Entwurfsbeschreibung des Vorprojekts

swp-helios

7. April 2014

Inhaltsverzeichnis

1 Allgemeines

Das Ziel des Vorprojekts war es, ein aussagekräftiges Benchmark für Planungsalgorithmen zu programmieren, um damit später unseren Algorithmus mit anderen wie z.B. HELIOS oder einem kanonischen Planer vergleichen zu können. Dazu war die Anforderung, verschiedene Linkspezifikationen mit unterschiedlichen Wissensdatenbanken und komplexen Verlinkungskriterien zu finden, um möglichst abwechslungsreiche Testbedingungen zu gewährleisten.

2 Produktübersicht

Das Produkt besteht aus zwei wesentlichen Bestandteilen: 10 verschiedenen LIMES-Konfigurationsdateien, die zum Benchmarken benötigt werden, und dem eigentlichen, auf LIMES aufbauenden Benchmarkprogramm. Die Konfigurationsdateien bestehen aus verschiedenen Definitionen mit unterschiedlichen zu verlinkenden Wissensdatenbanken. Die Ausgangsdatenbank ist meistens DBpedia, da sie sehr groß ist und weite Themenbereiche zum Verlinken bietet. Das Programm ermittelt dann die durchschnittliche Geschwindigkeit verschiedener Planungsalgorithmen bei der Abarbeitung der gegebenen Konfigurationsdateien. Dies wird dann in eine Ausgabedatei geschrieben. Der Ordner, der die Dateien enthält, und der Name der Ausgabedatei werden dabei vom Nutzer per Kommandozeilen-Parameter bestimmt.

3 Grundsätzliche Struktur-und Entwurfsprinzipien

Linkspecs, die zum Benchmarken benötigt werden, können in der LIMES-üblichen Form als XML-Dateien benutzt werden. Die vorgefertigten Linkspecs sind im Unterordner “Linkspecs” zu finden. Die Benchmark-Funktion an sich wurde in einem (aktuellen) LIMES build in der von uns neu erstellen Datei im Verzeichnis
`de/uni_leipzig/simba/execution/planner/PlannerBenchmark.java`
integriert. Im kompilierten Build wird der Benchmark wie folgt im Terminal aufgerufen: ‘‘`java -jar limes-0.7-SNAPSHOT.jar dir output [times]`’’, wobei mit ‘‘`dir`’’ der Pfad des Verzeichnisses mit den Konfigurationsdateien, mit ‘‘`output`’’ der Name der Ausgabedatei und mit ‘‘`times`’’ optional die Anzahl der Durchläufen (100, wenn nicht spezifiziert) angegeben wird.

4 Struktur-und Entwurfsprinzipien einzelner Pakete

4.1 Linkspecs erstellen:

Es werden die Beispiel-Konfigurationsdateien, die LIMES liefert, als Vorlage verwendet. Um große Datenmengen verarbeiten zu können und somit verwertbare Werte zu erhalten, bedienen wir uns der Vielzahl an Onlinedatenbanken im Internet. Die Konfigurationsdateien sind in XML geschrieben.

Template:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE LIMES SYSTEM "limes.dtd">
<LIMES>
    <PREFIX>
        <NAMESPACE></NAMESPACE>
        <LABEL></LABEL>
    </PREFIX>
    ...
    <PREFIX>
        <NAMESPACE></NAMESPACE>
        <LABEL></LABEL>
    </PREFIX>
    <SOURCE>
        <ID></ID>
        <ENDPOINT></ENDPOINT>
        <VAR></VAR>
        <PAGESIZE></PAGESIZE>
        <RESTRICTION></RESTRICTION>
        <PROPERTY></PROPERTY>
    </SOURCE>
    <TARGET>
        <ID></ID>
        <ENDPOINT></ENDPOINT>
        <VAR></VAR>
        <PAGESIZE></PAGESIZE>
        <RESTRICTION></RESTRICTION>
        <PROPERTY></PROPERTY>
    </TARGET>
    <METRIC></METRIC>
    <ACCEPTANCE>
        <THRESHOLD></THRESHOLD>
        <FILE></FILE>
```

