www.ijecs.in
International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume 5 Issue 09 September 2016 Page No.17842-17844

Solving n-Queen problem using modified Rakhya's approach for parallel system

Loveleen Kaur, Prateek Gupta
Department of Information Technology, Jabalpur Engineering College Jabalpur, India
loveleenkaurpabla@gmail.com

Abstract — **The N-queen problem is to place N queens on an NxN chessboard such that no two queens attack each other. Traditionally, algorithm has been written for serial computation recent advances in software and hardware allow parallelism. This paper investigate the possibility of finding the solutions of N-queen problem by recently introduced Rakhya' method modified to run on parallel system.**

I. INTRODUCTION

In the Nineteenth Century, Babbage used parallel processing in order to improve the performance of his Analytical Engine. Indeed, the first general purpose electronic digital computer, the ENIAC [1,2], was conceived as a highly parallel and decentralized machinery with twenty-five independent computing units. However, the early computer developers rapidly identified two obstacles restricting the widespread acceptance of parallel machines: the complexity of construction; and the seemingly high programming effort required. As a result of these early setbacks, the development thrust shifted to computer with a single computing unit, to the detriment of parallel designs.

Nowadays, multicourse Central Processing Units (CPUs) and Graphical Processing Units (GPUs) [3] are commonly available with personal computers. Indeed GPU is a many core processor which supports thousands of parallel run able threads. These devices use very large number of processing cores to achieve high parallelism. In order to fully utilize this large number of processing cores, programmers need to write parallel program which are scaled very well to take advantage of underlying architecture.

The N-queen problem is a classical problem in computer science in terms of backtracking. It is used for finding optimal solutions in Neural Networks [4], Las Vegas Algorithm [5] and genetic algorithm [6]. This paper discusses the ground of solving the N-queen problem using parallel processing concept on Rakhya's approach.

II. N-QUEEN PROBLEM

The n-queen problem is to place n non attacking queens on an nxn board. The generalization of the problem of

Putting eight non attacking queens on a chessboard, which was first posed in 1848 by M. Bezzel, a German chess player. The earliet paper on the general n-queens problem is F.J.E. Lionnet's 1869 work. Gauss is often cited as the originator of this problem or the first to solve it. Indeed, Bezzel seems to have come up with the problem before Gauss, and Gauss wasn't the first to solve the problem.[7] It is known that the maximum number of queens that can be placed on an nxn chessboard, so that no two attack one another, is n. Since each queen must be on a different row and column, we can assume that queen i is placed in i-th column. All solutions to n-queen problem can therefore represented as n-tuples (q1,q2…,qN) that are permutation of an n-tuple (1,2,3….,N). Position of a number in the tuples represent queen's column position, while its value represents queen's row position (counting from the bottom) Using this representation, the solution space where two of the constraints (row and column conflicts) are already satisfied should be searched in order to eliminate the diagonal conflicts. Complexity of this problem is O(n!). The problem with determining the correctness of solution for n-Queen problem is the same as for any combinatorial

problem: The solution is either right or wrong. Thus, a fitness function must be able to determine how close a wrong solution is to a correct one. The simple method of finding a conflict [8], consider an n-tuple i-th and j-th queen shared a diagonal if:

$$i-q(i)=j-q(j) \quad (1)$$

or

$$i+q(i)=j+q(j) \quad (2)$$

Which reduces to?

$$\| q(i)-q(j)\|=\|i-j\| \quad (3)$$

This simple approach results in fitness function with complexity of O(n^2). Generalization of fitness function is summarized in the following figure I.

```
set left and right diagonal counters to 0

for i=1 to n
        left_diagonal (i+qi)++
        right_diagonal(n-i+qi)++
end
sum=0
for i=1 to (2n-1)
        counter= 0
        if (left_diagonal[i] > 1)
                counter += left_diagonal[i]-1
        if (right_diagonal[i] > 1)
                counter += right_diagonal(i)-1
        sum += counter / (n-abs(i-n))
end
```

Figure I. Fitness function for N-queen problem

III. RAKHYA'S METHOD

Rakhya's method [9] is recently introduced where the solutions to the given problem are found with the help of two operations named as jiggling and shifting[6]. These operations are performed on the predefined arrangement of queens, known as A & B patterns. The solutions are obtained in two phases. In the first phase, a definite solution is found using a definite operation set. In the second phase, other solution is generated from the solution achieved in the prior phase. It provides the solution for all values of n>3. There are six cases in that operation set based on the value of n. These cases are actually six set of numbers. The union of these sets is a set of all integers greater than 3.

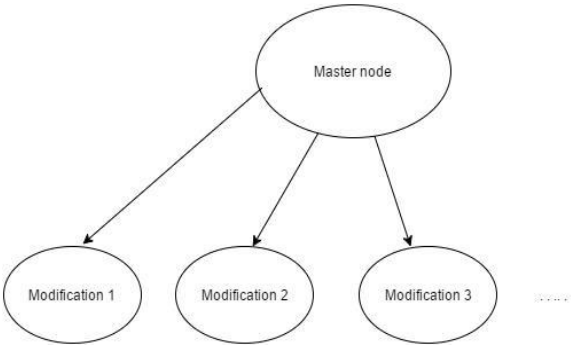
The principle behind this approach is the principle of consecutive correction i.e., if there is any error in the present arrangement of queens, a correction is done and the new arrangement is checked again for errors. The arrangement in which no more correction is required is said to be the final solution. Rakhya's method not only provides the solution in minimum possible steps but it also enables us to achieve other final solutions from the already obtained solutions. Rakhya (A) and Rakhya (B) patterns column and place the second queen there. This process continues for the remaining queens and thus a solution is achieved these two patterns achieve conflict free positions for all the queens. They are designed to remove the vertical and horizontal conflicts, although there might still be one or more diagonal conflicts left. Jiggling and shifting operations are used in the second level, to remove the diagonal conflicts. Applications of these conflicts does not hinder the vertical and horizontal conflict free arrangement of queens attained in the first level. Jiggling is always performed in pairs. Tabular representation of equations of different cases and their respective operations set is given in Table I. where N = no. of queens.

