A study on

VARIOUS METHODS OF SUDOKU SOLVING

Using Genetic Algorithms
and
Cultural Genetic Algorithms

Department Of Computer Science And Engineering
National Institute of Technology Calicut
Calicut, Kerala 673 601

Winter Semester, 2012

Department Of Computer Science And Engineering
National Institute of Technology Calicut

Certificate

This is to certify that this is a bonafide record of the project presented by the students whose names are given below during Winter Semester 2011-12 in fulfilment of the requirement of the course on mini project.

Names of Students	Roll No
Aviral Nigam	B090871CS
Manish Kumar	B090925CS

Guide

Lijiya A.
(Asst. Professor)

Date

Abstract

The Sudoku puzzle is most frequently a 9 X 9 grid made up of 3 X 3 sub-grids. Some cells already contain numbers. The goal is to fill in the empty cells, one number in each, so that each column, row, and sub-grids contains the numbers 1 through 9 exactly once. Each number in the solution therefore occurs only once in each of three directions. Most puzzles are ranked as to difficulty, but the rankings vary from designer to designer. Genetic Algorithms are adaptive heuristic search and optimization algorithm premised on the mechanism of biological evolution of chromosomes. The basic concept of GAs is designed to simulate processes in natural system necessary for evolution, specifically those that follow the principles first laid down by Charles Darwin. Cultural Algorithms are a branch of evolutionary computation where there is a knowledge component that is called the belief space in addition to the population component. In this sense, Cultural Algorithms can be seen as an extension to a conventional Genetic Algorithm. The objective of this project is to study the concepts of Genetic Algorithm and Cultural Genetic Algorithm and finding an efficient method for solving Sudoku puzzles.

Contents

Chapter 1

Introduction

Puzzles have been fascinating humans since their development as the most intelligent beings on this planet and finding their solution has been one of their most common time pass activity. But since last century these puzzles have found immense importance in solving day to day problems and since then mathematicians and algorithmist have been researching on finding easiest and most efficient solutions to these problems. One of the most common puzzles seen these days is Sudoku and there have been lot of research going on in finding solutions to this puzzle using various mathematical and computational techniques.

Chapter 2

Problem Statement

To study Genetic Algorithm and Cultural Genetic Algorithm and finding its application in Sudoku solving.

Chapter 3

What is Sudoku?

Sudoku is a logical puzzle game, originally created in puzzle books and then made available in countless newspaper worldwide. Number puzzles appeared in newspapers in the late 19th century. On July 6th, 1895 La France framed a puzzle almost similar to modern Sudoku. These weekly puzzles were a feature of French newspaper for about a decade but disappeared about the time of First World War. The modern Sudoku was most likely designed by Howard Gans, a 74 year old retired architect and freelance puzzle constructor from Indiana. The puzzle was introduced in Japan by Nikoli in the paper "Monthly Nikolist" in April, 1984. In 1986, Nikoli introduced two innovations: the numbers given were restricted to no more than 32 and the puzzles became symmetrical.

Sudoku is a Japanese, fun puzzle game. It requires the player to fill in the 9x9 square grids with numbers one to nine. The numbers should be arranged in such a way that each row, column or box contains one of each number. Some of the characteristics of Sudoku are :

- A Sudoku should have 30 or less initial values filled in out of 81 total.
- Sudoku should have rotational symmetry.
- Sudoku solving should be entirely logical, no guess work should be required.
- There must be only one solution.
- The difficulty of a Sudoku is inversely proportional to the number of initially filled cells.

3.1 Simple Sudoku Solution

1. Take a Sudoku and check for its validity. A Sudoku is valid if none of the values in it are repeated in their corresponding row, column or

grid.

2. Now initialize the empty cells by zero.
3. Now start traversing the Sudoku from left to right column wise in each row and whenever a zero is encountered start replacing it with values from 1 to 9 and check for the Sudoku validity with each new value and stop when validity is achieved.
4. Stop when all the rows have been checked.

3.2 Code Implementation

C language was used for implementing the code. As such no major problems were faced while implementing the solution code. Complexity of the solution is polynomial in order of n, where n is the number of initially unfilled cells.

