Volume 3, Issue 12, December 2013 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

DNA based Heuristic Approach for N Queen's Problem

Priyaa.T*, Evangeline.D

Assistant Professor

IT Department & Kongu Engineering College, India

Abstract— This paper discusses the effective performance improvement in solving Constraint Satisfaction Problems using DNA Computing. Backtracking, a common approach to find finite solutions to the problem is considered. N-Queens problem, a traditional example of Backtracking is discussed and its solution is found using different combinations of DNA (Deoxyribonucleic Acid) strands. It further compares the time complexity of our solution with the traditional ones. Although researches have been done for providing solutions using silicon based computing, there is a vast reduction in time for calculating the solution of N Queen's problem using DNA computing.

Keywords— DNA strands, Silicon based computing, DNA Computing, Deoxyribonucleic Acid, N-Queens Problem.

I. INTRODUCTION

A Constraint Satisfaction Problem (CSP) is defined by a set of **variables** V1, V2, … Vn, and a set of **constraints**, C1, C2, … Cm. Each variable Vi has a non-empty **domain** Di of possible **values**. Each constraint Cj involves some subset of the variables and specifies the allowable combinations of values for that subset. A state of the problem is defined by an **assignment** of values to some or all of the variables. A **solution** to a CSP is an assignment that satisfies all the constraints. If a CSP requires a solution that maximizes or minimizes an objective function it is called "constraint optimization problem". **Backtracking** is a general algorithm for finding all (or some) solutions to some computational problem. A classic example for backtracking is N-Queen's Problem.

The N-Queen's problem is defined as assigning N queens on the N*N chess board so that no queens can attack each other. This problem is one of the NP complete problems, where number of possible solutions increases exponentially with the number of queens. There are many ordinary solutions for N-Queen's problem. Silicon based computing could process the data simultaneously not parallelly. DNA Computing is an emerging area which provides solutions to problems parallelly which is not possible in existing silicon based computing. This is the reason why the DNA computational time complexity is far less when compared to the traditional computing. In this paper, we propose a modified DNA computing algorithm for solving the N-Queen's problem which implements the ordinary idea with the help of DNA strands.

II. DNA COMPUTING

DNA computing is an emerging field where biological science can be applied to mathematical computation. It focuses on manipulations on DNA (Deoxyribonucleic Acid) which is found in all organisms. DNA is a double-stranded helix of nucleotides made up of four nitrogen-base polymers Adenine (A), Cytosine (C), Guanine (G) and Thymine (T) where C always pairs with G and T always pairs with A. Enzyme DNA polymerase which is the maker of life produces a "Watson-Crick" complementary strand in which every C is replaced by G, every G by C, every A by a T and every T by an A. For example, a molecule ATGC is polymerised to produce a new molecule with sequence TACG. This polymerization is applied in our proposed approach for taking the row and column values (i.e In chess board, if ATGC is the input value for row **1** then its complementary sequence TACG is value of column **1**). Polymerase is the amazing enzyme which makes DNA to reproduce which in turn makes cell to reproduce. A single strand of DNA is represented using $4*10^{21}$ bits and it is proved that 1 gram of DNA can hold about $1x10^{14}$ MB of data. So, the number of CDs required to hold this amount of information, lined up edge to edge, would circle the Earth 375 times, and would take 163,000 centuries to listen to. This makes it evident that DNA computers are faster with minimal power and storage requirements. DNA computing can be applied in cryptography, steganography, authentication, medical diagnosis and also in solving mathematical problems like Travelling Salesman Problem, Graph Colouring Problem, Chess Problem, etc... DNA computing is definitely the technology to watch out for in the coming years.

