I.J. Computer Network and Information Security, **2011, 3, 19-25**
Published Online April 2011 in MECS (http://www.mecs-press.org/)

An Algebraic Method for the N-Queens Problem Based on Permutation Operation Group

Jun Zhang[a, b]
[a]State Key Laboratory of Digital Manufacturing Equipment and Technology of China, Huazhong University of Science and Technology, Wuhan 430074, PR China
[b]Hubei Province Key Laboratory of Intelligent Robot, Wuhan Institute of Technology, Wuhan 430073, PR China
Email: wity@263.net

Zili Zhang
Shenzhen Graduate School, Harbin Institute of Technology, Shenzhen, 518055, PR China
Email: zhzili@gmail.com

***Abstract*—to analyze N-Queens problem in permutation space, this paper defines isomorphic operations of permutation to dihedral group D_4. With these operations to find elements within an orbit, two operations on orbits are also defined to generate new orbit from existing ones. Orbit signature is proposed to uniquely identify different orbits in orbit space. A search algorithm based on orbit signature is presented, and finally the effectiveness of the algorithm is illustrated by an example.**

***Index Terms*—N-Queens problem; Permutation operations; Orbit operations; Orbit signature; Search algorithm**

I. INTRODUCTION

N-Queens problem is a traditional but open issue in computer discipline; many literatures have been devoted to algorithms to find solutions for N-Queens problem [1]-[15]. N-Queens problem is a test bed of efficient search algorithms. It can also be used in parallel memory storage schemes, VLSI testing, traffic control and deadlock prevention. In communication field, the solutions are used to arrange channels of communication for conflict free access and maximum efficiencies of communication. Besides, the problem is also used for encoding schema for *Q*-matrix LPDC (Low Density Parity Code). One advantage of LPDC is its high performance of parameter design of code length and code rate.

There are two variants of N-Queens problem, regular N-Queens problem, and toroidal N-Queens problem. The difference between regular one and toroidal one is that the latter uses extended diagonals to avoid attacks of queens by modular arithmetic. In this paper, analyses mainly focus on the regular N-Queens problem, which is simply referred to as N-Queens problem, or just as order N in the following sections.

A recent paper [6] gives a survey of all known results of N-Queens problem. Approaches to solve N-Queens problem are basically geometric ones and algebraic ones. The geometric ways treat N-Queens problem as 2-dimensional chessboard, queens as points on the square. Searching algorithms use one or two dimensional arrays to simulate positioning, attacking of queens [4], [5], [8]-[9]. These algorithms are always intuitionistic but memory consuming. The algebraic ways need formulating the problem by arithmetic methods, such as constraint satisfaction problem, integer programming, and permutation generation problem [2], [3], [10]-[16]. Algebraic approaches offer different perspectives and novel methods of N-Queens problem [15]. At most time, algorithms synthesizing geometric and algebraic analysis are more widely used. However, among these methods two works are instructive to this article. One is the paper [10] which gives a method to find non-isomorphic solution based on less than relation of permutation. The other is paper [13] which gives optimal search based on group actions for N-Queens problem.

This paper will give some novel results by analyzing solutions of N-Queens problem based on permutation space. The analyses integrate geometric approaches, such as dihedral group D_4, and algebraic approaches, such as group action, isomorphism, orbit and generator, but main contributions of this paper are presented in algebraic form. The remainder of this paper is organized as follows. To analyze permutations as solutions of N-Queens problem, this paper firstly defines isomorphic operations of D_4 in permutation spaces in section 2. By these operations, other solutions isomorphic to an existing solution can be obtained easily. Except operations among elements within an orbit, Section 3 proposes two operators to generate new orbits from existing orbits. Section 4 presents a method to distinguish a newly generated orbit from the existing one by the concept of orbit signature. Section 5 gives an algorithm to search solutions based on orbit signature, also an example as the application of the algorithm is presented.

II PERMUTATION OPERATION GROUP PO AND ITS ISOMORPHIC MAP

For a permutation $\begin{pmatrix} 1 & 2 & ...,i,... & N \\ p_1 & p_2 & ...,p_i,... & p_N \end{pmatrix}$, briefly marked as ($p_1$, p_2,…, p_i,…, p_N) in the following, p_i is row index of a queen in column *i*, and $p_1p_2…p_\mathrm{i}…p_\mathrm{n}$ is an arrangement of the sequence from 1 to *N*. Sometimes p_i is called permutation element, *i* permutation subscript. The inherent advantages to represent solutions as permutations lie in that collisions of queens from the same row or the same column can never occur, and the only thing to do is to check whether there are diagonal collisions of queens. Diagonal collisions include attacks on positive diagonal (or sum diagonal) and negative diagonal (or difference diagonal), both of them can be expressed as follows (1):

$$(p_i\text{-}p_j)*(p_i\text{-}p_j)\neq(i\text{-}j)*(i\text{-}j),\ i\neq j. \qquad (1)$$

When a permutation is qualified as solution of N-Queens problem, can it be used to generate other solutions? A geometric thinking will solve this question.

For a $N\times N$ chessboard, any non-attacking layout of queens in the board can be exerted some geometric symmetric transformations to produce other layouts, which are qualified as solutions of N-Queens problem too. These geometric transformations form Dihedral group D_4[17]. Eight elements of D_4 include: g_1(identity map), g_2(reflection along negative diagonal), g_3(reflection along positive diagonal), g_4(vertical reflection along horizontal center line), g_5(horizontal reflection along vertical center line), g_6(rotation through $\pi/2$), g_7(rotation through π), g_8(rotation through $3\pi/2$). All rotations mentioned here are counterclockwise. Generally speaking, fixed points under all these 8 transformation are very few. So for most cases, obtaining one layout of queens means getting at most eight solutions simultaneously.

