Integer programming models for 3-dimensional Xingdu problem

Affiliation: Class 13, Grade 9

Tsinghua High School, China (中国、清华大学附属中学)

Authors： Xingshen Huang （黄省身）

Chuyao Peng（彭楚尧）

Xinyi Xiong（熊心宜）

Mentor: Qingming Yang（杨青明）

Integer programming models for 3-dimensional Xingdu problem

Abstract—Xingdu is an emerging mathematical game about searching for a corresponding polyline for a given polyline in a 2-dimensional or 3-dimensional mesh grid, meanwhile the searched polyline should have the same start and end points as those of the given polyline and the corresponding line segments on the searched and given polylines are perpendicular each other respectively. Studying from mathematical perspective the problem solving and designing of Xingdu is helpful of finding a general method for Xingdu problem solving, but still not attract much more attentions due to the relatively short history of Xingdu. Referring to the achievements in mathematical methods for Sudoku, we investigated preliminarily the possibility of using optimization theory to solve the problem of Xingdu. According to the definition of Xingdu, the points on the given polyline and the solution polyline have to be the grid nodes in the mesh grid so that their coordinates are integer. In addition, there should not be any points appearing repeatedly and any three sequential points on the solution polyline cannot be located on a same line. Due to these constraints, the optimization problem related to Xingdu belongs to integer programming with nonlinear constraints, which is very difficult and usually can only be solved by enumerating. For this reason, we solve the related problem first by neglecting the nonlinear constraints so that it can be converted as an integer programming with linear constraints only. Afterwards, we can check whether it fulfill the related nonlinear constraints if a feasible solution is obtained. Based on that, we proposed two integer programming models for Xingdu problem, which are the model using only line segments perpendicular property as constraints and the binary integer programming model using line segments perpendicular property together with no repeated points as constraints. The test results show that the proposed binary integer programming model has a very good performance in finding the solution of Xingdu problem. We also discussed the possibility of using the proposed binary integer programming model together with random simulation to design a Xingdu problem. Due to the challenge of discriminating the uniqueness of Xingdu solution, we finally presented an approach, inspired from the branch and bound method for integer programming, which solves and designs the Xingdu problem by constructing and pruning a search tree.

Keywords—3-dimensional Xingdu; integer programming; binary integer programmming; search tree

Background:
Study the Xingdu problem from the perspective of mathematics and try to find a representative and effective method for Xingdu problem solving and designing.

Highlight:
The paper proposed a binary integer programming model and a search tree based method for Xingdu problem solving, which provides a possibility of solving Xingdu problem efficiently and representatively from the perspective of mathematics.

1. Introduction

CG graph and Xingdu are emerging mathematical games about searching for a corresponding polyline for a given polyline in a 2-dimensional or 3-dimensional mesh grid, meanwhile the searched polyline should have the same start and end points as those of the given polyline and the corresponding line segments on the searched and given polylines are perpendicular each other, respectively. CG graph and Xingdu was introduced by Yang and Xu in 2011[1][2]. 3-dimensional CG graph and 3-dimensional Xingdu were introduced in the same year [3]. Figure 1 intuitively depicts a Xingdu in 3-dimensional mesh grid. The red polyline is the given polyline, and the blue polyline is the searched polyline. And corresponding line segments on the given polyline in red and the searched polyline in blue are perpendicular each other.

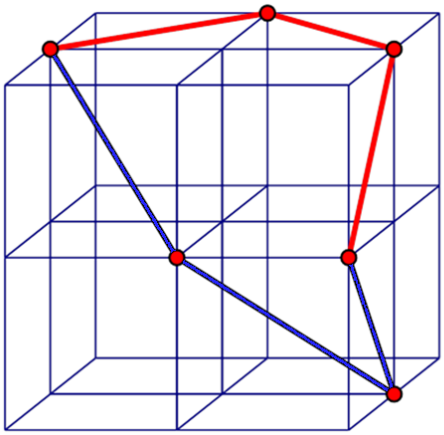


Figure 1. An example of 3-dimensional Xingdu

A number researchers have studied and discussed the properties with CG graph and Xingdu since their emerge [4][5]. On the other hand, due to the relatively short history of CG graph and Xingdu, solving and designing CG graph or Xingdu problems from perspective of mathematics have not widely drawn attentions yet.

Compared with CG graph and Xingdu, Sudoku is of a relatively long history and also more popular as a puzzle game. Recently, a number of scholars have systematically studied Sudoku as a mathematical problem [6]-[9]. In particular, remarkable achievements have been made on solving Sudoku problem by using integer programming or constrained optimization [8] [10]-[12]. For example, Bartlett reported an effective way by converting Sudoku problem as an integer programming or binary integer programming [12].

Currently, there are a couple of methods for Xingdu problem solving, including the method based on unique determination, the method based on perpendicular plane, and the start-end point successive approximation method[1][4]. The basics of these methods are finding the solution of a Xingdu problem by manually drawing lines with the help of necessary calculations. Unfortunately, with the increase of mesh grid and Xingdu problem scale, these methods become more and more impracticable, particularly when we extend the mesh grid from 2 dimensional into 3 dimensional. From this point of view, to find a universe and high

effective method for Xingdu problem solving has been being attractive.

