Karlsruher Institut für Technologie
Institut für Theoretische Informatik
Prof. Dr. Bernhard Beckert
Dr. Mattias Ulbrich, Mihai Herda

Formale Systeme, WS 2016/2017

Praxisaufgabe 1: Das Puzzle „Kuromasu“ mittels SAT-Solver lösen

Abgabe der Lösungen bis zum 08.01.2017 über das ILIAS-Portal zur Vorlesung

Hinweis: Bitte beachten Sie, dass die Praxisaufgabe in Einzelarbeit und selbständig zu bearbeiten ist. Wir behalten uns vor, die selbständige Bearbeitung dadurch stichprobenartig zu prüfen, dass wir uns die eingereichte Lösung erklären lassen.

Für die vollständige Lösung dieser Praxisaufgabe erhalten Sie 10 Punkte. Die erzielten Übungspunkte werden im Verhaältnis 1:10 als Bonuspunkte auf die bestandene Abschlussklausur angerechnet (max. ein Notenschritt Verbesserung).

Das Puzzle Kuromasu

Das Logik-Puzzle „Kuromasu“ ist eine NP-vollständige[1] Denksportaufgabe, die Sie in dieser Praxisaufgabe mit Hilfe eines Lösers für aussagenlogische Erfüllbarkeit (*engl.* satisfiability solver oder SAT solver) lösen sollen. Kuromasu wird auf einem rechteckigen Spielbrett gespielt. Manche Felder sind dabei mit Zahlen versehen (Zahlenfeld). Die Aufgabe ist nun, für ein Spielbrett eine Schwarz-Weiß-Einfärbung zu finden.

Im Einzelnen sind folgende Bedingungen von einer Einfärbung zu erfüllen:

1. Zwei schwarze Felder sind niemals horizontal oder vertikal direkt benachbart.
2. Alle Zahlenfelder sind weiß.
3. Für jedes Zahlenfeld ist die in ihm enthaltene Zahl gleich der Anzahl der von ihm aus sichtbaren weißen Felder. Ein Feld ist von einem anderen Feld aus sichtbar, wenn die Felder in der gleichen Zeile oder der gleichen Spalte sind und kein schwarzes Feld zwischen ihnen liegt. Dabei ist die Sichtbarkeits-Relation reflexiv; jedes Feld ist von sich selbst aus sichtbar.
4. Jedes weiße Feld ist von jedem anderen weißen Feld aus erreichbar. Dabei ist die Erreichbarkeits-Relation der transitive Abschluss der Sichtbarkeits-Relation.

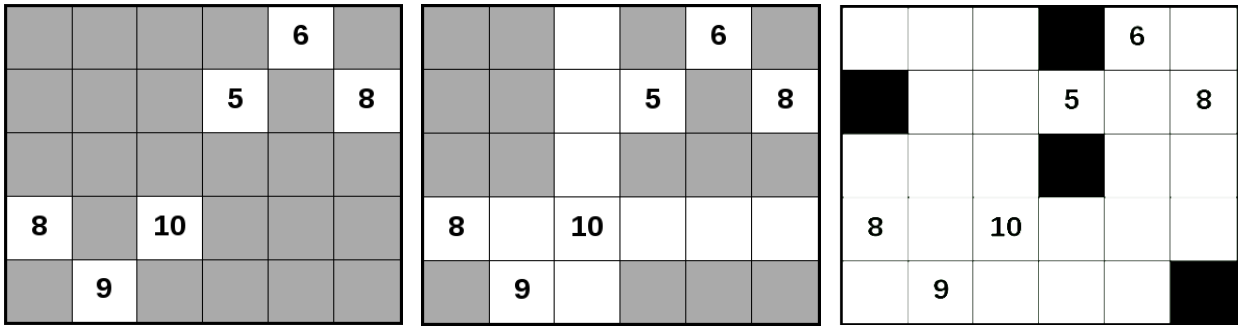
Weitere Informationen finden Sie auf Wikipedia (EN).

Die Aufgabe

Die Praxisaufgabe besteht darin, ein Java-Programm zu schreiben, das

1. Kuromasu-Spielbretter in aussagenlogische Klauselmengen transformiert, so dass jede erfüllende Belegung der Klauselmenge einer Lösung des Puzzles entspricht,
2. einen vorgegebenen SAT-Solver (SAT4J) aufruft, um eine solche erfüllende Belegung zu finden,
3. die gefundene Belegung in eine Lösung des Puzzles zurückübersetzt.

