ME 617: Design Automation

Homework 2: Automated design of Gear trains

*Chirag Shah*

1. ***Representation***: In this gear train problem, each node in the search tree represents a gearbox except the first layer(as only one gear present in the first layer)i.e. for example if the program finds the goal in the 4th layer of the tree, the node at the 4th layer will have gear combinations of all layers from 1st to 4th. The node will have two important values: (1 )gearbox size(total number of gears present in the node and (2) the number of teeth of the last gear. The program loops and prints the value of the last gear teeth once it finds the correct combination meeting the goal and does it till 10 levels of the tree.
2. ***Evaluation***: The candidates are evaluated using the condition given .i.e. the difference of two mating teeth not be greater than 30. Based on this condition, the child nodes are created from each parent node. Also It is important to note that this condition is not true for all the layers. Since its not really useful to form a simple gear train(as speed will only be dependent on the 1st and the last gear) ,transformation pairs are formed between every subsequent mating gears. i.e the gear train formed will be like:

MPMPMPMPMPMP….… P=Pair

M=Mate

Here the condition for each pair is different from the mate. Each gear can be paired with all the other gears except itself while each gear can only be mated with gears having less than 30 teeth difference from itself.

Now after having created the children for the nodes, for the evaluation of the goal first output velocity wc is calculated. Here, it is important to make sure whether the output speed is positive or negative because for odd number of gears in the gear train the input and output gears will have same direction of rotation while for even number of gears the gears will have opposite direction of rotation. So, in the program sign of wc remains positive for odd gears and changes to negative for even gears.

Based on the value of wc, the error is calculated which is our goal.

The results of the code for the AI tree search methods are shown in the excel file attached.

1. Average branching factor(AFG):
   1. Wout = -117 RPM

AFG = 1.5

* 1. Wout = 77 RPM

AFG = 2.952468007312614

* 1. Wout = 377 RPM

AFG = 2.9264214046822743

* 1. Wout = -20 RPM

AFG = 2.9293172690763054

* 1. Wout = -2345 RPM

AFG = 2.90644111906311

* 1. Wout = 2 RPM RPM

AFG = 2.9060180923351284

1. Comparing the results of BFS and DFS, it can be seen that the results for DFS has values of gear teeth close to highest value(127) which makes sense since in DFS we pop the last value in the stack and search all the way down through that so it can be said that DFS finds value more towards that search space. In the case of BFS, it goes through all values layer by layer so it can be seen from the results that due to this kind of search space the values are close to 11. It is important to note though that BFS gives better results for the cases as the set of gears is small in BFS compared to DFS, except for the **last two cases (wout = -2345 and 2) where BFS runs out of memory giving no results.**
2. **Transition function** for this problem would be to prioritize the priority queue such that we are able to pop out the smallest value in the queue every time we check goal and create children as we need to minimize the number of teeth.

Comparing the results of BFS and UCS, it can be seen that for most cases, minimizing the number of teeth leads to larger set of gears making the gear train large. Since UCS monitors the path travelled it reduced the search space extensively compared to BFS.