```
        <RELATION></RELATION>
    </ACCEPTANCE>
    <REVIEW>
        <THRESHOLD></THRESHOLD>
        <FILE></FILE>
        <RELATION></RELATION>
    </REVIEW>
    <EXECUTION></EXECUTION>
    <OUTPUT></OUTPUT>
</LIMES>
```

(Für mehr Informationen bitte das LIMES-Handbuch konsultieren)

4.2 Code für das Benchmaking:

- Einlesen der LIMES-Konfigurationsdatei mit der Klasse `de.uni_leipzig.simba.io.ConfigReader`.
- Ausführen der Linkspec
  Mit verschiedenen Planungsalgorithmen werden Pläne gefunden und diese ausgeführt. Zum finden der Pläne wird der nach dem `ExecutionPlanner` Interface (Abbildung 1) implementierte entsprechende Planer (bisher HELIOS und canonical) benutzt. Dies wird standardgemäß 100-mal durchgeführt.
  - Messen der Zeit
    Die Zeit wird vor und nach jedem Planfinden und-ausführen gemessen und die Differenzen ermittelt. Nach der letzten Ausführung werden die durchschnittlichen Zeiten berechnet
- Ausgabe in Datei
  Die Durchschnittszeiten für das Planen und das Ausführen, sowie der benutzte Planungsalgorithmus und die verwendete Linkspec werden in Tabellenform als Plaintext in eine Datei geschrieben. Für die Formatierung der Tabelle wurde eine eigene Klasse `Table` (Abbildung 2) eingeführt.

5 Datenmodell

Fast alle Bestandteile des Benchmark-Builds gleichen dem des normalen LIMES-Programms. Spezifikationen werden wie üblich als XML-Dateien eingelesen. Eine zusätzliche java-Datei, die in den LIMES-Build integriert

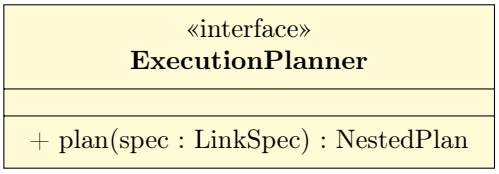

Abbildung 1: Das ExecutionPlanner Interface

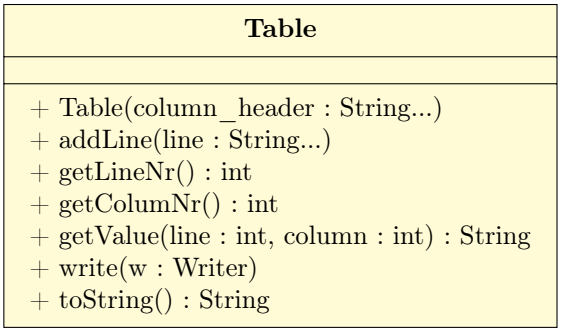

Abbildung 2: Die Klasse Table

ist, benutzt LIMES-Funktionalitäten zum Einlesen und Verwerten der Linkspecs und erstellt nach Fertigstellung der Berechnungen eine Datei, in die die Ergebnisse geschrieben werden.

6 Testkonzept

Parallel zur Software Entwicklung muss ein Testprozess ablaufen. Die Ergebnisse ermöglichen eine Beurteilung der Qualität der Software.

Der Planungsalgorithmus ist Hauptaufgabe des Projekts und sollte deshalb eine erhöhte Aufmerksamkeit im Testkonzept bekommen. Hierfür muss eine Testumgebung aufgebaut werden um den Algorithmus korrekt testen zu können. Das Vorprojekt bildet den Grundstein davon, indem es das Benchmark erstellt an dem sich der Algorithmus während der Entwicklung messen muss. Geplante Tests sind dabei:

1. Test mit dem gleichen Algorithmus, jedoch unterschiedlichen Parametern

2. Vergleich mit unterschiedlichen Algorithmen (den vorhandenen Planern canonical und HELIOS)

Die Anzahl der Tests sollte eine statistisch ausreichende Anzahl erreichen, um den Mittelwert über die Ergebnisse zu berechnen.

Der von uns entwickelte Algorithmus muss seine Tauglichkeit erst unter Beweis stellen. Deshalb ist es wichtig eine faire Testumgebung für die zu vergleichenden Algorithmen zu schaffen. Außerdem ist es wichtig die Tests mit verschiedenen Linkspezifikationen durchzuführen um einen Blick für die Stärken und eventuelle Schwächen des Algorithmus zu bekommen.

Dies verdeutlicht den Sinn des Vorprojekts. In unserem Fall werden es 10 komplexe Linkspezifikationen sein, welche ein aussagekräftiges Benchmark bilden sollen.

Die Software des Vorprojekts unterliegt natürlich auch dem Testkonzept und wie im Qualitätskonzept beschrieben gibt es verschiedene Teststufen, welche alle Entwicklungsstufen des Projekts abdecken sollen. Für das Vorprojekt ist hierfür wichtig:

7 Glossar

7.1 Benchmark

Ein Benchmark ist eine Vergleichende Analyse. In unserem Fall sollen Algorithmen verglichen werden. Dabei ist besonders die Performance entscheidender Bewertungsfaktor.

7.2 Linkspezifikation

Beschreibung der Mengen und deren Ähnlichkeitskriterien von zwei verschiedenen RDF-Ressourcen, die verknüpft werden sollen.

7.3 Planungsalgorithmus

Ein Planungsalgorithmus ist ein Algorithmus, der aus einem Abhängigkeitsgraph von Arbeitsschritten eine die Abhängigkeiten nicht verletzende lineare Abfolge der Arbeitsschritte erstellt.