[1] Siehe dazu https://www.jstage.jst.go.jp/article/ipsjjip/20/3/20_694/_article



(a) Das ursprüngliche Spielbrett (b) Eine Zwischenlösung (c) Eine gültige Belegung von weißen und schwarzen Feldern

Abbildung 1: Ein Beispiel-Spielbrett der Größe 5×6

Die Bedingungen müssen dazu aussagenlogisch formalisiert werden. Sie müssen entscheiden, was Sie durch AL-Variablen modellieren und die Forderungen aus der Problemstellung in Klauseln formulieren. Achten Sie bitte darauf, dass Ihre Lösung nachvollziehbar und verständlich ist.

Hinweis: Die Erreichbarkeitsrelation für weiße Feldern kann man über eine Erreichbarkeit in k Schritten definieren: Ein Feld A ist von einem Feld B genau dann in k Schritten erreichbar, wenn es ein Feld C gibt, das von B in $k-1$ Schritten erreichbar und direkt benachbart zu A ist.

Bitte beachten Sie: Die Klauselmenge wird von dem Java-Programm für jede Instanz (jedes Spielbrett) jeweils neu erstellt. Sie kodiert sowohl die allgemeinen Regeln des Spiels als auch die Lage der Wände auf dem gegebenen Spielbrett.

Rahmenwerk

Um Ihnen die Bearbeitung der Aufgabe zu erleichtern, stellen wir Ihnen ein Rahmenwerk zur Verfügung, so dass Sie sich für die Bearbeitung auf die Aufgabe konzentrieren können.

Auf der Seite der Vorlesung finden Sie ein Archiv mit folgenden Dateien:

- `MyKuromasuSolver.java`: Ein Skelett, das Sie für Ihre Implementierung der Lösung erweitern sollen. Falls Sie weitere Klassen benötigen, speichern Sie diese bitte ebenfalls in dieser Datei.
  Beachten Sie:
  - Der Name der Klasse `MyKuromasuSolver`, sowie die Signatur der Methode `solve()` und der Konstruktor sind fest vorgegeben, um Ihre Abgabe automatisch überprüfen zu können.
- `kuromasu-1.0-deps.jar`: Eine Java-Bibliothek (inkl. SAT4J) um:
  - Spielbrettobjekte aus Dateien zu laden (`Kuromasu.load`),
  - den Status einzelner Felder abzufragen und zu setzen (Methoden `getField` und `setField` in der Klasse `Solution`),
  - sich Spielbretter grafisch anzeigen zu lassen (Methode `Solution.show()`),

Hinweis: Die vollständige Beschreibung der Funktionalität des Pakets finden Sie auf der Webseite der Vorlesung als JavaDoc-Dokumentation. Ihre Implementierung von `MyKuromasuSolver` muss von `edu.kit.iti.formal.Ku` erben. Dadurch erhalten Sie Zugriff auf folgende Membervariablen:

`game : Kuromasu` beinhaltet die Informationen zum aktuellen Spielbrett.

`solution : Solution` Ein initialisiertes Objekt zum Speichern Ihrer Lösung.

`solver : ISolver` eine Instanz des SAT-Solver (SAT4J). Eine Dokumentation des Interfaces findet sich in der SAT4J-Dokumentation

Probelauf

Nach dem Entpacken des bereitgestellten Archivs sollten Sie einen Ordner mit folgenden Dateien vorfinden:

- kuromasu-1.0-deps.jar
- riddles/
- Makefile
- MyKuromasuSolver.java

Testen Sie Ihre Systemkonfiguration, indem Sie die Implementierung von `MyKuromasuSolver` zu überetzen versuchen:

```
javac -cp kuromasu-1.0-deps.jar MyKuromasuSolver.java
```

Sie können einen Testlauf starten mit:

```
java -cp .:kuromasu-1.0-deps.jar edu.kit.iti.KuromasuTest
```

Dabei wird Ihre Implementierung auf alle Spielbretter aus dem Ordner `riddles/` angewendet. Sie sollten mehrere Ausgaben folgender Art sehen:

```
== sat.6x5.1110 ========================================================
XXXXXX
XXXXXX
XXXXXX
XXXXXX
XXXXXX
Riddle is satisfible
Your answer is UNKNOWN
```

Weiße Spielfelder erscheinen als „w“, schwarze als „b“ und Felder ohne Angabe als „X“. In dieser Beispielausabge gibt das System an, dass dieses Puzzle lösbar ist, aber die Implementierung dazu keine Angabe gemacht hat.

