Applied Soft Computing 45 (2016) 27–39

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

A new membrane algorithm using the rules of Particle Swarm Optimization incorporated within the framework of cell-like P-systems to solve Sudoku

Garima Singh*, Kusum Deep

Department of Mathematics, Indian Institute of Technology Roorkee, Roorkee 247667, India

ARTICLE INFO

Article history:
Received 6 October 2015
Received in revised form 7 March 2016
Accepted 15 March 2016
Available online 4 April 2016

Keywords:
Sudoku
Membrane algorithm
P-system
Particle Swarm Optimization

ABSTRACT

Sudoku, of order n, is a combinatorial puzzle having partially filled $n^2 \times n^2$ grid consisting of sub-grids of $n \times n$ dimension. In this paper, a new membrane algorithm, namely MA_PSO_M, is presented. It uses the modified rules of Particle Swarm Optimization coupled with a carefully designed mutation operator within the framework of cell-like P-systems. Another significant contribution of this paper is the novel way in which the search space for solving the Sudoku problem is defined. Initially, the proposed algorithm is used to solve Sudoku puzzles of order 3 available in literature. On the basis of experiments performed on sample Sudoku puzzles of 'easy' and 'medium' difficulty levels it is concluded that the proposed membrane algorithm, MA_PSO_M, is very efficient and reliable. For the 'hard' and 'evil' difficultly levels, too the algorithm performs very well after incorporating an additional deterministic phase. The performance of the algorithm is further enhanced with an increased population size in a very small computational time. To further demonstrate efficiency of algorithm it is applied to Sudoku puzzles of order 4. The obtained results prove that the proposed membrane algorithm clearly dominates any of the PSO based membrane algorithm existing in the literature.

1. Introduction

Sudoku is a logic based number placement puzzle. It is a single player mind game played on a partially filled $n^2 \times n^2$ grid. The grid contains $\left(n^2\right)^2$ cells in which only digits ranging from 1 to n^2 can be placed. Some of the cells of grid already have a digit placed in them while a player has to fill the remaining empty cells. However, while doing so a player has to follow specified rules according to which each of these numbers should occur exactly once in all the rows, columns and sub-grids, besides the fact that none of the fixed cells can be changed. For a proper Sudoku puzzles, there exist a unique solution that satisfy all the given constraints. The solution for most popular form of Sudoku, i.e., Sudoku of order 3 having 9×9 grid, can be attained very quickly for any of its difficulty levels as the easy Sudoku can be solved just by using algorithms based on logic derived from human intuitions [1]. However, a bit harder puzzles can be solved using backtracking algorithms [1]. However, the computational time of the backtracking algorithm increases, with the increase in the number of guesses to be made to solve the Sudoku and hence its efficiency decreases. Moreover, a general problem of solving Sudoku, of order n, is NP-complete [2]. Thus, deterministic algorithms fail for the Sudoku of higher order. Hence, researchers tried to solve this problem using nature inspired optimization techniques as these techniques can solve even the difficult puzzles as efficiently as the easy puzzles.

Nature inspired optimization techniques explore and exploit the complete search space in order to find the global optimum solution for the problem. However, the search space besides having global optima may also contain many local optimal points. Hence, these algorithms sometimes tend to get stuck in region of local optima. In such cases, the operators like mutation help the algorithm escape the local optima. Sudoku, being constrained in nature, can have many local optima satisfying some of its constraint while not all of them. Hence, not all the nature-inspired techniques can successfully solve these puzzles. In literature, it can be found that techniques like Genetic Algorithms which have such operators as the part of the process are more successful as compared to algorithms like Particle Swarm Optimization [3].

* Corresponding author.
E-mail addresses: garimachauhan.iitr@gmail.com (G. Singh), kusumdeep@gmail.com (K. Deep).

http://dx.doi.org/10.1016/j.asoc.2016.03.020
1568-4946/

In this paper, a membrane algorithm MA_PSO_M, which uses the parallel and distributed framework of cell-like P-systems, is presented to solve the general form of Sudoku, i.e., Sudoku of order n. The rules used within the framework of cell-like P-systems are that of modified Particle Swarm Optimization coupled with specifically designed mutation. Since, PSO is a population based algorithm, the population from a defined search space has to be generated in each membrane of the system. However, in this paper, a different way (described later) to generate any individual of a population has been defined. Hence, the search space for any given Sudoku puzzle is different from that of others. The defined PSO rules have been applied on this differently generated population set. To avoid the trapping of the search process in any region of local optima a mutation operator has been applied following specified conditions. Initially, the algorithm is applied on the set of randomly selected Sudoku puzzles from the database available at Ref. [4]. The set contains 15 puzzles each of 'easy', 'medium', 'hard' and 'evil' difficulty levels. The results show that the algorithm works efficiently for 'easy' and 'medium' levels but success rate for puzzles of 'hard' and 'evil' levels is low. Thus, an additional deterministic rule is incorporated in the proposed algorithm and is applied on the same set of Sudoku puzzles. The obtained results show that updated algorithm, with an additional phase, can efficiently solve puzzles of all difficulty levels. To further prove the efficiency of updated algorithm it is applied to Sudoku puzzles of order 4 available in an online database at Ref. [5].

