2}$$

Zeigen Sie, dass für eine Galois-Verbindung $\langle \alpha, \gamma \rangle$ mit $\alpha\colon L \to M$ und $\gamma\colon M \to L$ folgende Bedingungen äquivalent sind:

(a) $\langle \alpha, \gamma \rangle$ ist eine Galois-Einbettung.

(b) α ist surjektiv.

(c) γ ist injektiv.

(d) γ erfüllt: $\forall m_1, m_2 \in M\colon (\gamma(m_1) \sqsubseteq \gamma(m_2) \iff m_1 \sqsubseteq m_2)$.