Referring to the achievements in solving Sudoku from mathematical perspective by using optimization theory, the motivation of this piece of work is to study Xingdu problem and try to find a universe and high effective method for Xingdu problem solving and designing.

2. Problem formulation for 3-dimensional CG graph and Xingdu

In this paper, we focus on 3-dimensional CG graph and Xingdu problem. 2-dimensionla CG graph and Xingdu can be considered as a special case of 3-dimensional ones. Here we give out the mathematical definitions of 3-dimensional CG graph and Xingdu.

Definition of 3-dimensional CG graph

In a $M \times P \times Q$ 3-dimensional mesh grid, where M, P, and Q are integers, given a polyline $\mathbf{C}$, which is $C_0C_1\cdots C_i\cdots C_nC_{n+1}$ and composed of $n+2$ grid nodes C_i, where $i = 0,1,\cdots,n,n+1$, and n is a positive integer. In addition, except the start and the end point there should not be any points appearing repeatedly and any three sequential points cannot be located on a same straight line on $\mathbf{C}$. If there exists a polyline $\mathbf{C}'$, which is $C'_0C'_1\cdots C'_i\cdots C'_nC'_{n+1}$ and composed of $n+2$ grid nodes C'_i, similarly, except the start and the end point there are not any points appearing repeatedly and any three sequential points cannot be located on a same straight line on $\mathbf{C}'$, furthermore, the start and end points of $\mathbf{C}'$ are the same as those of $\mathbf{C}$, i.e. $C'_0 = C_0$, $C'_{n+1} = C_{n+1}$, and the corresponding line segments on $\mathbf{C}'$ and $\mathbf{C}$ are perpendicular each other accordingly, i.e. $C'_{j-1}C'_j \perp C_{j-1}C_j$, where $j = 1,2,\cdots,n,n+1$, then the closed graph composed by $\mathbf{C}$ and $\mathbf{C}'$ is a 3-dimensional CG graph noted as $\mathrm{CG}(\mathbf{C},\mathbf{C}')$. $\mathbf{C}$ and $\mathbf{C}'$ are the problem and solution of $\mathrm{CG}(\mathbf{C},\mathbf{C}')$, respectively.

Definition of 3-dimensional Xingdu

Let $\mathrm{CG}(\mathbf{C},\mathbf{C}')$ be a 3-dimensional CG graph in a given 3-dimensional mesh grid $M \times P \times Q$, where M, P, and Q are integers. If the solution of $\mathrm{CG}(\mathbf{C},\mathbf{C}')$, i.e. $\mathbf{C}'$, is unique, then the closed graph composed by $\mathbf{C}$ and $\mathbf{C}'$ is a 3-dimensional Xingdu denoted as Xingdu $(\mathbf{C},\mathbf{C}')$. $\mathbf{C}$ and $\mathbf{C}'$ are the problem and solution of $\mathrm{CG}(\mathbf{C},\mathbf{C}')$, respectively.

In comparison to the definitions of CG graph and Xingdu in literature [4], we do not constrain that $\mathbf{C}$ and $\mathbf{C}'$ cannot be on a same plane so that 2-dimensional CG graph or Xingdu can be considered as a special case of 3-dimensional ones. The purpose of doing this is that the method for 3-dimensional problems can also be applied to 2-dimensional problems. On the other hand, we include a new constraint that any three sequential points on $\mathbf{C}$ or $\mathbf{C}'$ cannot be located on a same line to lower the under-determinacy.

Based on the aforementioned definitions, we can construct a 3-dimensional Cartesian coordinate system in a given 3-dimensional mesh grid $M \times P \times Q$ to fulfill that $0 \le x \le M-1$, $0 \le y \le P-1$, $0 \le z \le Q-1$. Accordingly, the i^{th} grid node on $\mathbf{C}$ can be noted as $C_i\left(cx_i, cy_i, cz_i\right)$, where $i = 0,1,\cdots,n,n+1$, and n is a positive integer. cx_i, cy_i, cz_i are x, y, z coordinates of C_i, which are non-negative integers less than M, P, Q, respectively. Similarly, the i^{th} grid node on $\mathbf{C}'$ is noted as $C_i'\left(x_i, y_i, z_i\right)$, where $i = 0,1,\cdots,n,n+1$, and n is a positive integer. x_i, y_i, z_i are x, y, z coordinates of C_i', which are non-negative integers less than M, P, Q, respectively. Solving a CG$\left(\mathbf{C},\mathbf{C}'\right)$ or a Xingdu $\left(\mathbf{C},\mathbf{C}'\right)$ is equivalent to finding the corresponding polyline $\mathbf{C}'$ for the given problem polyline $\mathbf{C}$, which is fulfill with the definition of CG graph and Xingdu, respectively. Designing a CG$\left(\mathbf{C},\mathbf{C}'\right)$ or a Xingdu $\left(\mathbf{C},\mathbf{C}'\right)$ is equivalent to selecting appropriate polyline $\mathbf{C}$ as problem so that it has a corresponding polyline solution $\mathbf{C}'$ fulfilling with the definition of CG graph and Xingdu, respectively.

Next, we deduce the formulations related to the constraints about corresponding line segments perpendicularity, no repeated grid nodes appearing on $\mathbf{C}$ or $\mathbf{C}'$, and any three sequential grid nodes on $\mathbf{C}$ or $\mathbf{C}'$ being not on a same straight line.