This paper is organized as follows: Section 2 briefly describes the concept of PSO based membrane algorithms and the rules of Sudoku. This section also presents a brief literature survey on the topic. The proposed membrane algorithm, MA_PSO_M, is presented in Section 3. Section 4 describes the deterministic phase introduced in algorithm to improve its performance. The obtained results are discussed in Section 5. Section 6 presents conclusions of the work done and the future aspects to be studied.

2. Background

2.1. Particle Swarm Optimization based Membrane Algorithms

P-systems are computing models, designed by Păun [6] in 1988, based on the functioning of living cells. They provide a parallel and distributed framework having pre-defined number of membranes. A cell-like P-system means that the membranes of P-system are arranged in hierarchical form. That is, there exists a single skin membrane containing all other membranes of the system. The internal membrane can however be either in hierarchical form, i.e., one membrane containing other membranes, or in elementary form, i.e., membrane containing no other membrane, or the combination of both. Each of these membranes contains a set of objects and rules. In order to process them some rules are required to evolve the objects present in the membranes of the model. Such rules are known as transformation or evolution rules. Few other rules are required to transfer data or interact among the membranes of the system which are known as communication rules. Neither of these rules is predefined for the system. They are, however, defined as per the requirements of the purpose for which they are used. The evolution rules can either be deterministic or stochastic. The algorithm using the update rules of Particle Swarm Optimization as an evolution rules within the membranes of P-system model are known as PSO based membrane algorithms. The update equation for the basic PSO as proposed by Kennedy and Eberhart [7], after studying the swarm behavior of the bird flocks, is given by:

$$v_{ij}(t+1) = v_{ij}(t) + \varphi_1 \times r^t_{1(ij)} \times \left(\text{pbest}_{ij}(t) - x_{ij}(t)\right) + \varphi_2 \times r^t_{2(ij)} \times \left(\text{gbest}_j(t) - x_{ij}(t)\right) \quad (1)$$

$$x_{ij}(t+1) = x_{ij}(t) + v_{ij}(t+1) \quad (2)$$

where, Eq. (1) is velocity update and Eq. (2) is position update of any j^{th} dimension of the i^{th} particle of the swarm or the population in any t^{th} iteration. φ_1 and φ_2 are acceleration constants while $r^t_{1(ij)}$ and $r^t_{2(ij)}$ are uniform random numbers in the range $[0, 1]$. However, later several modifications of PSO technique have been introduced in literature.

2.2. Rules of Sudoku

A Sudoku puzzle of order n, is a $n^2 \times n^2$ grid, consisting of n^4 cells, denoted by $\text{cell}(i,j)$, $\forall(i,j) = \{1, 2, \ldots, n^2\}$, arranged in the form of n^2 sub-grids of dimension $n \times n$, denoted by $\text{subgrid}(i,j)$, $\forall i,j = \{1, 2, \ldots, n\}$. Any Sudoku is filled using n^2 copies of digits ranging from 1 to n^2. In a given puzzle, some of the cells are filled, known as fixed cells. The objective of the problem is to fill the remaining empty cells, subject to the following constraints:

i *Fixed cell constraint*: The numbers placed in the fixed cells cannot be changed.
ii *Row constraint*: All numbers $\{1...n^2\}$ appear exactly once in each of the n^2 rows of Sudoku.
iii *Column constraint*: All numbers $\{1...n^2\}$ appear exactly once in each of the n^2 columns of Sudoku.
iv *Sub-grid constraint*: All numbers $\{1...n^2\}$ appear exactly once in each of the n^2 sub-grids of Sudoku.

An example illustrating the Sudoku puzzles and its corresponding solution is presented in Fig. 1.

2.3. Literature review

A fair quantity of research work related to the problem of obtaining exact solution for Sudoku can be found in literature. Initial research was on deriving the brute force techniques to find the solution, which, however, were efficient to only a limited set of the puzzles. As with the increase in the difficulty level or the order of the Sudoku the techniques failed due to their high computational cost. This failure inspired researchers to use nature inspired optimization techniques to attain the solution of the Sudoku puzzles. The puzzles were solved using various techniques like Genetic Algorithm [3,8], Particle Swarm Optimization [3,9–12], and so on. The results however showed that while techniques like GA performed considerably well but the performance of PSO is not quite effective. In 2007, Moraglio et al. [9,10] tried to solve the Sudoku problem using Geometric Particle Swarm Optimization (GPSO) technique. They however, concluded that GPSO is not best suited algorithm for the purpose of solving Sudoku puzzles correctly. In fact, they have suggested that in general, neither evolutionary algorithms nor meta-heuristics are the best techniques to solve Sudoku, as in these techniques there is no systematic manner to exploit the constraints of the problem to converge the search space associated with it.

In 2008, Perez and Marwala [3] used Repulsive Particle Swarm Optimization to attain the solution to Sudoku puzzles. This variant of basic PSO is designed for search spaces with many local minima. In basic PSO, particle's velocity is only dependent on its current and best velocity and the velocity of the best particle of the swarm as shown in Eq. (2) but in this modification a velocity of any randomly chosen particle of the swarm is used as a repulsive factor so as to prevent premature convergence. However, even this modification could not get success in achieving the solution of Sudoku puzzles. In the same year, Hereford and Gerlach [11] developed a different variation of PSO specifically for the purpose of discrete optimization