Grundlagen der Künstlichen Intelligenz (CS 205)

Prof. Dr. M. Helmert
G. Röger
Frühjahrssemester 2012

Universität Basel
Fachbereich Informatik

Übungsblatt 8

Abgabe: 10. Mai 2013

Aufgabe 8.1 (3+3+1 Punkte)

In dieser Aufgabe soll die Verallgemeinerung des 8- und 15-Puzzles zum $m \times n$-Puzzle mit m Zeilen und n Spalten von dem domänenunabhängigen Planungssystem *Fast Downward* gelöst werden. Auf der Vorlesungsseite finden Sie ein entsprechendes Softwarepaket für Linux. Zur Ausführung von Fast Downward benötigen Sie Python in einer Version ≥ 2.6 (diese sollte in Ihrer Distribution standardmässig mitgeliefert sein).
In dem Paket finden Sie weiterhin die Modellierung einiger Probleminstanzen verschiedener Grösse. Die Instanz `prob-m-n-i.pddl` enthält dabei die (zufällig generierte) i-te Instanz für das $m \times n$-Puzzle.
Verwenden Sie zur Evaluation ein Zeitlimit von 5 Minuten und ein Speicherlimit von 2 GB (bzw. „physisches RAM minus 256 MB“, falls Ihr Rechner über weniger RAM verfügt). Zeit- und Speicherlimit können Sie sich z.B. unter Linux mit `ulimit -t 300` bzw. `ulimit -v 2097152` für die aktuelle Shell-Sitzung setzen.
Führen Sie nach jedem Aufruf von Fast Downward das `cleanup`-Skript aus. Wenn Sie alle Instanzen einer Grösse nicht mehr gelöst werden, dürfen Sie davon ausgehen, dass alle grösseren Instanzen (also mit mindestens genauso vielen Zeilen und Spalten) in dieser Versuchsreihe auch nicht mehr gelöst werden können.

(a) Lösen Sie die Instanzen zunächst mit optimaler Suche. Verwenden Sie hierzu Fast Downward mit A*-Suche und der (zulässigen) LM-cut-Heuristik. Führen Sie hierzu

```
./plan mxn-puzzle-pddl-benchmarks/prob-m-n-i.pddl --search "astar(lmcut())"
```

aus, wobei `m`, `n`, und `i` jeweils geeignet ersetzt werden muss. Geben Sie die Ergebnisse (Laufzeiten, Kosten der Pläne) an.

(b) Lösen Sie die Instanzen mit suboptimaler Suche. Verwenden Sie hierzu Fast Downward mit sogenannter *Anytime Suche*. Anytime Suche sucht innerhalb des gegebenen Zeitfensters iterativ nach besseren Plänen. Führen Sie hierzu

```
./plan mxn-puzzle-pddl-benchmarks/prob-m-n-i.pddl ipc seq-sat-lama-2011
```

aus und geben Sie jeweils die Kosten für den *letzten* (=besten) gefundenen Plan an. Die Ergebnisse für die verschiedenen gefundenen Pläne während eines Laufs der Anytime Suche werden in der entsprechend benannten Datei *plan_numbers_and_cost* gespeichert.

(c) Vergleichen und diskutieren Sie die Kosten für die gefundenen Pläne der optimalen Suche aus a) mit den Kosten der jeweils besten gefundenen Pläne der suboptimalen Suche aus b).

Bemerkung: Bei Interesse finden Sie weitere und detailliertere Hinweise zu Fast Downward auf der Webseite `http://www.fast-downward.org/`.

Hinweis: Sie dürfen gerne ein Skript schreiben, um die Experimente einfacher durchführen zu können.