Considering the line segment $C'_{j-1}C'_j$ and $C_{j-1}C_j$ as vectors $\overrightarrow{C'_{j-1}C'_j}$ and $\overrightarrow{C_{j-1}C_j}$, $C'_{j-1}C'_j \perp C_{j-1}C_j$ can be expressed using equation (1).

$$\left(x_j - x_{j-1}\right)\left(cx_j - cx_{j-1}\right) + \left(y_j - y_{j-1}\right)\left(cy_j - cy_{j-1}\right) + \left(z_j - z_{j-1}\right)\left(cz_j - cz_{j-1}\right) = 0 \qquad (1)$$

where, $j = 1,2,\cdots,n,n+1$. From the definition of Xingdu, we have $C'_0 = C_0$, $C'_{n+1} = C_{n+1}$,

Therefore, finding the solution $\mathbf{C}'$ is equivalent to determine the grid node $C'_k(x_k, y_k, z_k)$ between C'_0 and C'_{n+1}, where $k = 1,2,\cdots,n$.

Equation (1) can be re-written in matrix form as

$$\mathbf{Au} = \mathbf{b} \qquad (2).$$

where,

$$\mathbf{u} = \begin{bmatrix} x_1 & y_1 & z_1 & x_2 & y_2 & z_2 & \cdots & x_n & y_n & z_n \end{bmatrix}^{\mathbf{T}} \qquad (3).$$

$\mathbf{A} = \left[a_{j,p}\right]$ is a $(n+1)\times 3n$ matrix. $\mathbf{u}$ is a $3n\times 1$ column vector. $\mathbf{b} = \left[b_j\right]$ is a $(n+1)\times 1$ column vector. Matrix $\mathbf{A}$ has the form as follow

$$\mathbf{A} = \begin{bmatrix}
a_{1,1} & a_{1,2} & a_{1,3} & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
a_{2,1} & a_{2,2} & a_{2,3} & a_{2,4} & a_{2,5} & a_{2,6} & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & a_{3,4} & a_{3,5} & a_{3,6} & a_{3,7} & a_{3,8} & a_{3,9} & \cdots & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & a_{n-1,3n-8} & a_{n-1,3n-7} & a_{n-1,3n-6} & a_{n-1,3n-5} & a_{n-1,3n-4} & a_{n-1,3n-3} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & a_{n,3n-5} & a_{n,3n-4} & a_{n,3n-3} & a_{n,3n-2} & a_{n,3n-1} & a_{n,3n} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & 0 & 0 & a_{n+1,3n-2} & a_{n+1,3n-1} & a_{n+1,3n}
\end{bmatrix} \qquad (4).$$

The element of matrix $\mathbf{A}$, i.e. $a_{j,p}$, can be described as

$$a_{j,p} = \begin{cases}
0, & \text{if } p < 3j-5 \\
-(cx_j - cx_{j-1}), & \text{if } p = 3j-5 \\
-(cy_j - cy_{j-1}), & \text{if } p = 3j-4 \\
-(cz_j - cz_{j-1}), & \text{if } p = 3j-3 \\
cx_j - cx_{j-1}, & \text{if } p = 3j-2 \\
cy_j - cy_{j-1}, & \text{if } p = 3j-1 \\
cz_j - cz_{j-1}, & \text{if } p = 3j \\
= 0, & \text{if } p > 3j
\end{cases}, \quad \text{where, } j = 1,2,\cdots n+1; p = 1,2,\cdots 3n \qquad (5).$$

The vector $\mathbf{b} = \left[b_j\right]$ has the following form

$$\mathbf{b} = \begin{bmatrix}
cx_0(cx_1 - cx_0) + cy_0(cy_1 - cy_0) + cz_0(cz_1 - cz_0) \\
0 \\
\vdots \\
0 \\
-cx_{n+1}(cx_{n+1} - cx_n) - cy_{n+1}(cy_{n+1} - cy_n) - cz_{n+1}(cz_{n+1} - cz_n)
\end{bmatrix} \qquad (6).$$

As to the constraints that there are not any grid nodes appearing repeatedly except the start point and the end point on polyline $\mathbf{C}$, the corresponding condition can be expressed as

$$-\left|\left(cx_i-cx_j\right)\right|-\left|\left(cy_i-cy_j\right)\right|-\left|\left(cz_i-cz_j\right)\right|<0,\ \text{where } i,j=1,2,\cdots n, i\neq j \quad (7).$$

Similarly, the constraint that there are not any grid nodes appearing repeatedly except the start and the end point on polyline $\mathbf{C}'$ can be expressed as

$$-\left|\left(x_i-x_j\right)\right|-\left|\left(y_i-y_j\right)\right|-\left|\left(z_i-z_j\right)\right|<0,\ \text{where } i,j=1,2,\cdots n, i\neq j \quad (8).$$

Suppose there are any three sequential grid nodes on $\mathbf{C}$ located on a same straight line, then we have

$$\begin{cases}\left(cx_k-cx_{k-1}\right)\left(cy_{k+1}-cy_k\right)=\left(cx_{k+1}-cx_k\right)\left(cy_k-cy_{k-1}\right)\\ \left(cx_k-cx_{k-1}\right)\left(cz_{k+1}-cz_k\right)=\left(cx_{k+1}-cx_k\right)\left(cz_k-cz_{k-1}\right)\end{cases},\ \text{where},\ k=1,2,\cdots n \quad (9).$$