III. DNA EXISTING APPROACHES

With the parallel processing ability of DNA molecules, DNA computing becomes a powerful area for solving complex problems. Adleman was the one who first used DNA molecules to solve Hamiltonian path (HP) problem. HP is a path which connects the source and destination passing through the routes exactly once. In this, each city is assigned a DNA sequence of 8 letters, in which first 4 letters refers the first name of the city and next 4 letters refers the last name of the city. For example ACTTGCAG is the DNA sequence for Atlanta in which ACTT refers the first name of Atlanta and GCAG refers the last name of Atlanta. Flight number connecting the source city and destination city is given by

concatenating the last name of the source and first name of the destination city. Within a second, solution to the problem was obtained using DNA computing. His findings opened the field of practical DNA computing.

Most of the researches on DNA computing concerns solving NP-complete problems. Lipton [Lipton (1995)] demonstrated that a large class of NP-complete problems also could be solved in a polynomial time by using DNA molecules. DNA computing has been used in solving many other problems such as finding maximal clique [Ouyang et al. (1997)], satisfiability (SAT) problem [Braich et al. (2002); Lipton(1995)], calculation of multiplication of Boolean matrix [Oliver (1999)], breaking DES Leier et al. (2000)], chess problem [Faulhammer et al. (2000)], simulation of Boolean circuits [Ahrabian et al. (2005); Ahrabian and Nowzari (2004); Ogihara and Ray (1999)], arithmetic and logic operation [Barua and Misra (2002); Frisco (2000); Gupta et al. (1999)], simulating Turing machine and automata [Beaver (1995); Benenson et al. (2003); Gao et al. (1999); Rothemund (1996)], and molecular expert system [Wasiewicz et al. (2000)].

The N-Queen problem is a well-known NP Complete problem. The N-Queen problem consists in placing N queens in N*N chess board such that no one offends the others. One method of solving this is backtracking. Due to large time complexity of back tracking, many solutions to this problem are proposed.

IV. OPERATIONS IN DNA COMPUTING

Techniques of DNA strands are

1. **Merging/Mixing** – This operation is used to combine two DNA strands to one.
2. **Watson-Crick Pairing/ Annealing** - This operation is used to pair complementary strands of DNA. These are paired to form the famous double-helix structure of Watson and Crick. This is achieved by cooling a DNA solution, which encourages Watson-Crick pairing. As we have discussed already, every strand of DNA has its Watson-Crick complement. For example consider a DNA sequence ATGC, it has its own Watson-Crick complement TACG in which A in original sequence is replaced by T, T is replaced by A, G by C and C by G. When the two sequences ATGC and TACG meet each other, they both twist around each other to form the famous double duplex.
3. **Amplifying/Copying:** Polymerase Chain Reaction is used to make copies of DNA. The DNA polymerase enzymes perform several functions including replication of DNA/complementation of DNA. It needs a shorter oligonucleotide **primer** that makes a single DNA strand to replicate. For example the strand **GCTA** and **primer** are combined to form its complementary pair strand **CGAT.**
4. **Melting** - Melting is exactly opposite operation to annealing. This is achieved by heating a strand of DNA which results in separation of single DNA strand to two DNA strands.
5. **Separation by length** – This is achieved by gel electrophoresis for separating DNA strands by increasing length. In this, longer strands move slowly and the shorter strands move fast.
6. **Separation by sequence/Extracting** – This operation is used to obtain the DNA strands that contain desired pattern as a substring. For example, GATCATGCGCTA is the input to this operation and the desired string is CATG. The original DNA strand that containing CATG is extracted in to two strands as CATG and GATCGCTA.
7. **Cutting**: This operation cuts the DNA sequence after finding the predetermined sequence of bases. Biologically, this is done by restriction endonucleases. For example, GATCATGCGCTA is the input to this operation and make it to cut after every G. The resulting strands are G, ATCATG, CG, CTA.
8. **Substituting/Inserting/Deleting:** These operations will do substitute, insert and delete DNA bases respectively. Biologically it is done by using PCR site specific oligonucleotide mutagenesis.
9. **Appending** - This process appends a character or strand to the end of each sequence thus making the existing DNA strand still longer. This is done by ligases.
10. **Detecting** - In order to determine whether or not it contains at least one strand of DNA that survived all other process that done previously, this operation is done.
11. **Discarding/Destroying**: This operation is used to discard the DNA strand.