Aufgabe 8.2 (1.5+1.5 Punkte)

Stellen Sie sich folgende Situation vor: Ein entfernter Verwandter stirbt in einem tragischen Autounfall. Wie sich herausstellt, sind Sie das einzige Familienmitglied, mit dem er nicht im Streit lag. Laut seinem Testament erben Sie daher sein ganzes Vermögen, das aus einem kleinen Haus mit zwei Räumen (*RaumA* und *RaumB*) und einigem Barvermögen besteht.
Auf so unverhoffte Weise reich geworden, beschliessen Sie, eine Party in *RaumB* zu geben. Leider ist der Raum mit einer Sammlung von Bällen vollgestopft, die Ihr Verwandter wohl gesammelt hat. Um Platz für die Party zu haben, müssen die Bälle in *RaumA* geräumt werden, wo ausreichend Platz ist. Sie finden in dem Haus einen Roboter, der wohl für den Transport von Bällen gebaut wurde. Da dieser leider defekt ist, werfen Sie ihn weg. Nun wissen Sie aber, dass Sie sich nicht selbst die Mühe machen müssen, die Bälle umzuräumen, und beschliessen, einen neuen solchen Roboter anzuschaffen.
Der Roboter wird mit Hilfe von PDDL kontrolliert, das Sie bereits aus der Universität kennen. Bei der Lieferung stellen Sie jedoch fest, dass die PDDL-Spezifikation nicht zu dem gelieferten Roboter passt. Der Roboter aus der Spezifikation hat nur zwei Arme, während Ihr neues Modell drei Arme besitzt. Laut Spezifikation sollen auch nur 4 Bälle transportiert werden, obwohl die Sammlung ihres Verwandten zehn Bälle umfasst. Als wäre das noch nicht genug, kann der Roboter aus der Spezifikation Bälle einzeln ablegen, während Ihr Modell nur alle Bälle, die er trägt, auf einmal ablegen kann.
Da es bereits Freitag Nachmittag ist und die Party Samstag Abend stattfinden soll, könen Sie die Firma nicht mehr rechtzeitig erreichen. Sie beschliessen daher, die PDDL-Spezifikation selbst anzupassen.

(a) Laden Sie die Dateien für die *gripper*-Planungsdomäne von der Vorlesungsseite herunter. Die Domänenbeschreibung finden Sie in `gripper.pddl`, die Problembeschreibungsdatei heisst `gripper-four.pddl`. Fügen Sie bei der Problembeschreibungs einen dritten Arm (*gripper*) namens `middle` hinzu sowie sechs zusätzliche Bälle namens `ball5` bis `ball10`. Alle Bälle sollen von `roomb` in `rooma` geräumt werden, wobei alle Arme des Roboters verwendet werden können. Lösen Sie das veränderte Problem mit Fast Downward (siehe Aufgabe 1) und geben Sie die neue Problembeschreibungsdatei und die Lösung ab.

(b) Ändern Sie die Domänenbeschreibung dahingehend, dass Sie die `drop`-Aktion durch eine neue Aktion `drop-all` ersetzen, die nur noch einen Raum `?room` als Parameter nimmt. Die neue Aktion soll alle Bälle ablegen, die der Roboter gerade trägt, vorausgesetzt der Roboter ist in Raum `?room`. Lösen Sie das neue Planungsproblem mit Fast Downward (wobei Sie die modifizierte Problembeschreibung aus der vorherigen Teilaufgabe verwenden) und geben Sie die neue Aktion und die Lösung von Fast Downward ab.

Hinweis: Sie werden einen bedingten Effekt und eine Quantifikation über zwei Variablen benötigen. Dies ist nicht mehr im STRIPS-Teilfragment von PDDL, sondern sie benötigen das `requirement :adl`. Quantifizierte Effekte haben die Form `(forall (?var1 ?var2 ... ?varn) <effect>)`. Der `<effect>` kann dabei ein einfacher oder konjunktiver Effekt sein, wie Sie sie aus der Vorlesung kennen, oder auch ein bedingter Effekt. Dieser hat die Form `(when <condition> <effect>)` und bedeutet, dass der Effekt nur ausgeführt wird, wenn zuvor die Bedingung wahr war.

Die Übungsblätter dürfen in Gruppen von zwei Studierenden bearbeitet werden. Bitte schreiben Sie beide Namen auf Ihre Lösung.