Accordingly, the constraint that any three sequential grid nodes on $\mathbf{C}$ cannot be located on a same straight line is equivalent to

$$\begin{aligned}&-\left|\left(cx_k-cx_{k-1}\right)\left(cy_{k+1}-cy_k\right)-\left(cx_{k+1}-cx_k\right)\left(cy_k-cy_{k-1}\right)\right|\\ &\quad-\left|\left(cx_k-cx_{k-1}\right)\left(cz_{k+1}-cz_k\right)-\left(cx_{k+1}-cx_k\right)\left(cz_k-cz_{k-1}\right)\right|<0,\ \text{where},\ k=1,2,\cdots n\end{aligned} \quad (10).$$

Similarly, the constraint that any three sequential grid nodes on $\mathbf{C}'$ cannot be located on a same straight line can be expressed as

$$\begin{aligned}&-\left|\left(x_k-x_{k-1}\right)\left(y_{k+1}-y_k\right)-\left(x_{k+1}-x_k\right)\left(y_k-y_{k-1}\right)\right|\\ &\quad-\left|\left(x_k-x_{k-1}\right)\left(z_{k+1}-z_k\right)-\left(x_{k+1}-x_k\right)\left(z_k-z_{k-1}\right)\right|<0,\ \text{where},\ k=1,2,\cdots n\end{aligned} \quad (11).$$

Thus far, we have obtained the mathematical formulations for $\mathrm{CG}\left(\mathbf{C},\mathbf{C}'\right)$ and Xingdu $\left(\mathbf{C},\mathbf{C}'\right)$ problem solving, which are described in (2) to (11).

3. Integer programming models for 3-dimensional Xingdu problem

Next, we discuss how to solve a Xingdu problem. Referring to the integer programming model based problem solving for Sudoku [10][12], we proposed the following integer programming model for Xingdu problem.

$$\min_{\mathbf{x}} \quad \mathbf{0}^T\cdot\mathbf{u}$$

$$\text{subject to:}\begin{cases}\mathbf{A}\cdot\mathbf{u}=\mathbf{b}\\ -\left|\left(x_i-x_j\right)\right|-\left|\left(y_i-y_j\right)\right|-\left|\left(z_i-z_j\right)\right|<0;\ \ i,j=1,2,\cdots n, i\neq j\\ -\left|\left(x_k-x_{k-1}\right)\left(y_{k+1}-y_k\right)-\left(x_{k+1}-x_k\right)\left(y_k-y_{k-1}\right)\right|\\ \quad -\left|\left(x_k-x_{k-1}\right)\left(z_{k+1}-z_k\right)-\left(x_{k+1}-x_k\right)\left(z_k-z_{k-1}\right)\right|<0; k=1,2,\cdots n\\ 0\le x_k\le M-1\\ 0\le y_k\le P-1\\ 0\le z_k\le Q-1\\ x_k,y_k,z_k\in\mathbf{Z};\forall k=1,2,\cdots,n\end{cases} \quad (12)$$

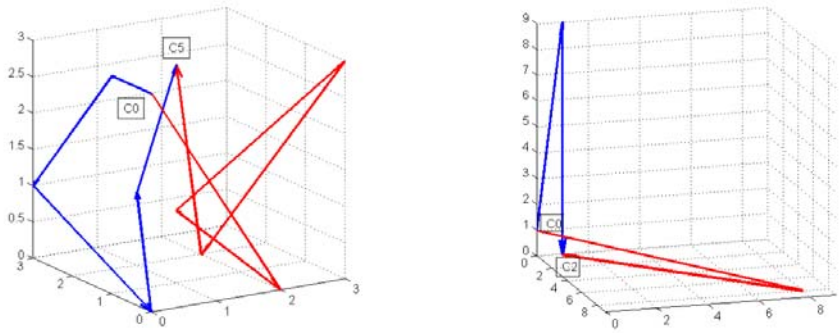
The basic ideal is introducing an objective function for the related Xingdu constraints described in section 2. In fact, our objective function is linear combination of decision variables, in which all coefficients are 0. This means that we actually do not optimize anything. The only purpose of objective function is that the developed integer programming methods can be used to a feasible solution fulfilling the Xingdu problem constraints. This trick is the same as that used in the reference [10], which using integer programming to solving Sudoku problem.

