|  |  |  |
| --- | --- | --- |
| Beispielnr. | Runtimes Algorithmus C | Durchschnitt Algorithmus C |
| 1 CSP2 | **410250, 498223, 481508** | 463327 |
| 2 CSPB1 | **528721, 535759, 594115** | 552865 |
| 3 test | **517284, 487960, 514939** | 506727 |
| 4 Example1 | **982371, 897330, 971228** | 950309 |
| 5 | **786190, 705548, 671531** | 721089 |
| 6 | **1114331, 1171514, 1272977** | 1186274 |
| 7 | **1073277, 1152160, 1159785** | 1128407 |
| 8 | **1013455, 847772, 958325** | 9398850 |
| 9 | **1050697, 995274, 919030** | 988333 |
| 10 | **713465, 643380, 601739** | 652861 |

f)

Implementationsbeschreibung:

a)

Das CSP besteht aus zwei ArrayLists. Eine vom Typ Variable und die andere vom Typ Constraint. Zudem gibt es einen Stack vom Typ Variable für die zweite Hälfte der Intervalle.

Eine Variable besteht aus drei Integern. Die untere und obere Grenze der Domain und eine Position, zur Identifikation.

Ein SimpleConstraint besteht aus einer ArrayList von SimpleBounds. Jeder SimpleBounds besteht aus zwei Variablen x und y sowie zwei Integern, die Konstanten auf der linken beziehungsweise rechten Seite der Gleichung darstellen.

c)

Im Folgenden werden die Implementierungen einzelnen Schritte des Algorithmus A beschrieben.

Der erste Schritt liest den aktuellen SimpleBound aus. Die Bounds der Variablen und die Konstanten werden ausgelesen. Mit den Werten werden die beiden Gleichungen aus Definition 4 ausgewertet und entsprechende booleans gesetzt. Mit diesen Booleans wird daraufhin der nächste Schritt des Algorithmus entschieden und ausgeführt.

Der Zweite Schritt prüft, ob der Stack mit den zweiten Invervallhälften leer ist. Ist er leer terminiert der Algorithmus, sonst wird die Variable auf dem Stack genutzt.

Im Dritten Schritt wird zuerst eine Variable ausgewählt, deren Bounds verkleinert werden sollen. In unserem Falle wird die erste Mögliche Variable aus der Variablenliste genommen. Dies ist mehr oder wenig zufällig. Die Bounds der gewählten Variablen werden daraufhin halbiert. Die untere Hälfte wird als neue Bounds benutzt und die obere Hälfte wird auf den Stack gelegt.

d)

Algorithmus B wurde wie Algorithmus A implementiert. Es wurde jedoch der erste Schritt erweitert und eine Methode zur Deduction implementiert.

Der erste Schritt speichert nun für jeden Constraint, ob bisher nur ein Inconclusive SimpleBound gefunden wurde. Ist dies beim letzten SimpleBound noch der Fall, so wird die Deduction Methode aufgerufen. Die Deduction Methode berechnet neue Grenzen und prüft, ob diese das Intervall verkleinern. Verkleinernde Grenzen werden verwendet. Bei neuen Grenzen wird Schritt 1 ausführt, ansonsten Schritt 3.