CS534 ARTIFICIAL INTELLIGENCE

Project Progress Report

Chen Chen, Jiacheng Wang, Yang Cao, Yunhe Tang

1. Summary of goals

This research focuses on investigating possible methods to improve the performance of generic algorithm in dealing with problem of Sudoku (9×9). Several ways of improving performance will be illustrated in this research. In addition, methods being proposed will be implemented in order to verify whether they are preferable optimization to the problem. In the original project proposal, some basic procedure are considered to be modified, including crossover process, mutation process, and selection of initial population. However, after deliberate consideration, we have decided to put more emphasis on fitness function instead of structure of generic algorithm. The revised project plan will be discussed with more detail in the third part.

2. Progress to date

So far we have already finished about half of work needed for this research. Specifically, we have finished reading papers concerned about taking advantage of genetic algorithm to solve Sudoku problem. In addition, we have proposed several methods that could be beneficial for speeding up solving the Sudoku problem. As mentioned in the first part, the target we are looking for improvement is all about fitness function, rather than other procedure. The papers that we read proposed two different version of fitness functions (with the precondition of each sub square is already satisfied the rule for sub square). They are:

1) F1: The fitness function returns the sum of all unique numbers in current matrix. If a number appears only once in current row or column, it is counted as a unique number. In this sense, taken the final solution as an input, the fitness function should zero, which means that there is no duplicate number existed in either same row or column.
2) F2: The fitness function check the sum of every number in a row or column and sum their difference with 45 (45 is the sum of 1 to 9). This fitness function consider that the more the sum of a row or column close to 45, the closer it is to the final solution.

Based on two fitness functions above, we have proposed another four fitness functions as possible optimization. These four fitness functions have not yet been proved mathematically that they are in one way or another better than the two above. We are going to implement them and analysis the result latter. The following are brief descriptions of four fitness functions: (with the precondition of each sub square is already satisfied the rule for sub square)

1) F3: The first fitness function we figured is based on F2 mentioned above. According to its definition, it reckons that, for a row or column, the closer the sum of every numbers get to 45, the better it is. If fitness function of matrix is zero, it claims that it has found the solution. However, we found it not true. If every row and column sum up to 10, it can be an incorrect solution, for a 4×4 Sudoku matrix. What we do is bring in an assist fitness function H, which can determine the total number of conflicts in matrix. In this base, we have:

   $$F3 = a \times F2 + b \times H \text{ (which } a + b = 1)$$

   F3 is better than F2 is because it can avoid incorrect solution. It guarantee that if F3 equals to zero, the solution found.

2) F4: although F3 guarantee the correct solution been found, it has a major drawback. Since one of the biggest advantage of F2 is it can calculate fast (only contain summation operation). But if we bring in function H, which calculates on conflicts, we sacrifice the advantage of F2. Therefore we consider not to use H (which is time costing) but use a cheaper function G. For a row or column: it calculates the production of each number and compare it with the 9!. So the formula becomes:

$$F4 = a \times F2 + b \times G \text{ (which } a + b = 1)$$

F4 is better than F3 in performance and F4 leads to correct solution. Because if there are 9 numbers which in range of 1 to 9, and sum of them is 45 and production of them is 9!, these nine numbers must be one of a random sequence of 1 to 9. In another words, our goal is to determine if nine numbers in same row (or column) are unique (no one shows up twice).

3) F5: if we consider each row or column is a vector, then we can calculate the distance from each row/column to a particular sequence 1 to 9. Imagine a nine dimension space, we have a point (1,2,3,4,5,6,7,8,9), and for each row or column represented as a point. To calculate the distance, we first have to sort the sequence from small to large. Then we use k-means algorithm to get the distance between two points. If we get zero, then we know that these nine numbers must be one of a random sequence of 1 to 9.

4) F6: the above 5 fitness functions have their pros and cons. We can combine multiple fitness functions into one by assigning weight to each function. the formula will be:

$$F = a \times F1 + b \times F2 + c \times F3\ldots \text{ (Which } a + + c + \ldots = 1)$$

Our next step is to find an appropriate assignment of weights.

There is a new light come out of this research. Currently we are looking for taking advantage of bit operation to get the result of fitness function. More specifically, using XOR to detect duplicates. Firstly, computer can execute XOR extremely fast. Secondly, if a number XOR with another number which has the same value, the result is zero. We are expecting to figure out a way to take advantage of this operating and use it in fitness function. It is still not clear that if it works.

3. Revised project plan

Our original plan is to optimize the structure of generic algorithm. However, we figure out that framework of generic algorithm is very general. While the basic structure remains the same. The crossover process, mutation process is pretty much determined and there is not much thing to change. Fitness function, however, as a method of summarizing and representing how close a given design solution to achieving the goal, has numbers of way to do. Therefore, we put more emphasis on fitness function instead of other optimization.

4. Revised Detailed Schedule

WEEK	ISSUE
11.30 – 12.6	Implement the proposed fitness functions
12.7 – 12.13	Analyze the result of experiment