Looking into the model in (12) finds that we have nonlinear constraints related to any three sequential points cannot be located on a same straight line on the solution $\mathbf{C}'$. This means that the integer programming related to the model (12) usually can be solved only by enumerating. For this reason, we neglect the those constraints with absolute value function and nonlinear function first and obtain a simplified model only with linear equation constraints, which gives the model as

$$\min_{\mathbf{u}} \quad \mathbf{0}^T \cdot \mathbf{u}$$

$$\text{subject to: } \begin{cases} \mathbf{A} \cdot \mathbf{u} = \mathbf{b} \\ 0 \le x_k \le M - 1 \\ 0 \le y_k \le P - 1 \\ 0 \le z_k \le Q - 1 \\ x_k, y_k, z_k \in \mathbf{Z}; \forall k = 1, 2, \cdots, n \end{cases} \qquad (13).$$

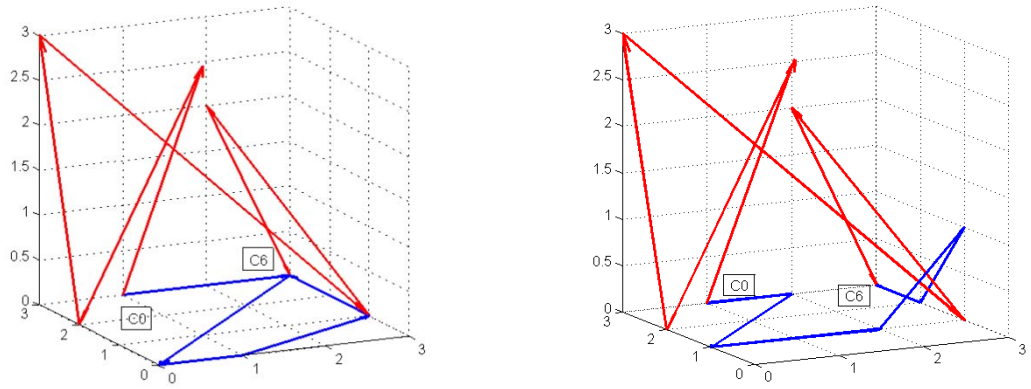
Integer programming related to the model in (13) has only linear constraints so that the well-developed methods for integer programming can be applied. If there exist a feasible solution related to (13), we can verify further whether the solution fulfill the constraints described in (8) and (11) to check whether the feasible solution is the true solution for our Xingdu problem.



(a) (b)

Figure 2. Examples for Xingdu problem solving by using the integer programming model described in (13), which true solutions are correctly found.

Figure 2 demonstrates Examples for Xingdu problem solving by using the integer programming model described in (13), in which red and blue polylines are problems and solutions, respectively.



(a) (b)

Figure 3. Example for Xingdu problem solving by using the integer programming model described in (13), which true solution is not correctly found.

Figure 3 is an example for Xingdu problem solving by using the integer programming model in (13), which true solution is not correctly found. The problem is the Xingdu problem 5-3-9 from the reference [4]. The problem polyline related to this problem is C=[1 3 0;1 1 3;0 2 0;0 3 3;3 1 0;2 3 2;3 3 0]. The blue polyline in Figure 2(a) is the solution obtained by using the integer programming model described in (13), which is corresponding the polyline C'=[1 3 0;3 3 0;0 0 0;1 0 0;1 0 0;3 1 0;3 3 0]. It is found there are two groups of points are the same. One is composed of the 2nd and the 7th point. Another is composed of the 4th and the 5th point. The true solution of this problem is C'=[1 3 0;2 3 0;0 1 0;2 1 0;3 1 1;3 2 0;3 3 0]，which is shown in Figure 2(b).

According to the above analysis, it is found that the performances of the integer programming model described in (13) for Xingdu problem solving are not satisfied enough. It is necessary to investigate other possibilities with good performance for Xingdu problem solving. Therefore, we need to re-investigate the Xingdu problem.

We reconsider the Xingdu problem solving as selecting and permuting appropriate mesh grid nodes to construct the solution polyline, meanwhile all constraints about Xingdu definition should be fulfilled. We use '1' to specify those mesh grid nodes that are selected and '0' to stand for those mesh grid nodes that are not selected. This is very analogy to the traditional 'Knapsack problem' and may solvable by using binary integer programming.

To implement this, we need to number all nodes in the mesh grid. Together with the

coordinate system established in previous context, we can derive the relationship between the index number of a mesh grid and its coordinate, which are expressed in equation (14) and (15). This is based on the numbering mesh grid nodes in the direction of x, y, z axis sequentially. Equation (14) is the mapping from coordinates to index number. Equation (15) is the mapping from index number to mesh grid node coordinates.

$$index = x + M \times y + M \times P \times z + 1 \tag{14}$$

$$\begin{cases} z = fix\left(\left(index - 1\right)/\left(M \times P\right)\right) \\ y = fix\left(\left(index - z \times M \times P - 1\right)/M\right) \\ x = index - z \times M \times P - y \times M - 1 \end{cases} \tag{15}$$

Where, $fix(\ )$ stands from round down a number as an integer. Here is an example in the mesh grid $4 \times 4 \times 4$. The coordinates of the node No.55 can be obtained according to equation (15)， which are $(2,1,3)$. Next, we introduce an index matrix $\mathbf{W}$ as follow

$$\mathbf{W} = \begin{array}{c} \begin{matrix} C'_0 & C'_1 & C'_2 & \cdots & C'_{n-1} & C'_n & C'_{n+1} \end{matrix} \\ \begin{bmatrix} 0 & 0 & 1 & \cdots & 0 & 0 & 0 \\ 1 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 1 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & 1 & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 1 \end{bmatrix} \end{array} \tag{16}.$$