V. N QUEEN'S PROBLEM USING DNA COMPUTING

Over the past few decades, there were many researches made on N Queen's problem since it is one of the hard problems to solve. It is proved that DNA computing can solve this problem parallelly which is not even possible in fastest electronic computers. The reason why the DNA computers work parallelly is that the DNA molecules contain massive storage capacity as mentioned previously. Hence, its intermediate results are easily stored and retrieved which is impossible in silicon based computing. The important property of this problem is that it requires large number of computations. N-Queen's problem is the problem of placing 'N' non-offending queens on an N*N chess board. A queen can attack another queen vertically, horizontally, or diagonally.

A general N-Queen's problem is given the following constraints.

1. Only one queen can be placed in any row.
2. Only one queen can be placed in any column.
3. Only one queen can be placed on any diagonal.
4. Exactly *N* queens must be placed on the grid.

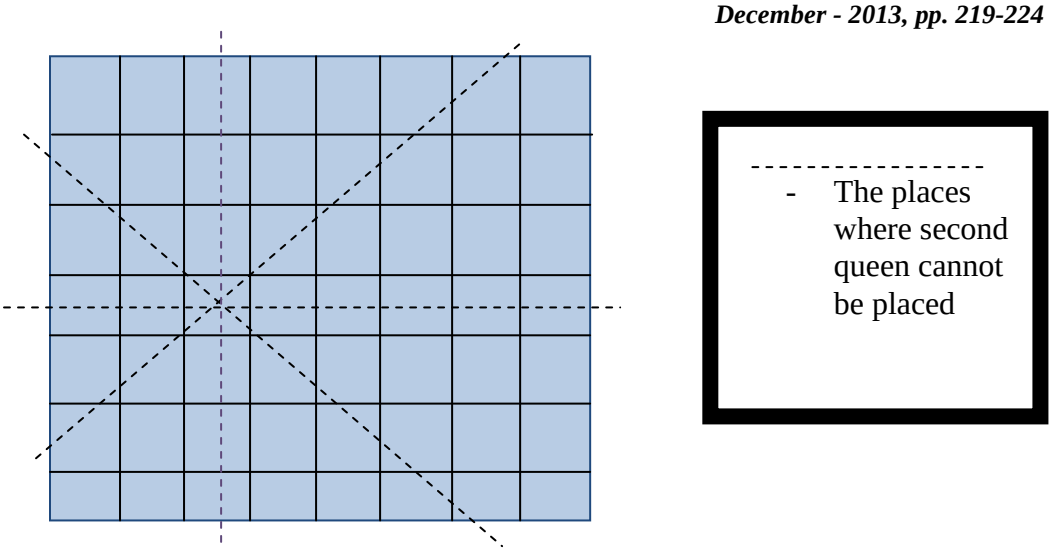


Fig 1: Placement restriction for second queen.

VI. PROPOSED APPROACH

In this section, we propose our approach to solve N Queens problem. Consider a 8*8 chess board. Usually the chessboard is numbered from 1,2,3,4,5,6,7,8 for rows and a,b,c,d,e,f,g,h for columns. To refer 5th row and 5th column, it is represented as **5e.** In our proposed approach, instead of giving 1 to 8 for representing rows and a to h for columns, we have given DNA sequence for the each row and its complementary DNA sequence is given to its corresponding column which is show in below table.

TABLE I

SAMPLE DNA STRANDS FOR ROWS AND COLUMNS

	ATGC	TGCA	GCAT	CATG	TACG	ACGT	CGTA	GTAC
TACG								
ACGT								
CGTA								
GTAC								
ATGC								
TGCA								
GCAT								
CATG								

Thus row values 1, 2,3,4,5,6,7,8 are replaced by DNA sequence TACG, ACGT, CGTA, GTAC, ATGC, TGCA, GCAT, CATG respectively and column values a, b, c, d, e, f, g, h are replaced by the complementary of row values such that the column values for our scenario are ATGC, TGCA, GCAT, CATG, TACG, ACGT, CGTA and GTAC respectively. These complementary strands are produced by amplification which uses Polymerase Chain Reaction as stated earlier. For representing the individual position eg. 5th row 5th column, DNA sequence ATGCTACG is used which is obtained by merging ATGC (5th row DNA sequence) and TACG (5th column DNA sequence). The following table gives the DNA sequence of each and individual position after merging.