Chapter 4

Genetic Algorithm

Genetic Algorithm is a search heuristic and optimization algorithm that mimics the process of natural evolution. Before going into further details of genetic algorithms, let us learn a little more about its background. Genetic Algorithm is based on one of the most important theories human kind has ever come across i.e., "The Theory of Evolution" proposed by Charles Darwin. This theory mainly states how such a complex human structure like us evolved from a single celled organism and how it has undergone large number of biological changes over the century to become what we are now. The theory has one very important postulate i.e., "Survival Of The Fittest" which forms the basis of genetic algorithms. By this he explains the most fundamental truth of the natural world that there exists a "Natural Selection" of favorable variations and a fierce "Struggle for Existence" in all living beings, thereby ensuring the survival of the fittest.

Genetic Algorithm was designed by John Holland at the University of Michigan. One of the key concepts in Genetic Programming using Genetic Algorithms is its "Fitness Function". A fitness function is a particular type of objective function that is used to summarize as a single figure of merit, how close a given design solution is to achieving the set aim.

The key concepts of Genetic Algorithm are discussed below -:

- Each design solution is represented by a string of numbers (preferably zero and one i.e., binary representation).
- Survival of the fittest: Algorithm replaces 'n' worst solutions from the first generation with 'n' newly generated solutions in every phase.
- Each genetic algorithm goes through 4 basic steps - Initialization, Selection, Reproduction and Termination.
    1. Initialization phase comprises of selecting a population which acts upon the problem and from which solution can be derived.

2. Selection phase deals with the fitness function with selects the best individuals from population for reproduction.
3. Reproduction phase has two major portions Crossover and Mutation.
   (a) Crossover is done by selection by selecting one or more than one crossover points from where crossover can be done by interchanging the two halves of the solution.
   (b) Mutation is done by slightly changing the population to get a new better population.
4. Termination decides when to stop the algorithm after an optimal solution has been achieved.

Basic Pseudocode for Genetic Algorithm :

```
Begin:
        t=0;
        Initialize Population POP(t);
        Evaluate Initial Population POP(t);
        repeat:
                Perform competitive selection on POP(t);
                Create population POP(t) from POP(t-1)
                by Crossover and Mutation;
                Evaluate Population POP(t);
        until Termination
End
```

4.1 Sudoku Solution using Genetic Algorithm

4.1.1 Solution Space Representation

To represent the solution space, we treated all the empty cells as a combination of integers ranging between 1 and 9, as one individual. Therefore the solution space consists of only one population group which allows for greater individual interactions making it computationally less demanding.

4.1.2 Fitness Function

Initialize the value for the row fitness, column fitness and grid fitness with zero. For fitness function, check the number of repetition in that particular row and every time when there is repetition of any number except zero, increase the fitness counter of row by one. Same should be applied to columns and grids. For calculating the fitness value of each solution of Sudoku, add column fitness, row fitness and grid fitness. The fitness of the solution decreases with the increase in the value of the fitness function.

4.1.3 Cell Fix Function

For all the elements present in the Sudoku: Take each cell and compare it with all the cells in its corresponding row, column and grid, and whenever there is repetition of that number, increase the value of fix by one. If the value of fix after comparing all the elements is zero, then fix that element.

4.1.4 Mutation

Mutations are applied only inside a grid. Mainly, three different mutation strategies are commonly used: Swap Mutation, 3-Swap Mutation, and Insertion Mutation. We have used swap mutation strategy. In swap mutation two unfixed cells are randomly chosen and are swapped. If it is illegal to perform swap, mutation is omitted. During reproduction, the mutation function is implemented on each individual of the population. This is done to ensure that the population converges to a global minimum as opposed to a local minimum.

4.1.5 Termination Condition

The solution terminates when the value of the fitness function for the solution is zero.

4.2 Code Implementation

C++ language was used for implementing the code. The complete code was composed of basic functions of genetic algorithm. The major problem faced while implementation was in maintaining an efficient interaction between these functions, which didn't turn out that well, resulting in loss of certain valuable information, which could have been extracted from that code. There arose a problem while comparing and commenting on the time complexity of code as compared to the brute force method we implemented earlier, but indeed the space complexity was higher, due to the inclusion of various functions as required by genetic algorithm.

