**Artificial Intelligence**

**Report for Project 03: Golomb Ruler**

**Question 1: Plain Backtracking (BT) [45 points]**

**Analysis:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Backtracking** | | | | |
| **Test Cases** | | **Running Time (in sec)** | **Nodes Expanded**  **(For finding optimal length)** | **Critical Analysis** |
| **L** | **M** |
| **0** | **1** | **5.3882598** | **2** | Backtracking has a runtime complexity that is exponential in nature, as it expands all nodes. Therefore, max value of L and M for which the code executes within a few seconds are: **L=45, M=9 with time 12 secs.** |
| **4** | **3** | **9.8943710** | **13** |
| **18** | **6** | **0.0156798** | **809** |
| **25** | **7** | **0.1196920** | **7190** |
| **35** | **8** | **1.4845309** | **63814** |
| **45** | **9** | **11.928540** | **535135** |
|  |  |  |  |
|  |  |  |  |

**Analysis**: In backtracking we are assigning values in increasing order and if a ruler of Length L and M is founded, then we check for L-1 and order M ruler in order to find an optimal length ruler.

So, in backtracking nothing is checked except whether the assigned value is consistency or not.

**Question 2: BT + Forward Checking (FC) [45 points]**

**Analysis:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Backtracking with Forward Checking** | | | | |
| **Test Cases** | | **Running Time (in sec)** | **Nodes Expanded**  **(For finding optimal length)** | **Critical Analysis** |
| **L** | **M** |
|  |  |  |  | We do not proceed ahead with assignment of variables in case no more legal values are left on assigning a particular variable. This prunes some of the branches of the backtracking search tree and thus BT+FC reduces the runtime compared to BT. The max and L and M values for which the code executes within a few seconds are: **L=55, M=10 with time = 100 secs** |
| **0** | **1** | **1.8835067** | **2** |
| **4** | **3** | **9.5844268** | **11** |
| **18** | **6** | **0.0084350** | **418** |
| **25** | **7** | **0.0860779** | **4089** |
| **35** | **8** | **1.1445369** | **43028** |
| **45** | **9** | **8.9750049** | **341777** |
| **55** | **10** | **100.70552** | **3433284** |

**Analysis**: In backtracking with FC we are assigning values in increasing order and if a ruler of Length L and M is founded, then we check for L-1 and order M ruler in order to find an optimal length ruler.

So, in backtracking with FC we removed inconsistent values from domain of other variables and if no value remains in domain then we backtrack.

That is why the nodes expanded is lesser in FC then plain backtracking.

**Question 3: (BONUS) BT + Constraint Propagation (CP) [20 points]**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Backtracking with Constraint Propagation** | | | | |
| **Test Cases** | | **Running Time (in sec)** | **Nodes Expanded**  **(For finding optimal length)** | **Critical Analysis** |
| **L** | **M** |
|  |  |  |  | We do not proceed ahead with assignment of variables in case no more legal values are left on assigning a particular variable. This prunes some of the branches of the backtracking search tree and thus BT+FC reduces the runtime compared to BT. The max and L and M values for which the code executes within a few seconds are: **L=55, M=10 with time = 81 secs** |
| **0** | **1** | **1.4066696** | **2** |
| **4** | **3** | **0.0001411** | **11** |
| **18** | **6** | **0.0092501** | **286** |
| **25** | **7** | **0.0722851** | **2485** |
| **35** | **8** | **0.8290870** | **22853** |
| **45** | **9** | **8.0242061** | **167385** |
| **55** | **10** | **81.449174** | **1569839** |

**Analysis**: In backtracking with CP we are assigning values in increasing order and if a ruler of Length L and M is founded, then we check for L-1 and order M ruler in order to find an optimal length ruler.

So, in backtracking with CP we removed values from domain of other variables and if nothing is valid for domain of other variables by checking consistency then we backtrack.

That is why the nodes expanded is lesser in CP then plain backtracking and FC.