To analyze N-Queens problem in permutation space, some permutation operations similar to transformations in D_4 must be found. The following will define three basic permutation operations first, then give all the eight operations corresponding to eight symmetric elements of D_4, finally discuss the isomorphism of permutation operation and D_4.

Three basic permutation operations are defined as follows.

(1)Define operation *Inv*(*p*) to get inverse of permutation *p*. The operation swaps every element p_i with its subscript *i* of *p* firstly, then sorts new permutation elements by its new subscript ascending. The following example illustrates computation of inverse operation *Inv*.

$$\begin{pmatrix}1&2&3&4&5\\1&3&5&2&4\end{pmatrix}\text{ à }\begin{pmatrix}1&3&5&2&4\\1&2&3&4&5\end{pmatrix}\text{ à }\begin{pmatrix}1&2&3&4&5\\1&4&2&5&3\end{pmatrix}$$

(2)Define operation *Cmpl*(*p*) to get complement of permutation *p*, the complement is based on *N*+1. For every element p_i of *p*, substitute it with value *N*+1-p_i. The following example illustrates computation of complement operation *Cmpl*.

$$\begin{pmatrix}1&2&3&4&5\\1&3&5&2&4\end{pmatrix}\text{ à }\begin{pmatrix}1&2&3&4&5\\6-1&6-3&6-5&6-2&6-4\end{pmatrix}\text{ à }$$

$$\begin{pmatrix}1&2&3&4&5\\5&3&1&4&2\end{pmatrix}$$

(3)Define operation *Rev* (*p*) to get reverse of permutation *p*. The operation just reverses all elements p_i of *p*. The following example illustrates computation of reverse operation *Rev*.

$$\begin{pmatrix}1&2&3&4&5\\1&3&5&2&4\end{pmatrix}\text{ à }\begin{pmatrix}1&2&3&4&5\\4&2&5&3&1\end{pmatrix}$$

Multiplication of these three operations complies with the composition rule. Obviously, for operation *Inv*, *Cmpl* and *Rev*, twice consecutive operations will make a permutation revert, that is to say, when *f* refers to one of them, *p* any permutation, there exists $f^2(p)=p$.

Based on the three basic permutation operations, the eight operations for permutation can be defined following symmetric operations g_i(i=1, 2,…, 8) of D_4, they are denoted in the form of po_i(i =1, 2,…, 8).

(1)For identity map g_1, the identity permutation *e*(1, 2, 3,…, *N*) is just the identity operation po_1 of permutation, for any permutation *p*, $po_1(p)=e(p)=p$.

(2)For reflection g_2, the equivalent permutation operation po_2 is *Inv*, the operation to get inverse of a permutation, for any permutation *p*, $po_2(p)=Inv(p)$.

(3)For vertical reflection g_4, the equivalent permutation operation po_4 is *Cmpl*, the operation to get complement of a permutation, for any permutation *p*, $po_4(p)=Cmpl(p)$.

(4)For horizontal reflection g_5, the equivalent permutation operation po_5 is *Rev*, the operation to get reverse of a permutation, for any permutation *p*, $po_5(p)=Rev(p)$.

(5)For reflection g_3, the equivalent permutation operation po_3 can be obtained by applying po_2 firstly, then applying po_4, finally po_5, for any permutation *p*, $po_3(\text{p})=Rev(Cmpl(Inv(p)))$.

(6)For rotation g_6, its equivalent permutation operation po_6 can be obtained by applying po_2 firstly, then applying po_4, for any permutation *p*, $po_6(p)=Cmpl(Inv(p))$.

(7)For rotation g_7, its equivalent permutation operation po_7 can be obtained by applying po_4 and po_5 successively by any sequence due to commutatively of them, for example, for any permutation *p*, $po_7(p)=Rev(Cmpl(p))$.

(8)For rotation g_8, its equivalent permutation operation po_8 can be obtained by applying po_4 firstly, then applying po_2, for any permutation *p*, $po_8(p)=Inv(Cmpl(p))$.

In fact, most of the operations can be implemented by reusing other operations, such as $po_6(p)=Cmpl(po_2)$, $po_3(p)=Rev(po_6)$, $po_7(p)=Rev(po_4)$, $po_8(p)=Inv(po_4)$.

Multiplication of any two permutation operations is given as Table 1.

Table 1: Multiplication of permutation operations (elements of *PO*)

	po_1	po_2	po_3	po_4	po_5	po_6	po_7	po_8
po_1	po_1	po_2	po_3	po_4	po_5	po_6	po_7	po_8
po_2	po_2	po_1	po_7	po_6	po_8	po_4	po_3	po_5
po_3	po_3	po_7	po_1	po_8	po_6	po_5	po_2	po_4
po_4	po_4	po_8	po_6	po_1	po_7	po_3	po_5	po_2
po_5	po_5	po_6	po_8	po_7	po_1	po_2	po_4	po_3
po_6	po_6	po_5	po_4	po_2	po_3	po_7	po_8	po_1
po_7	po_7	po_3	po_2	po_5	po_4	po_8	po_1	po_6
po_8	po_8	po_4	po_5	po_3	po_2	po_1	po_6	po_7