Where the component of matrix $\mathbf{W}$ in the r^{th} row and the l^{th} column is $w_{r,l}$. And $r = 1,2,\cdots,R$， $R = M \times P \times Q, l = 1,2,\cdots,n,n+1,n+2$. The row number of matrix $\mathbf{W}$ equals the number of nodes in the mesh grid. The column number of matrix $\mathbf{W}$ equals the number of mesh grid nodes on the polyline $\mathbf{C}'$ or $\mathbf{C}$. $w_{r,l}$ can only be 0 or 1. Furthermore, there is one and only one ’1’ in each column of $\mathbf{W}$. And there is at most one ’1’ in each row of $\mathbf{W}$. $w_{r,l} = 1$ stands for that the point C'_{l-1} on $\mathbf{C}'$ is the r^{th} mesh grid node.， $w_{r,l} = 0$ stands for that the point C'_{l-1} on $\mathbf{C}'$ cannot be the r^{th} mesh grid node. Here we still take an example in a $4 \times 4 \times 4$ mesh grid. $w_{3,1} = 1$ means C'_0 on $\mathbf{C}'$ is the mesh grid node No.3. $w_{16,2} = 1$ means C'_0 on $\mathbf{C}'$ is the mesh grid node No.16. Re-arranging the component of

$\mathbf{W}$ in column component-wise gives

$$\mathbf{v}=\begin{bmatrix} w_{1,1} & w_{2,1} & \cdots & w_{R,1} & w_{1,2} & w_{2,2} & \cdots & w_{R,2} & \cdots & w_{1,n+2} & w_{2,n+2} & \cdots & w_{R,n+2} \end{bmatrix}^{\mathrm{T}} \quad (17).$$

v_t is the t^{th} component of $\mathbf{v}$, where $t=1,2,\cdots,(n+2)\times M\times P\times Q$. Taking $\mathbf{v}$ as decision variables, we can rewrite the constraints related to the Xingdu problem as follows.

$$\sum_{t=i\times M\times P\times Q+1}^{(i+1)\times M\times P\times Q} v_t=1,\forall i=0,1,\cdots,n+1 \quad (18)$$

$$\sum_{i=0}^{n+1}\sum_{t=i\times M\times P\times Q+r} v_t\le 1,\forall r=1,\cdots,M\times P\times Q \quad (19)$$

$$\begin{cases} index\_C_0=cx_0+M\times cy_0+M\times P\times cz_0+1 \\ t=index\_C_0 \\ v_t=1 \end{cases} \quad (20)$$

$$\begin{cases} index\_C_{n+1}=cx_{n+1}+M\times cy_{n+1}+M\times P\times cz_{n+1}+1 \\ t=(n+1)\times M\times P\times Q+index\_C_{n+1} \\ v_t=1 \end{cases} \quad (21)$$

$$\begin{cases} \left(\sum_{t=j\times M\times P\times Q+1}^{(j+1)\times M\times P\times Q} v_t\cdot\begin{bmatrix} v_{t\_x} \\ v_{t\_y} \\ v_{t\_z} \end{bmatrix}-\sum_{t=(j-1)\times M\times P\times Q+1}^{j\times M\times P\times Q} v_t\cdot\begin{bmatrix} v_{t\_x} \\ v_{t\_y} \\ v_{t\_z} \end{bmatrix}\right)^{\mathrm{T}}\begin{bmatrix} cx_j-cx_{j-1} \\ cy_j-cy_{j-1} \\ cz_j-cz_{j-1} \end{bmatrix}=0 \\ v_{t\_z}=fix\left(\mathrm{mod}\left(t-1,M\times P\times Q\right)/\left(M\times P\right)\right) \\ v_{t\_y}=fix\left(\left(\mathrm{mod}\left(t-1,M\times P\times Q\right)-z\times M\times P\right)/M\right) \\ v_{t\_x}=\mathrm{mod}\left(t-1,M\times P\times Q\right)-z\times M\times P-y\times M \\ \forall j=1,2,\cdots,n+1;\forall t=1,2,\cdots,(n+2)\times M\times P\times Q \end{cases} \quad (22)$$

Where, $fix(\ )$ stands from round down a number as an integer. $\mathrm{mod}(a,b)$ is the integer remainder of integer a divided by integer b.

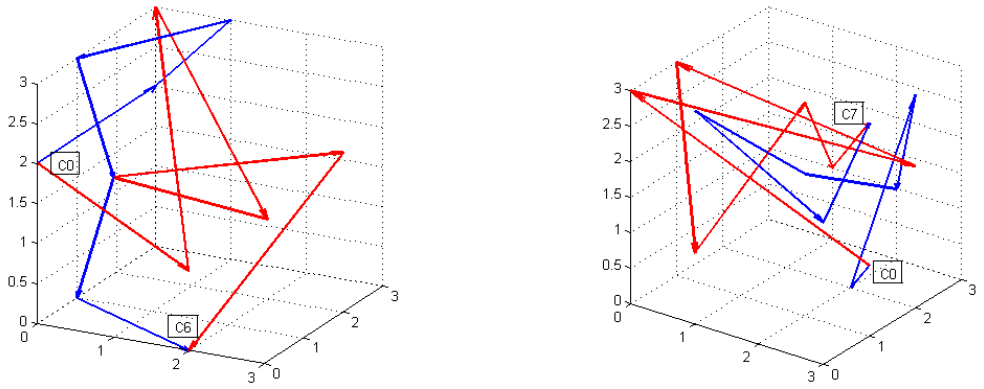
Equation (18) describes that there is just only one '1' in each column of $\mathbf{W}$, i.e. the point C'_{l-1} on the polyline $\mathbf{C}'$ can only certain mesh grid node. Equation (19) describes that there is at most one '1' in each row of $\mathbf{W}$, i.e. certain mesh grid node can only appear at most once time on the polyline $\mathbf{C}'$, which is equivalent to that no repeated mesh grid nodes on the polyline $\mathbf{C}'$. Equation (20) and (21) describe how to obtain the index numbers of start and end point on

