Protokollbeschreibung zum vierten CAU-FLS-Programmierwettbewerb

21. Dezember 2017

Die Auswertung erfolgt automatisch durch ein Auswertungsprogramm. Die Programme aller Wettbewerbsteilnehmer werden mit denselben Instanzen getestet. Für jede Instanz wird ein neues Prozessabbild des Teilnehmerprogramms gestartet.

1 Schnittstelle zwischen Teilnehmer- und Auswertungsprogramm

Die Kommunikation zwischen Teilnehmerprogramm und Auswertungsprogramm erfolgt zeilenweise über die Standard-Ein und Ausgabestrme der Teilnehmerprogramme. Jede Ein- und Ausgabe wird mit einem Zeilenumbruch \n abgeschlossen.

Achtung: Je nach Umsetzung der Ausgabe kann es in manchen Programmiersprachen nötig sein, den Ausgabepuffer zu leeren (flush).

Achtung: Um die Kommunikation nicht zu stören, sollten Ausgaben (z.B. zu Debug-Zwecken) auf der Standardfehlerausgabe ausgegeben werden (fprintf(STDERR, ...) in C, cerr >> ... in C++, System.err.print(...) bzw. System.err.println(...) in Java, print >> sys.stderr, ... in Python).

Verstanden werden folgende Befehle:

INSTANCE Erfragt die Instanz. Antwort ist ein String, der Dateiname. Dieser enthält die Instanz in einem der in Abschnitt 4 beschriebenen Formate.

TIMELEFT Erfragt die verbleibende Zeit. Antwort ist ein einzelner Integer, die verbleibende Zeit in Mikrosekunden. Vor Beginn der Bearbeitungszeit ist dies die gesamte zur Verfügung stehende Zeit.

SOLUTION <<<< Beginnt die Ausgabe einer Lösung. Eine Lösung umfasst mehrere Zeilen. Sie wird beendet mit <<<< in einer einzelnen Zeile. Das Programm darf mehrfach Lösungen abgeben. Gewertet wird die letzte vollständige und zulässige Lösung.

<<<< Ende der Ausgabe einer Lösung. Antwort ist der String OK, oder der String INFEASIBLE, wenn die Lösung unzulässig oder nicht im richtigen Format ist.

Jeder andere Befehl wird mit dem String UNKNOWN COMMAND beantwortet. Es ist zu beachten, dass das Programm die Antwort auf Befehle immer vollständig (bis einschließlich zum Newline-Zeichen) lesen sollte, da es sonst zu einem Deadlock kommen kann, wenn der Puffer der Standardeingabe voll wird. Die Bearbeitungszeit beginnt ab der ersten Anfrage der Instanz, vorher kann das Programm initialisiert werden. Alle Befehle können jederzeit und beliebig oft verwendet werden. Das Programm des Teilnehmers wird nach Ablauf des Zeitlimits mit dem Signal term beendet.

2 Abgabeformat

Die Abgabe des Programms sollte als Archiv (.zip, .tar, .tar.gz oder .tar.bz2) erfolgen. Das Archiv wird vom Auswertungsprogramm automatisch in einen Ordner entpackt. Dieser erstellte Ordner ist zu Beginn der

Ausführung das Arbeitsverzeichnis. Das Archiv muss eine Datei info.cfg enthalten. In dieser ist anzugeben, wie das Programm aufgerufen wird. Optional kann ein Kommando zum Erstellen des Programms angegeben werden. Das Erstellen des Programmes zählt nicht zum Zeitlimit dazu. Beispiel für ein in Java geschriebenes Programm:

```
command = java MyProgram
make-command = javac MyProgram.java
```

Grundsätzlich wird die Wahl der Programmiersprache freigestellt. Sollte ein Teilnehmer eine Sprache verwenden wollen außer Java, C/C++ und Python wird er gebeten sich rechtzeitig zu melden, damit ggf. benötige Pakete installiert werden.

3 Testumgebung

Es wird auf der Webseite eine Testumgebung zur Verfügung gestellt. Diese enthält neben einigen Instanzen ein Auswertungsprogramm, geschrieben in Python 2.7, das die hier beschriebenen Protokolle einhält. Ein Beispielaufruf in der Testumgebung:

```
python test.py fls2018 test_config1 stupid.zip
```

Der erste Parameter (fls2018) ist die Problemstellung. Er sollte immer den Wert fls2018 haben. Der zweite Parameter (test_config1) ist Test-Szenario bzw. die Instanz. Die Szenarien befinden sich im Ordner problems/fls2018/testsuites. Der dritte Parameter (stupid.zip) ist das Archiv (oder der Ordner) des zu testenden Algorithmus.