It can be verified that permutation operation po_i(i=1, 2,…, 8) satisfies four axioms of group, so they form a group, denoted by Permutation Operation Group *PO*. (1)Closure. It is clear that product of any two elements po_i and po_j of *PO* still lies in the set, so multiplication of any two elements from *PO* is close. (2) Associatively. For any three elements po_i, po_j and po_k (not necessarily distinct) of *PO*, $(po_i\ po_j)\ po_k = po_i\ (po_j\ po_k)$. (3) Identity. The element po_1 is just the one. For any element po_i of *PO*, $po_i\ po_1 = po_i = po_1\ po_i$. (4) Inevitability. Inverse of identity element po_1 is itself. For reflection operation, inverse of po_i (i=2, 3, 4, 5) is themselves respectively. For rotational operation, $po_6po_8=po_1$, $po_7\ po_7=po_1$, so $po_6^{-1}=po_8$, $po_8^{-1}=po_6$, and $po_7^{-1}=po_7$. So inverse of element po_i still belongs to *PO*.

From definitions of permutation operations, it can be inferred that D_4 and *PO* are isomorphic groups. Define the map $g_i \rightarrow po_i$(i=1,2,…,8), and denote the map as ψ: $D_4 \rightarrow PO$. Clearly, map ψ is bijection. Define the combination manner of elements from D_4 and *PO*: if $g_i \rightarrow po_i$, $g_j \rightarrow po_j$, then $g_ig_j \rightarrow po_ipo_j$. So $\psi(g_ig_j)=po_ipo_j=\psi(g_i)\psi(g_j)$, for all $g_i, g_j \in D_4$. Thus D_4 and *PO* are isomorphic groups.

Isomorphism of D_4 and *PO* ensures completeness of permutation operations. For N-Queens problem, all solutions are in the form of permutation, so group *PO* is the right tool to operate on permutation space. When a solution is found, action of *PO* on it will generate some other solutions. Moreover, from points of view of group action, these solutions are on the same orbit, and the first solution is generator of the orbit. So action of *PO* will find all elements on an orbit.

III ORBIT OPERATIONS TO GENERATE NEW ORBITS

Permutation operation group *PO* generates all elements on an orbit from the generator, which means finding a solution equals finding the orbit, and altering an element on an orbit equals altering the orbit, so discussions can be focused on orbit space rather than permutation space. To operate on the orbit, some orbit operations must be defined on orbit space. The following will reach this by defining two operators. One is *Inc*, the operator to upgrade an orbit by increasing a queen to permutations of order *N* so as to get solutions of order *N*+1. The other is *Mov*, the operator to transmute an orbit by moving a queen within a permutation to get another solution. Both upgrade operation and transmutation operation should assure integrity of a permutation as the solution of N-Queens problem.

A. Definition of operator Inc

The operator *Inc* is defined to upgrade an orbit by increasing a queen to an solution, and its usage form is *Inc*(*Row*, *Col*), where *Row* and *Col* , the permutation element and permutation subscript respectively, indicate the position to place the new queen.

First discussions go to these two parameters. To avoid collision with existing queens, new columns can't be those existing ones. So the column to be added should be a virtual value between two columns, It is done by expressing *Col* in the form of a fraction with 0.5, and the newly generated column will be *Col*+0.5. For example, column 4.5 indicates that a new column will be added between column 4 and column 5, and that the new column will be column 5 if added successfully. By that means, column 0.5 indicates the left most column and the forthcoming column 1, and column 5.5 for *N*=5 indicates the right most column and a will-be column 6. The similar discussions apply to parameter *Row*.

The operation *Inc*(*Row*, *Col*) can be decomposed into two actions denoted by *IncRow*(*Row*), to increase a new row marked as *Row*, and *IncCol*(*Col*), to increase a new column marked as *Col*. Separately, when implementing *IncRow*(*Row*), all rows bigger than *Row*, i.e. permutation elements between (*Row*, *N*], should be added 1. And when implementing *IncCol*(*Col*), all columns bigger than *Col*, i.e. permutation subscripts between (*Col*, *N*], should be added 1. However, to keep integrity of a solution, operation *IncRow* must accompany operation *IncCol* and vice versa. So operation *Inc*(*Row*, *Col*) should be done as follows: firstly apply operation *IncRow*(*Row*) to increase permutation elements in the range of (*Row*, *N*] by 1, secondly apply operation *IncCol*(*Col*) to increase permutation subscript in the range of (*Col*, *N*] by 1, then fill the blank position of permutation subscript *Col*+0.5 with a value *Row*+0.5 indicating the position of the newly added queen.

Some examples are given as follows to illustrate operation *Inc*. All of them increase a queen to the solution (2, 4, 1, 3) of *N*=4, and generate a solution of *N*=5.

