Muli: Constraint-Programmierung in Java auf symbolischer JVM

Jan C. Dageförde and Herbert Kuchen

Westfälische Wilhelms-Universität Münster, D-48149 Münster
{dagefoerde,kuchen}@uni-muenster.de

Abstract. In der Praxis hat sich objektorientierte Programmierung mit Sprachen wie Java an vielen Stellen durchgesetzt, u. a. durch Features wie Kapselung von Struktur und Verhalten sowie Vererbung. Java ist allerdings nur bedingt geeignet zur Lösung von Suchproblemen, wie man sie beispielsweise bei der Personaleinsatzplanung oder der Produktionsplanung vorfindet.
Muli Lang (Münster Logic-Imperative Language) ergänzt die Programmiersprache Java um Möglichkeiten zur vereinfachten Lösung von Suchproblemen. Durch minimale sprachliche Ergänzungen führt Muli Lang logische Variablen und eingekapselte Suche ein.
Das zugehörige Laufzeitsystem *Muli Env* (Münster Logic-Imperative Environment) basiert auf einer symbolischen Java Virtual Machine (SJVM), welche Muli- und Java-Programme sowohl symbolisch als auch regulär zur Ausführung bringt. Das Laufzeitsystem wird außerdem um Komponenten wie Choice Points und Trail erweitert, welche von abstrakten Maschinen für logische Programmiersprachen, wie der Warren Abstract Machine (WAM), bekannt sind. Dadurch wird das Laufzeitsystem im Rahmen der eingekapselten Suche Backtracking-fähig, sodass auch nichtdeterministische Muli-Programme ausgeführt werden können.
Die enge Integration von Constraint-Programmierung und objektorientierter Programmierung macht Muli insbesondere für diejenigen Suchprobleme interessant, die aus Java-Anwendungen heraus berechnet werden und im Zeitverlauf inkrementell um Constraints ergänzt werden.

Keywords: Java Virtual Machine, Warren Abstract Machine, free variables, constraint solving, programming language integration