Tipp: Beginnen Sie Ihre Tests mit wenigen Spielbrettern im Ordner `riddles` und legen Sie nach und nach weitere Dateien dort hinein. Sollten Sie Problemfälle Ihrer Implementierung erkennen, erzeugen Sie sich für diese Fälle minimale Spielbretter, um diese Probleme isoliert zu lösen.

Kodierung der Spielbretter

Spielbretter werden in Dateien gepeichert und können über das bereitgestellte Framework eingelesen werden. Das Format ist zeilenbasiert. Am Anfang der Zeile steht ein Schlüsselwort „h“, „w“ oder „c“ gefolgt von einer positiven Zahl. Dabei gibt

$h\ \langle N \rangle$ die Höhe des Spielbrettes, sowie

$w\ \langle N \rangle$ die Breite des Spielbrettes an.

c ⟨*row*⟩ ⟨*column*⟩ ⟨*value*⟩ definiert ein Zahlenfeld in der entsprechenden Spalte, Reihe und Wertigkeit (Indexierung beginnt bei 0).

Das Spielbrett aus Abbildung 1 wäre wie folgt kodiert:

```
w 6
h 5
c 0 4 6
c 1 3 5
c 1 5 8
c 3 2 10
c 3 0 8
c 4 1 9
```

Im Archiv sind bereits einige Spielbretter mitgeliefert.

Der SAT-Solver SAT4J

Für diese Aufgabe verwenden wir das Werkzeug SAT4J[2]. SAT4J ist eine reine Java-Implementierung eines SAT-Solvers. SAT4J wird in vielen Java-basierten Systemen eingesetzt, u.a. im Alloy Analyzer[3] oder auch im Eclipse Framework[4].

Eine Klausel wird in SAT4J als Liste von Ganzzahlen repräsentiert, dabei entsprechen positive (negative) Zahlen positiven (negativen) Literalen. Beispielsweise entspricht die aussagenlogische Formel $(\neg P_1 \vee P_2) \wedge (P_1 \vee \neg P_3 \vee \neg P_2)$ etwa folgenden Aufrufen der Methode `addClause` eines `SATSolver`-Objekts `solver`:

```
solver.addClause(new VecInt(new int[]{-1, 2}));
solver.addClause(new VecInt(int[]{1, -3, -2}));
```

Die Erfüllbarkeit der so gesetzten Klauselmenge überprüft man mittels SAT-Solver durch Aufruf der Methode `findModel()` der Klasse `ISolver` – diese Methode liefert `null` zurück, falls die Klauselmenge unerfüllbar ist; gibt es eine erfüllende Interpretation, so wird ein Array von Ganzzahlen zurückgegeben – dabei werden diejenigen AL-Variablen, die als wahr interpretiert werden, durch eine positive ganze Zahl und diejenigen, die zu falsch ausgewertet werden, durch eine negative dargestellt. Das zurückgelieferte Modell ist partiell, nur AL-Variablen, für die der SAT-Solver eine Entscheidung getroffen hat, sind aufgeführt. Ist eine Variable nicht im Modell enthalten, ist diese beliebig belegbar.

Abgabe der Lösung

Bitte reichen Sie den Quelltext Ihres Java-Programms (*eine* Datei als ASCII-Text) bis spätestens 08.01.2017, 23:59 Uhr im **ILIAS-Portal** unter dem Punkt „Praxisblatt 1“ ein. **Auch für Programme, die nicht vollständig korrekte Ergebnisse liefern, werden Bonuspunkte anteilig vergeben.**

[2] http://www.sat4j.org/
[3] http://alloy.mit.edu/alloy4/
[4] http://mail-archive.ow2.org/sat4j-dev/2008-03/msg00001.html