(1) *Inc*(0.5, 0.5): increase a queen on the top left.

$$\begin{pmatrix}1&2&3&4\\2&4&1&3\end{pmatrix}\xrightarrow{Inc(0.5,0.5)}\boxed{\xrightarrow{IncRow(0.5)}\begin{pmatrix}1&2&3&4\\3&5&2&4\end{pmatrix}\xrightarrow{IncCol(0.5)}}\begin{pmatrix}1&2&3&4&5\\1&3&5&2&4\end{pmatrix}$$

(2) *Inc*(4.5, 4.5): increase a queen on the bottom right corner.

$$\begin{pmatrix}1&2&3&4\\2&4&1&3\end{pmatrix}\xrightarrow{Inc(4.5,4.5)}\boxed{\xrightarrow{IncRow(4.5)}\begin{pmatrix}1&2&3&4\\2&4&1&3\end{pmatrix}\xrightarrow{IncCol(4.5)}}\begin{pmatrix}1&2&3&4&5\\2&4&1&3&5\end{pmatrix}$$

(3) *Inc*(0.5, 4.5): increase a queen on the top right corner.

$$\begin{pmatrix}1&2&3&4\\2&4&1&3\end{pmatrix}\xrightarrow{Inc(0.5,4.5)}\boxed{\xrightarrow{IncRow(0.5)}\begin{pmatrix}1&2&3&4\\3&5&2&4\end{pmatrix}\xrightarrow{IncCol(4.5)}}\begin{pmatrix}1&2&3&4&5\\3&5&2&4&1\end{pmatrix}$$

(4) *Inc*(4.5, 0.5): increase a queen on the bottom left corner.

$$\begin{pmatrix}1&2&3&4\\2&4&1&3\end{pmatrix}\xrightarrow{Inc(4.5,0.5)}\boxed{\xrightarrow{IncRow(4.5)}\begin{pmatrix}1&2&3&4\\2&4&1&3\end{pmatrix}\xrightarrow{IncCol(0.5)}}\begin{pmatrix}1&2&3&4&5\\5&2&4&1&3\end{pmatrix}$$

(5) *Inc*(2.5, 2.5): increase a queen on the center.

$$\begin{pmatrix}1&2&3&4\\2&4&1&3\end{pmatrix}\xrightarrow{Inc(2.5,2.5)}\boxed{\xrightarrow{IncRow(2.5)}\begin{pmatrix}1&2&3&4\\2&5&1&4\end{pmatrix}\xrightarrow{IncCol(2.5)}}\begin{pmatrix}1&2&3&4&5\\2&5&3&1&4\end{pmatrix}$$

B. Definition of operator Mov

The operator *Mov* is defined to transmute an orbit by moving a queen within a solution, and its usage form is *Mov*(*OldRow*, *OldCol*, *NewRow*, *NewCol*), where (*OldRow*, *OldCol*) is the old position of the queen, (*NewRow*, *NewCol*) the new position. Note that for order *N*, values of *NewRow* and *NewCol* are virtual positions, but *OldRow* and *OldCol* are real position, and all of them are uniformly marked as the value before any changes to row and column. The operation can be done by two steps: to delete the queen at (*OldRow*, *OldCol*), to add a queen to (*NewRow*, *NewCol*).

To delete a queen, define operation *Del*(*Row*, *Col*) to do something contrary to operation *Inc*(*Row*, *Col*). The Operation can be done by two steps: to delete the queen from row by operation *DelRow*(*Row*), and to delete it from column by operation *DelCol*(*Col*). The three operators *Del*, *DelRow*, and *DelCol* can be defined following *Inc*, *IncRow*, *IncCol*. To realize *DelRow*(*Row*), decrease all permutation elements in the range of (*Row*, *N*] by 1. To realize *DelCol*(*Col*), first to remove permutation element at subscript *Col*, then decrease all permutation subscripts remained in the range of (*Col*, *N*] by 1. To realize *Del*(*Row*, *Col*), use *DelRow*(*Row*) firstly, *DelCol*(*Col*) secondly. The final result is that a solution of order *N* is changed to the solution of order *N*-1. The following example illustrates the operation of *Del* to delete a queen from the solution of *N*=5 to get a solution of *N*=4.

$$\begin{pmatrix}1&2&3&4&5\\2&5&3&1&4\end{pmatrix}\xrightarrow{Del(3,3)}\boxed{\xrightarrow{DelRow(3)}\begin{pmatrix}1&2&3&4&5\\2&4&3&1&3\end{pmatrix}\xrightarrow{DelCol(3)}}\begin{pmatrix}1&2&3&4\\2&4&1&3\end{pmatrix}$$

With operator *Inc* and *Del*, the operator *Mov* can be achieved easily. However, considering the relationship of new position (*NewRow*, *NewCol*) and old position (*OldRow*, *OldCol*), more attention should be paid to the sequence to do *Inc*(*NewRow*, *NewCol*) and *Del*(*OldRow*, *OldCol*), otherwise wrong modification of row and column by the former operation certainly results in a wrong parameter which definitely induces the latter operation to go wrong. Some useful tips are that the former operation should impact as less as possible on the position for the latter operation. The following examples give some explanations.

(1)to fulfill *Mov*(1, 1, 3.5, 3.5) within the solution (1, 3, 5, 2, 4) of *N*=5, perform *Inc*(3.5, 3.5) firstly, *Del*(1, 1) secondly.

$$\begin{gathered}\begin{pmatrix}1&2&3&4&5\\1&3&5&2&4\end{pmatrix}\xrightarrow{Move(1,1,3.5,3.5)}\\ \boxed{\xrightarrow{Inc(3.5,3.5)}\begin{pmatrix}1&2&3&4&5&6\\1&3&6&4&2&5\end{pmatrix}\xrightarrow{Del(1,1)}}\begin{pmatrix}1&2&3&4&5\\2&5&3&1&4\end{pmatrix}\end{gathered}$$

(2)to fulfill *Mov*(5, 5, 3.5, 3.5) within the solution (2, 4, 1, 7, 5, 3, 6) of *N*=7, perform *Del*(5, 5) firstly, *Inc*(3.5, 3.5) secondly.