TABLE III

DNA STRANDS FOR POSITIONING AFTER MERGING

	ATGC	TGCA	GCAT	CATG	TACG	ACGT	CGTA	GTAC
TACG	TACGATGC	TACGTGCA	TACGGCAT	TACGCATG	TACGTACG	TACGACGT	TACGCGTA	TACGGTAC
ACGT	ACGTATGC	ACGTTGCA	ACGTGCAT	ACGTCATG	ACGTTACG	ACGTACGT	ACGTCGTA	ACGTGTAC
CGTA	CGTAATGC	CGTATGCA	CGTAGCAT	CGTACATG	CGTATACG	CGTAACGT	CGTACGTA	CGTAGTAC
GTAC	GTACATGC	GTACTGCA	GTACGCAT	GTACCATG	GTACTACG	GTACACGT	GTACCGTA	GTACGTAC
ATGC	ATGCATGC	ATGCTGCA	ATGCGCAT	ATGCCATG	ATGCTACG	ATCCACGT	ATGCCGTA	ATGCGTAC
TGCA	TGCAATGC	TGCATGCA	TGCAGCAT	TGCACATG	TGCATACG	TGCAACGT	TGCACGTA	TGCAGTAC
GCAT	GCATATGC	GCATTGCA	GCATGCAT	GCATCATG	GCATTACG	GCATACGT	GCATCGTA	GCATGTAC
CATG	CATGATGC	CATGTGCA	CATGGCAT	CATGCATG	CATGTACG	CATGACGT	CATGCGTA	CATGGTAC

We apply the following Heuristic algorithm which was experimented previously with silicon-based computing. Except number of queens, N=2 or 3 the following algorithm can be applied for others. For 2 queens and 3 queens placement on 2*2 and 3*3 chessboard respectively, there is no solution as described below.

Resultant DNA strands	Complementary DNA Strands	Position for placing 8 queens after megrging

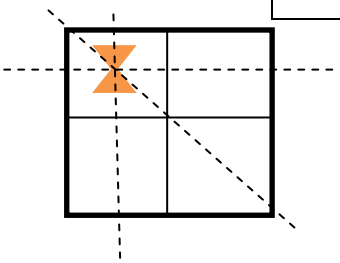


Fig 2: Placement restriction for second queen.

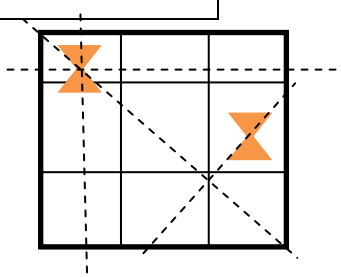


Fig 3: Placement restriction for third queen

Heuristic Algorithm:

1. Initially the sample N sequences are taken in T_0 which represents row values.
2. Let 'N' be the number of queens which should be placed on N*N chess board.
3. Polymerase T_0 to produce complementary strands which represent column values.
4. For step5 to 7 use original sample DNA sequences not the complementary ones.
5. If N%6!=2 or 6, then gel electrophoresis is used to separate all strands by length 4. Merge all 4 even (2,4,6,8) DNA sequences, with all 4 odd (1,3,5,7) DNA sequences.
6. If N%6==2, then cut 3rd strand of odd sequence and substitute it before 1st strand of odd sequence and extract 5th strand of the odd sequence and append it to the last of the odd sequence.
7. If N%6==3, then append the 2nd strand of even sequence to the end of even pair sequence and append 1 and 3 strands to the end of odd sequence.
8. Append odd sequence strands to the end of even sequence strands.
9. Merge the resultant DNA strands and complementary DNA strands value to find the positions of queen.