Chapter 5

Cultural Genetic Algorithm

Culture is defined by Durham as a "System of symbolically encoded conceptual phenomenon that is socially and historically within and between populations". Cultural Genetic Algorithm is an evolutionary optimization technique where individuals are influenced both genetically as well as culturally.

CGA is a variant of Genetic Algorithm which includes a belief space. Belief space represents the domain of knowledge about the population. Belief space is updated after every generation depending upon the new population,which can be selected by means of fitness function.

The belief space can be divided into -:

- Normative Belief: where there is a particular range of values to which an individual is bound.
- Domain Specific Belief: where the information about the domain of the problem is available.
- Temporal Belief: where the information about important events in search space is available.
- Spatial Belief: where the topographical information of the search space is available.

Basic Pseudocode for Cultural Genetic Algorithm :

```
Begin:
        t=0;
        Initialize Population POP(t);
        Initialize Belief Space BLF(t);
        repeat:
                Evaluate Population POP(t);
                Adjust(BLF(t), Accept(POP(t));
```

```
                Adjust(BLF(t));
                Variation(POP(t) from POP(t-1));
        until Termination
End
```

5.1 Sudoku Solution using Cultural Genetic Algorithm

5.1.1 Population Space

We have used the same population space as used in the solution using genetic algorithm.

5.1.2 Belief Space

Consists of 4 basic categories:

- Normative Belief: each 3x3 grid contains entries between 1 and 9.
- Domain Specific Belief: each 3x3 grid can only contains entries which are integers.
- Temporal Belief: where the information about important events in search space is available.
- Spatial Belief: when a mutation is applied to a grid, it cannot alter the value of a fixed cell. Every time mutation is called, belief space is updated.

5.1.3 Evaluate Population

Calculates the fitness of population and the fitness function is same as one used in the solution using GA.

5.1.4 Adjust Belief Space

Belief space is updated according to the new population after each mutation.

5.1.5 Variation

It is same as the mutation function used in the solution using GA where the new individuals replace the individuals in the population which have the lowest fitness values and the new generations can be produced using current state of the belief space.

5.1.6 Termination Condition

Termination condition is same for both GA and CGA.

5.2 Code Implementation

Implementation of cultural genetic algorithm has not been covered in this project, but that can be done on the lines of genetic algorithms.

Chapter 6

Conclusion

Sudoku puzzles can be solved with a simple approach, Genetic Algorithm and Cultural Genetic Algorithm. Simple solution is a brute force method checking just the minimal necessary constraints. GA can solve Sudoku puzzles relatively effectively.The GA results can of course be enhanced by adding problem related rules. However, if one adds too much problem specific logic to the Sudoku solving, there will be nothing left to optimize, therefore we decided to omit most problem specific logic and try a more straight forward GA optimization approach. And same conditions apply for Cultural Genetic Solution where all the problem related rules are defined in the belief space and hence the solution becomes more problem specific making it more efficient than the other two methods.

References

[1] Goldberg, David E (1989), "Genetic Algorithms in Search, Optimization and Machine Learning", *Kluwer Academic Publishers*, Boston, MA.

[2] R. Reynolds.(1994). "An Introduction to Cultural Algorithms." *Proceedings of the 3rd Annual Conference on Evolutionary Programming.* River Edge, New Jersey:World Scientific;131-139.

[3] Meir Perez and Tshilidzi Marwala.(2008). "Stochastic Optimization Approaches For Solving Sudoku." *School of Electrical and Information Engineering, University of the Witwatersrand*, Johannesburg, South Africa.

[4] T. Mantere and J. Koljonen.(2007). "Solving and Rating Sudoku Puzzles using Genetic Algorithms." *Proceeding of the IEEE Congress on Evolutionary Computation*, 1382-1389.

[5] Kedar Nath Das and Sumit Bhatia. "A Retrievable GA for Solving Sudoku Puzzles."

[6] Darrell Whitley. "A Genetic Algorithm Tutorial." *Computer Science Department, Colorado State University,* Fort Collins.

[7] D Nagesh Kumar.*Lecture Notes*,IISc, Bangalore.

[8] Sudoku. http://en.wikipedia.org/wiki/Sudoku.

[9] Genetic Algorithm. http://en.wikipedia.org/wiki/Genetic_algorithm.

[10] Cultural Genetic Algorithm. http://en.wikipedia.org/wiki/Cultural_algorithm.