$$\begin{gathered}\begin{pmatrix}1&2&3&4&5&6&7\\2&4&1&7&5&3&6\end{pmatrix}\xrightarrow{Move(5,5,3.5,3.5)}\\ \boxed{\xrightarrow{Del(5,5)}\begin{pmatrix}1&2&3&4&5&6\\2&4&1&6&3&5\end{pmatrix}\xrightarrow{Inc(3.5,3.5)}}\begin{pmatrix}1&2&3&4&5&6&7\\2&5&1&4&7&3&6\end{pmatrix}\end{gathered}$$

IV Orbit Signatures

While operating on orbit space, it is necessary to identify a new orbit. If the orbit holding a new solution already exists, then generation of the new solution is not necessary. So this is the question on orbit identification, whose goal is to give a global unique identity to any orbit, and the identity can be calculated by any element on the orbit. By doing so, when a solution is reached, first to calculate its global unique identity, then inquire its existence, if the identity is found, then abandon the new solution, otherwise retain it as the generator of a new orbit.

To identify an orbit, it means to give it a signature. Requirements of orbit signature include: (1) Uniqueness. For any order *N*, no orbit will have a duplicate signature. The Map from an orbit to its signature is a bijection. (2) Coincidence. Every solution on one orbit has identical structure, and the signature is a description of the inherent property, so every solution on the orbit should have unanimous value, which is used as orbit signature. (3) Easy-computation. For efficiency of search, calculation of orbit signature should be easily done, and some compromise on computation should be allowed.

To describe orbit inherence by its elements, which are obtained by operations similar to symmetric transformations, something invariant under these geometric operations must be found. A second thinking on D_4 will find that such transformations as reflections and rotations are all orthogonal, whose invariants include the distance between two points and the included angle between two sides formed by three points. For computing convenience, the included angle for two sides is substituted by the area of the three points. So distance and area are ideal parameters to depict internal structure of a geometric figure.

In any solution, every three adjacent queens construct a triangle. The signature of a triangle $Sig_{\triangle}$ is defined as a *Length*-*Area*-triple as (2)

$$Sig_{\triangle}=(Length_{\mathrm{side1}}, Area, Length_{\mathrm{side2}}) \tag{2}$$

where side1 and side2 are any two adjacent sides of the triangle, and $Length_{\mathrm{side1}}$ and $Length_{\mathrm{side2}}$ are their length respectively, and *Area* is the area of the triangle. For a triangle formed by three points $a(x_a, y_a)$, $b(x_b, y_b)$, and $c(x_c, y_c)$, the length of side $\overline{ab}$ is computed in (3), and the area of the triangle is computed in (4). A little deviation of length and area from their common definition is to leave them as integers for computing simplicity.

$$Length(\overline{ab})=(x_a-x_b)\times(x_a-x_b)+(y_a-y_b)\times(y_a-y_b) \tag{3}$$

$$Area(\overline{abc})=\left\|\begin{matrix}x_a & y_a & 1\\ x_b & y_b & 1\\ x_c & y_c & 1\end{matrix}\right\|=|(x_a-x_b)\times(y_c-y_b)-(x_c-x_b)\times(y_a-y_b)| \tag{4}$$

Clearly, if the signature of two triangles is the same, they are equivalent.

In a solution of order *N*, all queens form *N*-2 adjacent triangles. As for a set of *N* triangles, its signature $Sig_{\triangle s}$ is defined as (5).

$$Sig_{\triangle\mathrm{s}}=\{Sig_{\triangle\mathrm{i}}|\mathrm{i}=1, 2,\ldots, N\} \tag{5}$$

For any layout shown in fig. 1(a), there are two ways to form adjacent triangles. One is along x directions in fig. 1(b), by which any three points with continuous x value form a triangle. The other is along y directions in fig. 1(c), by which any three points with continuous y value form a triangle. These two ways give respectively two signatures, $Sig_{\triangle\mathrm{x}}$ and $Sig_{\triangle\mathrm{y}}$. Define the signature of a solution as (6).

$$Sig_{\mathrm{s}}=\{Sig_{\triangle\mathrm{x}}, Sig_{\triangle\mathrm{y}}\} \tag{6}$$

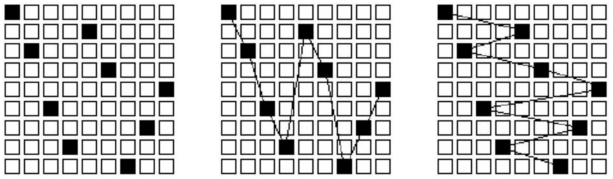
(a) the layout of a solution (b) along x direction (c) along y direction

fig. 1 Computation of a solution's signature

In permutation space, for any permutation *p*, the pair of permutation elements p_i and its subscript *i*, (i, p_i), is arranged by *i* ascendingly, so this form is fit for computing $Sig_{\triangle x}$, denoted as Sig_{px} here. According to definitions of (2) - (5), Sig_{px} of permutation *p* is calculated as (7).

$$Sig_{px}=\{(1+(p_{i+1}-p_i)^2, |p_{i+2}-2p_{i+1}+p_i|, 1+(p_{i+2}-p_{i+1})^2)|i=1, 2,\ldots, N-2\} \quad (7)$$

To calculate $Sig_{\triangle y}$, denoted as Sig_{py} here, it is necessary to get the pair (i, p_i) arranged by p_i ascendingly, which means to get inverse of *p*, so there exist (8)

$$Sig_{py}=Sig_{Inv(p)x} \quad (8)$$

By (7) and (8), the signature Sig_p of permutation *p* is calculated as (9).

$$Sig_p=\{Sig_{px}, Sig_{py}\} \quad (9)$$

Due to effects of symmetric transformations, there are sequential differences within the signatures of two permutations on the same orbit. So comparing criterion must be defined to recognize the same signatures of the orbit, it is Sequential-Reverse criterion.