VII. RESULTS AND CONCLUSION

For placing 8 queens on the 8*8 chessboard, if above algorithm is applied for the TABLE II, we get the following results. N is considered as 8. Since 8%6 is 2, then steps 6,8,9 will be performed. Resultant sequences in each step are listed in the following tables.

Original Sequences

Step 6: extract 5th strand and append it to end of odd sequence

Strand Numbers	Odd Sequence	Strand Numbers	Even Sequence
1	TACG	2	ACGT
3	CGTA	4	GTAC
5	ATGC	6	TGCA
7	GCAT	8	CATG

Step 6: Cut 3rd of odd list substitute before 1st strand.

Step 8 : Append odd list to end of even list

Strand Numbers	Odd Sequence	Strand Numbers	Even Sequence
3	CGTA	2	ACGT
1	TACG	4	GTAC
5	ATGC	6	TGCA
7	GCAT	8	CATG

Strand Numbers	Odd Sequence	Strand Numbers	Even Sequence
3	CGTA	2	ACGT
1	TACG	4	GTAC
7	GCAT	6	TGCA
5	ATGC	8	CATG

Strand Numbers	Resultant Sequence
2	ACGT
4	GTAC
6	TGCA
8	CATG
3	CGTA
1	TACG
7	GCAT
5	ATGC

Step 9 : Merge resultant DNA strands and complementary DNA strands.

ACGT	ATGC	ACGTATGC
GTAC	TGCA	GTACTGCA
TGCA	GCAT	TGCAGCAT
CATG	CATG	CATGCATG
CGTA	TACG	CGTATACG
TACG	ACGT	TACGACGT
GCAT	CGTA	GCATCGTA
ATGC	GTAC	ATGCGTAC

Output: Result for placing 8 queens with respect to above table.

	ATGC	**TGCA**	**GCAT**	**CATG**	**TACG**	**ACGT**	**CGTA**	**GTAC**
TACG						TACGACGT		
ACGT	ACGTATGC							
CGTA					CGTATACG			
GTAC		GTACTGCA						
ATGC								ATGCGTAC
TGCA			TGCAGCAT					
GCAT							GCATCGTA	
CATG				CATGCATG				

Below is the table listing Number of queens, total and unique solutions available and its corresponding execution time when executed using silicon based computers and DNA computers.