CASE	EQUATIONS	OUTPUT	METHODS
A	N≠6K+2	EVEN	R(A)
B	N≠6K+3	ODD	R(B)
C	N=6K+2 ,N/2=0	EVEN	R(A) J(r) or R(A) J(l)
D	N=6K+2, N/2≠0	ODD	R(A) J(r) S(r) OR R(A)J(l)S(l)
E	N=6K+3, ((N+1)/2)≠0	ODD	R(B)J(r)S(R) S(R)
F	N=6K+3, ((N+1)/2)=0	EVEN	R(B)J(l)S(L) S(L)

TABLEI
CASES AND THEIR RESPECTIVE OPERATIONS

IV. MODIFIED RAKHYA'S APPROACH

Like the regular parallel algorithm Rakhya's method which has been original written down for serial computation can be modified to execute over parallel architecture. Fig III shows the overall structure of modified algorithm.




```
for i = 1 to slave_iterations
        modify base pattern
        evaluate validity of outcome
        if solution found
        return solution to master
end
```

Fig III. Slaves pseudocod

The main idea is to distribute the variations of conflict removal methods across slaves. In a classic configuration, the master generate the basic pattern which is free from horizontal and vertical conflict on the basis of one of the six cases in Rakhya's approach. But there might still be way to remove the diagonal conflicts but, traditional one or more diagonal conflict left in order to remove that the basic skeleton which lead to the group of final solutions is distributed to slave along with the variations in correction techniques. Slave generate the modified pattern which may or mayn't be free from diagonal conflict but if they are according to fitness function they are sent back to the master. Which collaborate the result and generate the group of similar solutions for the value of n. Fig II shows the pseudocode for master and Fig III shows the pseudocode for slaves in the modified version.

```
create initial pattern

create slaves
distribute modifications to slaves

while not done
        start slaves
        wait for slaves to finish
        return valid results to master
end
```

Fig II. Master pseudocode

After efficiency and speed of the algorithm were improved this way. Problem size used were 8, 104, 500 and 1004 to cover the

diversity of cases. For all problem implementing the above pseudo code in java, we observed that sizes, two threads were executing the execution of jiggling and shifting operations on either of the halfs. Ten runs were performed for each problem size, and average results are given in the table II. The machine used was core i5,third generation OSX El Capitán. Each values in the table are in microseconds.

V.CONCLUSIÓN

For the purpose of finding the answer, n-queen problem is classified in 3 classes: 1. Finding all answers 2. Finding some answers and 3. Finding the first answer. Clearly Rakhya's approach belongs to class 2 set of solutions. Original method defines there are more than one possible computational algorithm ends with one solution by followwing one of the many method. This paper discuss that N-queen problem can be solved using modified Rakhya's method on parallel architecture to generate the group of similar solutions in a single execution. Although n-queen problem has limited practical use, it represents a large class of NP problems that cannot be solved in a reasonable amount of time using deterministic methods. As the number of solutions increased exponentially with the increase in N.

TABLE II
AVERAGE VALUES OF THE TEST RUN

QUEENS	T 1	T 2
8	0.3	4.8
104	4.3	2.8
500	5.2	3.4
1004	5.5	4.9

VI. REFERENCES

[1] Arthur w. Burks (1947), "Electronic Computing Circuit of the ENIAC", Proceeding of the institute of Radio Engineers Vol. 35 No. 8, pp 1-3.

[2]https://en.wikipedia.org/wiki/History_of_computing
-hardware "First Generation Machine"

[3] General Purpose Computation Using Graphics Hardware https://www.gpgpu.com

[4] Y. Ding, Ye Li, M. Xiao, Q. Wang, and D. Li, "A high order neural network to solve N-queens problem," in UCNN, July 2010, pp. 1 -6.

[5] T. J. Rolfe, "Las vegas does n-queens," vol. 38, no. 2. New York, NY,USA: ACM, June 2006

[6] c. Zeng and T. Gu, "A novel assembly evolutionary algorithm for nqueens problem," in Computational Intelligence and Security

Workshops, 2007. CISW 2007. International Conference on, dec. 2007.

[7] Y. Alavi, D.R. Lick, J. Liu, Strongly diagonal Latin squares and permutation cubes, in: Proceedings of the Twenty-fifth Southeastern International Conference on Combinatorics, Graph Theory and Computing (Boca Raton, FL, 1994), vol. 102, 1994, pp. 65–70.

[8] Ellis Horowitz and Sartaj Sahni, Fundamentals of Computer Algorithms, Computer Science Press Inc. Rockville, MD, 1978

[9] Saurabh Rakhya, Saurabh Singh "Rakhya's Method: A case based approach to solve n-queen problem", Electronics, Computer and Computation (ICECCO), 2014 11th International Conference 2014