(1)For two signatures $Sig_{\triangle 1}(Length_{11}, Area_1, Length_{21})$ and $Sig_{\triangle 2}(Length_{12}, Area_2, Length_{22})$ of the same triangle, they are equal if sequentially $Area_1=Area_2$, $Length_{11}=Length_{12}$, $Length_{21}=Length_{22}$, or reversely $Area_1=Area_2$, $Length_{11}=Length_{22}$, $Length_{12}=Length_{21}$.

(2)For two signatures $Sig_{\triangle s1}\{Sig_{\triangle 11},Sig_{\triangle 12},\ldots,Sig_{\triangle 1N}\}$ and $Sig_{\triangle s2}\{Sig_{\triangle 21},Sig_{\triangle 22},\ldots,Sig_{\triangle 2N}\}$of the same set of *N* triangles, they are equal if sequentially $Sig_{\triangle 11}=Sig_{\triangle 21}$, $Sig_{\triangle 12}=Sig_{\triangle 22}$, …, $Sig_{\triangle 1N}=Sig_{\triangle 2N}$, or reversely $Sig_{\triangle 11}=Sig_{\triangle 2N}$, $Sig_{\triangle 12}=Sig_{\triangle 2\ N-1}$, …, $Sig_{\triangle 1N}=Sig_{\triangle 21}$.

(3)For two signatures $Sig_{s1}\{Sig_{\triangle x1}, Sig_{\triangle y1}\}$ and $Sig_{s2}\{Sig_{\triangle x2}, Sig_{\triangle y2}\}$ of a solution, they are equal if sequentially $Sig_{\triangle x1}=Sig_{\triangle x2}$, $Sig_{\triangle y1}=Sig_{\triangle y2}$, or reversely $Sig_{\triangle x1}=Sig_{\triangle y2}$, $Sig_{\triangle y1}=Sig_{\triangle x2}$.

The comparing criterion applies to the signature of permutation too. Under those definition, signatures of permutations on one orbit of *N*=9 shown in table 2 are all the same.

Table 2: signatures of permutation on one orbit of *N*=8

permutation	signature
(1, 6, 8, 3, 7, 4, 2, 5)	{{(26, 3, 5), (5, 7, 26), (26, 9, 17), (17, 7, 10), (10, 1, 5), (5, 5, 10)}, {(37, 9, 10), (10, 5, 5), (5, 0, 5), (5, 8, 37), (37, 9, 10), (10, 5, 5)}}
(1, 7, 4, 6, 8, 2, 5, 3)	{{(37, 9, 10), (10, 5, 5), (5, 0, 5), (5, 8, 37), (37, 9, 10), (10, 5, 5)}, {(26, 3, 5), (5, 7, 26), (26, 9, 17), (17, 7, 10), (10, 1, 5), (5, 5, 10)}}
(6, 4, 7, 1, 3, 5, 2, 8)	{{(5, 5, 10), (10, 9, 37), (37, 8, 5), (5, 0, 5), (5, 5, 10), (10, 9, 37)}, {(10, 5, 5), (5, 1, 10), (10, 7, 17), (17, 9, 26), (26, 7, 5), (5, 3, 26)}}
(8, 3, 1, 6, 2, 5, 7, 4)	{{(26, 3, 5), (5, 7, 26), (26, 9, 17), (17, 7, 10), (10, 1, 5), (5, 5, 10)}, {(5, 5, 10), (10, 9, 37), (37, 8, 5), (5, 0, 5), (5, 5, 10), (10, 9, 37)}}
(5, 2, 4, 7, 3, 8, 6, 1)	{{(10, 5, 5), (5, 1, 10), (10, 7, 17), (17, 9, 26), (26, 7, 5), (5, 3, 26)}, {(37, 9, 10), (10, 5, 5), (5, 0, 5), (5, 8, 37), (37, 9, 10), (10, 5, 5)}}
(8, 2, 5, 3, 1, 7, 4, 6)	{{(37, 9, 10), (10, 5, 5), (5, 0, 5), (5, 8, 37), (37, 9, 10), (10, 5, 5)}, {(10, 5, 5), (5, 1, 10), (10, 7, 17), (17, 9, 26), (26, 7, 5), (5, 3, 26)}}
(4, 7, 5, 2, 6, 1, 3, 8)	{{(10, 5, 5), (5, 1, 10), (10, 7, 17), (17, 9, 26), (26, 7, 5), (5, 3, 26)}, {(5, 5, 10), (10, 9, 37), (37, 8, 5), (5, 0, 5), (5, 5, 10), (10, 9, 37)}}
(3, 5, 2, 8, 6, 4, 7, 1)	{{(5, 5, 10), (10, 9, 37), (37, 8, 5), (5, 0, 5), (5, 5, 10), (10, 9, 37)}, {(26, 3, 5), (5, 7, 26), (26, 9, 17), (17, 7, 10), (10, 1, 5), (5, 5, 10)}}

As can be seen, the signatures of all permutations on an orbit are all the same in the light of the definition of permutation signature and Sequential-Reverse criterion, so any one of them can be chosen to represent the signature of the orbit. Thus orbit signature is defined as the signature of any permutation on it. The global uniqueness of orbit signature is clear. If two orbits are from space of different order, whose numbers of *Length-Area*-triple are also different, so their signatures are definitely different. If two different orbits from the same space have the same signatures, this means their all *Length-Area*-triple are the same sequentially or reversely, that is to say, all adjacent triangles formed by every three adjacent queens are equivalent, so the layout of queens of these two permutations are totally the same, thus the two orbits must be the same. In a word, orbit signature gives a global unique identity to the orbit.