the solution polyline $\mathbf{C}'$, which are already known and the same as those on the problem polyline $\mathbf{C}$. Equation (22) express that the corresponding segmented lines on the problem polyline $\mathbf{C}$ are perpendicular to those the solution polyline $\mathbf{C}'$.

Re-organizing（16）~（22）and using the similar strategy of zero coefficient objective function, we obtained the following binary integer programming model for Xingdu problem solving.

$$\begin{aligned} &\min_{\mathbf{v}} && \mathbf{0}^T \cdot \mathbf{v} \\ &\text{subject to:} && \begin{cases} \mathbf{A}_1 \cdot \mathbf{v} = \mathbf{b}_1 \\ \mathbf{A}_2 \cdot \mathbf{v} \le \mathbf{b}_2 \\ 0 \le v_t \le 1 \\ v_t \in \mathbf{Z}; \forall t = 1, 2, \cdots, (n+2) \times M \times P \times Q \end{cases} \end{aligned} \qquad (23)$$

Where，$\mathbf{A}_1$，$\mathbf{b}_1$ are coefficient matrix and right hand side related to all equalities in (18), (20), (21), and (22), respectively. $\mathbf{A}_2$， $\mathbf{b}_2$ are coefficient matrix and right hand side related to all inequalities in (19). It is worth noted that our proposed binary integer programming model for Xingdu problem solving in (23) does not include the constraint that any three continuously sequential points on the solution polyline $\mathbf{C}'$ cannot locate on a same straight line. Therefore, we still need to check whether it is fulfill this constraint if a feasible solution is obtained.



(a) (b)

Figure 4. Examples for Xingdu problem solving by using the binary integer programming model described in (23). (a) mesh grid scale: 4×4×4, number of line segments of problem:6; (b) mesh grid scale: 4×4×4, number of line segments of problem:7

Figure 3. demonstrates two examples for Xingdu problem solving by using our proposed binary integer programming model described in (23), which the problems are the problem 5-3-7 and the problem 5-3-10 from the reference [4]. The true solutions of these two problems cannot be obtained by using the aforementioned simplified integer programming model in (13). But using the binary integer programming model in (13), we obtained feasible solutions fulfilling all constraints about Xingdu, i.e we obtained the true solutions of the problem 5-3-7 and the problem 5-3-10. As to the problem 5-3-9 mentioned in previous context. We also obtained its true solution. According to these comparable analyses, it is found that our proposed binary integer programming model for Xingdu problem solving in (23) is of improving the performances of finding the true solution.

Even though, we still need to point out certain key features about our proposed binary integer programming model in (23). If we know a problem is a Xingdu problem and find by using the model in (23) a solution that fulfills all constraints about Xingdu, we can conclude that the solution is the true solution. This is because we already know that the problem is a Xingdu problem and Xingdu only has only a unique solution. On the other hand, if we have to solve a problem but do not know whether it is a Xingdu. Even if we find a solution by using the model in (23) a solution that fulfills all constraints about Xingd, we can only conclude that the problem is a CG graph and the solution found is only one solution of this CG graph.

4. CG graph and Xingdu problem designing

In aforementioned section, we discussed the integer programming models for CG graph and Xingdu problem solving and demonstrated a number of examples solved by using our proposed models. In practice, a question may be raised is where are from these CG graph or Xingdu problems and how they are designed. Next, we will discuss this topic in detail.

Our first intuition is to implement the Xingdu problem designing by enumerating. Give a mesh grid scale and the number of segmented lines of the problem, we can enumerate all possible polylines in the mesh grid and discriminate whether any polyline has only one corresponding polyline fulfilling the Xingdu constraints. Unfortunately, we will face the problem of combinatorial explosion. Here we provide an example in Table 1.

Table 1. Combinatorial explosion of Xingdu problem designing by enumerating

Mesh grid scale	Number of segmented lines of the problem	Possible polylines in the mesh grid	Number of Xingdu found	Running time
3×3×3	4	$\approx 9.7\times10^{6}$	$\approx 2.8\times10^{5}$	≈122 seconds
5×5×5	5	$\approx 3.4\times10^{12}$	N/A	≈492 days

Table 1 shows that, due to the combinatorial explosion, designing a Xingdu problem by enumerating is trivial from point view of practical applications. Therefore, we proposed a

strategy that may be used in practice for CG graph or Xingdu problem designing, which is depicted in Figure 5.