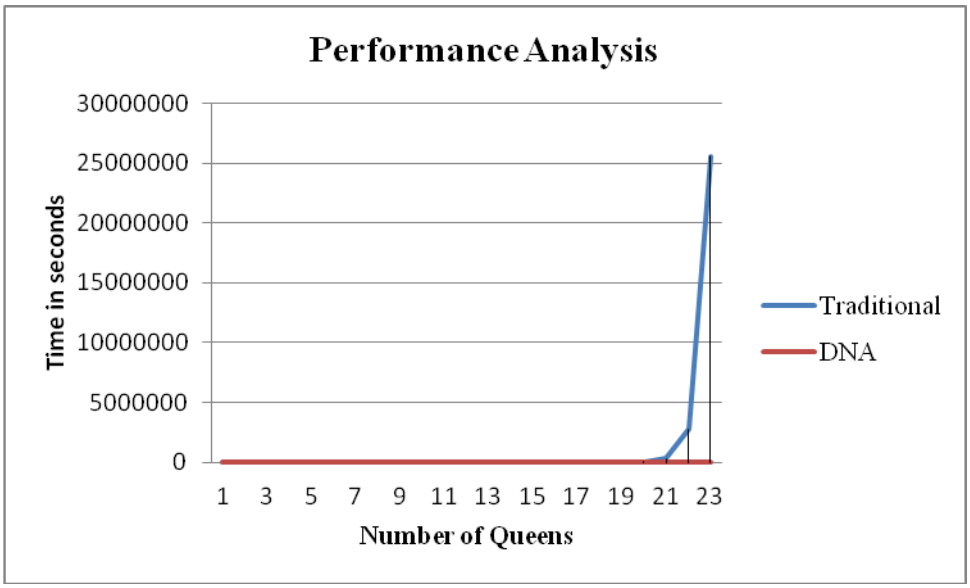
TABLE IIIII EXISTING SOLUTIONS FOR N QUEEN

Number of Queens	**Solutions**		**Execution Time using silicon based computers (in seconds)**	**Execution Time using DNA computers (in seconds)**
	Total	**Unique**		
1	1	1	Negligible	Negligible
2	0	0	Negligible	Negligible
3	0	0	Negligible	Negligible
4	2	1	Negligible	Negligible
5	10	2	Negligible	Negligible
6	4	1	Negligible	Negligible
7	40	6	Negligible	Negligible
8	92	12	Negligible	Negligible
9	352	46	Negligible	Negligible
10	724	92	Negligible	Negligible
11	2,680	341	Negligible	Negligible
12	14,200	1,787	Negligible	Negligible
13	73,712	9,233	Negligible	Negligible
14	365,596	45,752	0.2	Negligible
15	2,279,184	285,053	1.9	Negligible
16	14,772,512	1,846,955	11.2	Negligible
17	95,815,104	11,977,939	77.2	Negligible
18	666,090,624	83,263,591	576	Negligible
19	4,968,057,848	621,012,754	4500	Negligible
20	39,029,188,884	4,878,666,808	36720	Negligible
21	314,666,222,712	39,333,324,973	313920	Negligible

22	2,691,008,701,644	336,376,244,042	2756160	Negligible
23	24,233,937,684,440	3,029,242,658,210	25574400	Negligible
24	227,514,171,973,736	28,439,272,956,934	?	Negligible
25	2,207,893,435,808,352	275,986,683,743,434	?	Negligible
26	22,317,699,616,364,044	2,789,712,466,510,289	?	Negligible

In the above table the computational complexity is measured by using execution time. For obvious reasons, execution time increases with the increase in problem complexity. From the table, one can infer that the time is not estimated from 24 to 26 queens. More surprisingly, for placing 23 queens in 23*23 chess board, the execution time is 296 days by an ordinary computer! So far the number of solutions for 25*25 and 26*26 is not found yet. It is highly possible to make a DNA computer to execute this N Queen's problem in a fraction of second with the help of its parallel processing ability taking the proof established by Adleman's experiment to HP problem. Also, the solution for 25*25 and 26*26 Queens Problem could be found.

Based on the table, a graph is plotted to compare and analyse the performance (execution time) of both the techniques. The graph compares the execution time (in seconds) of backtracking solutions in N Queens Problems for different values of N using traditional silicon computers and DNA computers. It clearly reveals the superiority of DNA computers over silicon computers in finding solution to any kind of problem owing to its high speed and incomparable storage capacity.



References

[1] Leonard M.Adleman, *Molecular Computation of Solutions to Combinatorial Problems*, Science, Vol. 266, pages 1021-1024; November 11, 994.

[2] David K.Gifford, *On the path to Computation with DNA,* Science, Vol.266, pages 993-994; November 11, 1994.

[3] Richard J.Lipton, *DNA Solution of Hard Computational Problems,* Vol.268, pages 542-545; April 28, 1995.

[4] Leonard M.Adleman, *Computing with DNA*, Scientific American, August.1998.

[5] Ahrabian.H, Mirzaei.A, Nowzari-Dalini.A, *A DNA sticker Algorithm for Solving N-Queen Problem,* International Journal of Computer Science and Applications, Vol. 5, No.2, pages 12-22, 2008.

[6] Kashif Hammed, *DNA Computation Based Approach for Enhanced Computing Power,* International Journal of Emerging Sciences ISSN:2222-4254(1) April 2011.

[7] Ramin Maazallahi , Aliakbar Niknafs, *A Modified Dna Computing Approach To Tackle The Exponential Solution Space Of The Graph Coloring Problem*, International Journal in Foundations of Computer Science & Technology, Vol.3, No. 2, March 2013.