V Search algorithm based on Orbits

Based on orbit signature, an orbit-based search algorithm to solve N-Queens problem can be proposed as follows, and an example to demonstrate its effectiveness is given.

A. Search algorithm

The idea is realized by a two-step search. The first step is to apply operator *Inc* to upgrade all orbits of order *N*. The second step is to apply operator *Mov* to current orbits in space of order *N*+1.

Orbit-based search algorithm

```
procedure: search_orbits
begin
  //Step 1: use operator Inc to upgrade orbits from order N-1 to N
    for every solution on orbits in the space of order N-1
      for every position (0.5, 1.5, …, N-0.5) ×(0.5, 1.5, …, N-0.5)
      apply operator Inc to try to increase a queen;
      calculate orbit signature of the new solution;
      if the signature is unique, retain it and the orbit;
    end for
  end for
  //Step 2: use operator Mov to transmute current orbits
  repeat
    access orbits in the space of order $N$ by breadth-first strategy or depth-first
    for every solution on orbits in the space of order $N$
      for every queen in the solution
        for every position $(0.5, 1.5, \ldots, N+0.5) \times (0.5, 1.5, \ldots, N+0.5)$
          apply operator Mov to try to move a queen;
          calculate orbit signature of the new solution;
          if the signature is unique, retain it and the orbit;
        end for
      end for
    end for
  until all orbits have been extended by the search strategy
end
```

The positions of queens are given in the form of Descartes product. Advantages of the algorithm lie in that orbits can be obtained gradually from orbit space of order N-1 and N. Moreover, to obtain an orbit means to get at most eight solutions, which helps to promotes search efficiency for N-Queens problem.

B. *Case study*

Taking N=8 for example, the following illustrates generation of all orbits based on solutions of N=7. During the search, a lexicographical generation algorithm for permutation in STL is used. Numbers of permutations are given by sequence of their generation, and numbers of 12 orbits are given by sequence of their generator.

(1)Use operator *Inc* to upgrade orbits from N=7 to N=8.

Orbit 1: Apply *Inc*(7.5, 0.5) to (2, 4, 1, 7, 5, 3, 6), and gets a generator (8, 2, 4, 1, 7, 5, 3, 6).

Orbit 2: Apply *Inc*(0.5, 7.5) to (2, 4, 1, 7, 5, 3, 6), and gets a generator (3, 5, 2, 8, 6, 4, 7, 1).

Orbit 9: Apply *Inc*(6.5, 0.5) to (2, 4, 1, 7, 5, 3, 6), and gets a generator (7, 2, 4, 1, 8, 5, 3, 6).

There are three orbits generated in the first step, which set up the basis of the second step.

(2)Use operator *Mov* to transmute orbits of N=8.

Orbit 6: Apply *Mov*(3, 1, 2.5, 8.5) to (3, 6, 4, 2, 8, 5, 7, 1), solution 22 on orbit 1, and gets a generator (6, 4, 2, 8, 5, 7, 1, 3), i.e. solution 77 on orbit 6.

Orbit 8: Apply *Mov*(4, 1, 3.5, 8.5) to (4, 2, 7, 3, 6, 8, 5, 1), solution 33 on orbit 1, and gets a generator (2, 7, 3, 6, 8, 5, 1, 4), i.e. solution 10 on orbit 8.

Orbit 7: Apply *Mov*(7, 1, 8.5, 3.5) to (7, 2, 6, 3, 1, 4, 8, 5), solution 83 on orbit 8, and gets a generator (2, 6, 8, 3, 1, 4, 7, 5), i.e. solution 9 on orbit 7.

Orbit 11: Apply *Mov*(5, 3, 2.5, 0.5) to (4, 8, 5, 3, 1, 7, 2, 6), solution 46 on orbit 7, and gets a generator (3, 5, 8, 4, 1, 7, 2, 6), i.e. solution 17 on orbit 11.

Orbit 3: Apply *Mov*(4, 6, 3.5, 1.5) to (4, 8, 5, 3, 1, 7, 2, 6), solution 9 on orbit 7, and gets a generator (3, 5, 8, 4, 1, 7, 2, 6), i.e. solution 5 on orbit 3.

Orbit 5: Apply *Mov*(8, 5, 2.5, 0.5) to (5, 7, 1, 3, 8, 6, 4, 2), solution 57 on orbit 3, and gets a generator (6, 8, 2, 4, 1, 7, 5, 3), i.e. solution 23 on orbit 5.

Orbit 4: Apply *Mov*(1, 2, 8.5, 1.5) to (6, 1, 5, 2, 8, 3, 7, 4), solution 65 on orbit 3, and gets a generator (5, 8, 4, 1, 7, 2, 6, 3), i.e. solution 64 on orbit 4.

Orbit 12: Apply *Mov*(7, 2, 1.5, 7.5) to (2, 7, 5, 8, 1, 4, 6, 3), solution 11 on orbit 9, and gets a generator (3, 6, 8, 1, 5, 7, 2, 4), i.e. solution 24 on orbit 12.