4 Instanzformat

Beispiel für das Format, in dem die Instanzen übergeben werden:

```
NUM_EXCHANGE 3
NUM_DOCS 10
MAX_TRANSFER_TIME 24000
MAX_TIME 48000
DRIVING_TIMES
0 5903 4578 12419 14630 5664 13468 10784 10963 12177 9630 7564 14613 8463
5903 0 9113 8190 10494 2229 9239 7383 6205 7755 5401 12036 10388 4884
4578 9113 0 15400 15900 8900 16011 13992 14540 14500 12851 4135 16900 11914
12419 8190 15400 0 6399 7582 1049 5627 11132 7814 6067 17884 2826 12367
14630 10494 15900 6399 0 9793 6328 7481 14105 11872 8266 19400 7885 14615
5664 2229 8900 7582 9793 0 8631 6682 6919 7243 4731 11792 9718 6406
13468 9239 16011 1049 6328 8631 0 6676 10196 7856 7116 18933 1890 11979
10784 7383 13992 5627 7481 6682 6676 0 10986 8811 5098 16300 7916 11504
10963 6205 14540 11132 14105 6919 10196 10986 0 3865 6474 17300 9198 4137
12177 7755 14500 7814 11872 7243 7856 8811 3865 0 4788 17702 6929 5602
9630 5401 12851 6067 8266 4731 7116 5098 6474 4788 0 15300 8197 8358
7564 12036 4135 17884 19400 11792 18933 16300 17300 17702 15300 0 20292 14984
14613 10388 16900 2826 7885 9718 1890 7916 9198 6929 8197 20292 0 10979
8463 4884 11914 12367 14615 6406 11979 11504 4137 5602 8358 14984 10979 0
```

Die Bedeutung der Daten ist wie folgt.

NUM_EXCHANGE Anzahl der Orte, die allein für die Übergabe von Proben zwischen den Fahrzeugen zu verwenden sind.

NUM_DOCS Anzahl der Orte (Praxen), von denen Proben abgeholt werden müssen.

MAX_TRANSFER_TIME Längste erlaubte Zeit (in Minuten) zwischen Abholung einer Probe und Ablieferung im Labor.

MAX_TIME Zeitraum (in Minuten) in dem die Fahrten stattfinden. Nach dieser Zeit müssen alle Fahrzeuge zurück im Labor sein und dort bleiben.

DRIVING_TIMES Matrix $A[i, j]$ von Fahrtzeiten. Sie hat NUM_EXCHANGE + NUM_DOCS + 1 Zeilen und Spalten. $A[i, j]$ bezeichnet die benötigte Zeit um von Ort i nach Ort j zu fahren. Die Tabelle ist symmetrisch und erfüllt die Dreiecks-Ungleichung. Der erste Ort ist das Labor. Die nächsten NUM_EXCHANGE vielen Orte bilden die Übergabepunkte. Die übrigen Orte sind die Praxen.

5 Lösungsformat

Die Lösung wird übergeben als eine Aneinanderreihung von Touren. Jede Tour beginnt in dem Labor (Ort 0) und endet in selbigem. Sie beschreibt das Verhalten eines einzelnen Fahrzeugs. Eine Tourdefinition beginnt mit dem Wort 'tour'. Anschließend werden Aktionen ausgeführt. Es gibt die Aktionen 'move', 'load', 'unload'. Mit 'move i j t' wird das Fahrzeug angewiesen zum Zeitpunkt t (Minuten) von Ort i abzufahren mit Ziel j. Mit 'load t' wird das Fahrzeug angewiesen zum Zeitpunkt t Proben aufzunehmen. Dies geschieht entweder an einer Praxis oder an einem Übergabepunkt. Wenn es an einem Übergabepunkt geschieht, dann muss zur gleichen Zeit ein Fahrzeug aus einer anderen Tour Proben abgeben (mit 'unload'). 'unload t' definiert, dass das Fahrzeug zu Zeitpunkt t alle seine Proben abgibt. Dies geschieht entweder im Labor oder an einem Übergabepunkt bei dem ein anderes Fahrzeug gleichzeitig 'load t' ausführt. Zu einem Zeitpunkt t kann an jedem Übergabepunkt immer nur ein Fahrzeug 'load t' ausführen. Sonst ist die Lösung unzulässig. Der Zeitplan muss bezüglich der Fahrtzeiten einhaltbar sein. Ein Beispiel für eine Lösung:

```
SOLUTION <<<<
tour
move 0 10 0
load 100
move 10 1 100
load 300
move 42 0 300
unload 400
tour
move 0 8 0
load 50
move 8 9 50
load 70
move 9 1 70
unload 300
move 1 5 300
load 350
move 1 0 350
unload 500
...
<<<<
```

Hier übergibt das Fahrzeug aus der unteren Tour ihre Proben an das der oberen Tour an Übergabepunkt 1.