SE3

Aufgabenblatt 11

Softwareentwicklung 3: Logikprogrammierung - WS 2014/2015 - W. Menzel

Defaultschließen

Gesamtpunktzahl: 30 Abgabe der Lösungen bis zum 19.1.2015

Aufgabe 1: Defaults: Vorfahrtsregeln 14 Punkte
maximale Bearbeitungszeit: 80 Minuten

1. Implementieren und diskutieren Sie ein Regelsystem, mit dem Sie berechnen können, ob ein Verkehrsteilnehmer vorfahrsberechtigt ist. Achten Sie dabei auf eine möglichst redundanzfreie Modellierung und berücksichtigen Sie vor allem die folgenden (Default-)Regeln:

   – Fahrzeuge mit Sondersignal haben immer Vorfahrt.
   – Fahrzeuge im Kreisverkehr haben Vorfahrt.
   – Fahrzeuge auf der Hauptstraße haben Vorfahrt.
   – Fahrzeuge, die an einer gleichberechtigten Kreuzung von rechts kommen, haben Vorfahrt.

2. Modellieren Sie die folgenden Beobachtungen als Fakten und ermitteln Sie, ob den jeweiligen Fahrzeugen Vorfahrt zu gewähren ist.

   – Eine Feuerwehr fährt mit Sondersignal.
   – Ein grünes Auto fährt im Kreisverkehr und kommt von links.
   – Ein Fahrrad fährt im Kreisverkehr und kommt von rechts.
   – Ein Traktor kommt von links.
   – Ein Bus befindet sich auf der Hauptstraße und kommt von links.
   – Ein LKW kommt von rechts.
   – Eine Pferdekutsche fährt auf der Hauptstraße.
   – Ein Jeep kommt auf der Nebenstraße von rechts.

3. Erweitern Sie Ihre Modellierung so, dass auch eine Begründung mit ausgegeben wird, aufgrund welcher Fakten, bzw. Defaultschlüsse das Berechnungsergebnis zustande gekommen ist. Beachten Sie dabei insbesondere, dass in bestimmten Fällen mehrere Fakten bzw. Defaultregeln miteinander kombiniert werden müssen.

4. Da Prolog im negativen Fall keine Variablenbindungen ermitteln kann, benötigen wir ein komplementäres Prädikat, das berechnet, wann ein Verkehrsteilnehmer *keine* Vorfahrtsberechtigung besitzt. Definieren Sie dieses Prädikat und kombinieren Sie es mit der Definition aus Aufgabenteil 3 zu einem Prädikat, das in jedem Falle eine Begründung für seine Entscheidung bereitstellt.

Aufgabe 2: Rekursive Berechnungsvorschriften mit Memoisierung 16 Punkte
maximale Bearbeitungszeit: 60 Minuten

A rectangular chocolate bar consists of $m \times n$ small rectangles and you wish to break it into its constituent parts. At each step, you can only pick one piece and break it along any of its vertical or horizontal lines. How should you break the chocolate bar using the minimum number of steps (breaks)?

E.F. Meyer III, N. Falkner, R. Sooriamurthi, Z. Michalewicz (2014)
Guide to Teaching Puzzle-based Learning, Springer-Verlag, London.

1. Solve the puzzle the CS way: Define a predicate which selects the minimum number of steps among all alternative possibilities to break the chocolate bar into pieces.
2. Construct a structure on an additional argument position, which tells you where and how to break the bar and what to do with the resulting two pieces.
3. Modify the previous predicate that it keeps its partial results in the data base for later reuse instead of re-computation.
4. Compare the runtime behaviour of the two predicates for bars of different size and for repeated computations of the same size. To measure the duration of the execution use the predicate `time/1`. If necessary you can increase the size of the global stack by calling the Prolog system with the parameter -G, e. g. the call

   ```
   $ swipl -G1g
   ```

   sets the stack to a size of 1 GByte (instead of the 128 MByte default size).
5. Implement a solution for housekeeping, to remove all intermediate results from the data base.