Orbit 10: Apply *Mov*(4, 5, 2.5, 0.5) to (5, 2, 8, 1, 4, 7, 3, 6), solution 53 on orbit 12, and gets a generator (3, 5, 2, 8, 1, 7, 4, 6), i.e. solution 14 on orbit 10.

Nine other orbits are generated gradually as above. The path to generate all 12 orbits of N=8 is shown in fig. 2. The trajectory of orbit generation forms a graph. However, generator of an orbit can be other solutions besides the one just used above, and sequence to generate all orbits varies in the light of different search strategies.

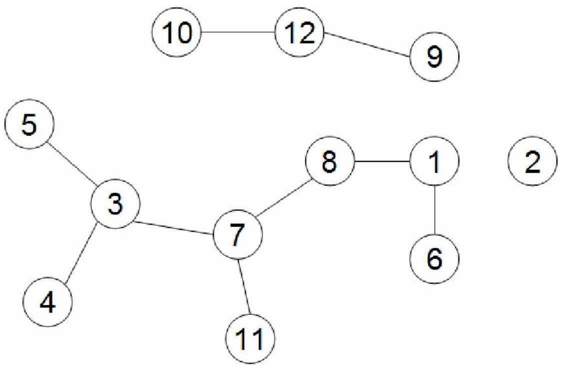

Fig. 2 One trajectory graph of orbit generation of N=8

VI Conclusion

In this paper, an algorithm to search solutions of N-Queens problem is proposed based on the orbit and group actions.

The main idea is composed of 3 steps. Step 1: to get a solution p by any algorithm [3]. Step 2: Apply permutation operation group on p to get the other solutions on the same orbit o. Step 3: Apply orbit operations on o to get the other orbits. Then all permutation elements on all orbits are all solutions of N-Queens problem.

To identify different orbits, this paper brings forward the concept of the orbit signature which gives every orbit a global unique identity. To operate on the orbit, this paper defines two operators, *Inc* and *Mov*, to upgrade and transmute the orbits respectively. To operate on elements on the orbit, this paper gives an isomorphism from D_4 to permutation operation group *PO*. All the discussions utilize the theory of orbits, action of the group, and permutation.

There are two topics worth consideration in the future. The first one may focus on promotion of the computing efficiency and a faster implementation of the algorithm. The second one may focus on the optimal path to generate all orbits.

ACKNOWLEDGEMENTS

This work is supported by the National Key Basic Research Program of China under Grand 2005CB724103.

REFERENCES

[1] I. Rivin, I. Vardi, P. Zimmermann. The n-queens problem, Amer. Math. Monthly 101 (1994)629-639.

[2] J.S. Rohl, A faster lexicographical n-queens algorithm. Inform. Process. Lett. 17(1983)231-233.

[3] Rok Sosic, Jun Gu. A polynomial time algorithm for the N-Queens problem. SIGART Bulletin, 1(3)(1990)7-11.

[4] Sosich R., Gu J. Fast search algorithms for the N-queens problem. IEEE Transactions on System, Man, and Cybernetics 21(6)(1991)1572-1576.

[5] K. Kise, T. Katagiri, H. Honda, T. Yuba. Solving the 24-queens problem using MPI on a PC cluster. Technical Report UEC-IS-2004-6, Graduate School of Information Systems, The University of Electro-Communication, June 2004.

[6] Jordan Bell, Brett Stevens. A survey of known results and research areas for n-queens. Discrete Mathematics 309(2009) 1-31.

[7] S.P. Nudelman. The modular n-queens problem in higher dimensions. Discrete Math. 146(1-3) (1995)159-167.

[8] M. Vasquez. New result on the queens n2 graph coloring problem. J. Heuristics, 10(2004)407-413.

[9] Zhang Dakun. One eighth rule in N-queens problem based on group theory and morphological gene combinations. International Forum on Information Technology and Applications (2009) 651-654.

[10] P. Cull, R. Pandey. Isomorphism and the n-queens problem. SIGCSE Bull. 26(3)(1994) 29-36.

[11] A.P. Burger, C.M. Mynhardt. An upper bound for the minimum number of queens covering the n×n chessboard. Discrete Applied Math. 121(2002)51-60.

[12] A.P. Burger, C.M. Mynhardt. An improved upper bound for queens domination numbers. Discrete Math. 266(2003)119-131.

[13] Matthias R. Engelhardt. A group-based search for solutions of the n-queens problem. Discrete Mathmematics 307(2007) 2535-2551.

[14] Zsuzsanna Szaniszlo, Maggy Tomova, Cindy Wyels. The N-queens problem on a symmetric Toeplitz matrix. Discrete Math. 309(2009)969-974.

[15] C. Erbas, S. Sarkeshik, M.M. Tanik. Different perspectives of the n-queens problem. in: Proceedings of the 1992 ACM Annual Conference on Communication, ACM Press, 1992, pp.99-108.

[16] A. Panayotopoulos. Generating stable permutations. Discrete Math. 62(2) (1986) 219-221.

[17] M.A. Armstrong. Groups and symmetry. Springer-Verlag New York Inc., 1988.

Jun Zhang is a PhD student in Huazhong University of Science and Technology. He received his MS degree of major of mechatronics from School of Mechanics Science and Engineering, Huazhong University of Science and Technology in 2002. His research interests are in system dynamics of digital manufacturing.

Zili Zhang is a PhD student in Harbin Institute of Technology. He received his MS degree of technology of computer application from Harbin University of Science and Technology. His research interests are in 3D reconstruction and graphic recognition.