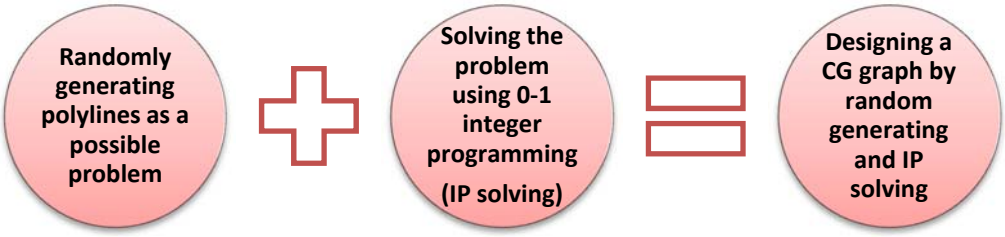
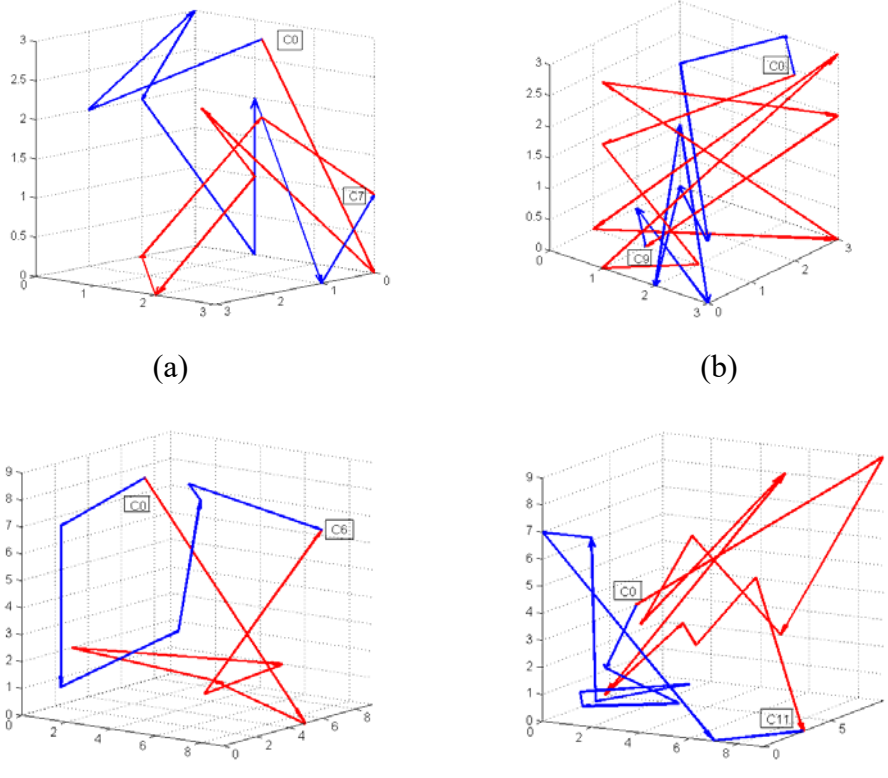


Figure 5. The strategy for CG graph problem designing by random generating and IP solving

Given a mesh grid scale and the number of segmented lines of the problem, we randomly generate a polyline as a problem accordingly and then use our proposed binary integer programming model to solve this problem. If a feasible solution fulfilling the Xingdu constraints is found, we say that the problem is a CG graph. However, we cannot judge whether this problem is a Xingdu because the uniqueness of the solution cannot be guaranteed by integer programming. In Figure 6, we demonstrated certain CG graph examples designed by the strategy in Figure 5.



(a) (b)

(c) (d)

Figure 6. CG graph examples designed by using the strategy in Figure 4. (a) mesh grid scale: 4×4×4, number of line segments of problem:7; (b) mesh grid scale: 4×4×4, number of line segments of problem:9; (c) mesh grid scale: 10×10×10, number of line segments of problem:6; (d) mesh grid scale: 10×10×10, number of line segments of problem:10.

The mesh grid nodes construct the problem polyline and the solution polyline for the CG graph examples in Figure 5 are provided as follows.

Figure 6(a)： C=[1 2 3; 0 3 0; 1 1 2; 0 1 1; 3 2 0; 1 0 0; 1 2 2; 0 3 1]；
C'=[1 2 3; 2 0 2; 0 0 3; 1 0 2; 0 1 0; 0 1 2; 1 3 0; 0 3 1];

Figure 6(b)： C=[3 2 3; 1 0 2; 2 1 0; 1 0 0; 3 3 3; 0 1 0; 3 3 0; 1 0 3; 3 3 2; 1 1 0]；
C'=[3 2 3; 2 3 3; 0 3 2; 3 0 1; 0 3 0; 2 0 0; 0 3 1; 3 0 0; 0 2 0; 1 1 0];

Figure 6(c)： C=[4 2 9; 9 5 0; 2 9 0; 0 3 2; 5 9 1; 3 7 0; 9 6 7]；
C'=[4 2 9; 1 1 7; 1 1 1; 7 0 4; 8 0 9; 6 2 9; 9 6 7];

Figure 6(d)： C=[0 7 3; 9 9 9; 7 5 3; 5 2 7; 4 0 4; 5 9 8; 2 1 1; 3 5 3; 3 6 2; 6 5 5; 9 3 0]；
C'=[0 7 3; 2 1 2; 5 1 1; 0 3 0; 1 1 1; 6 0 2; 0 4 0; 2 0 7; 0 0 7; 7 0 0; 9 3 0];

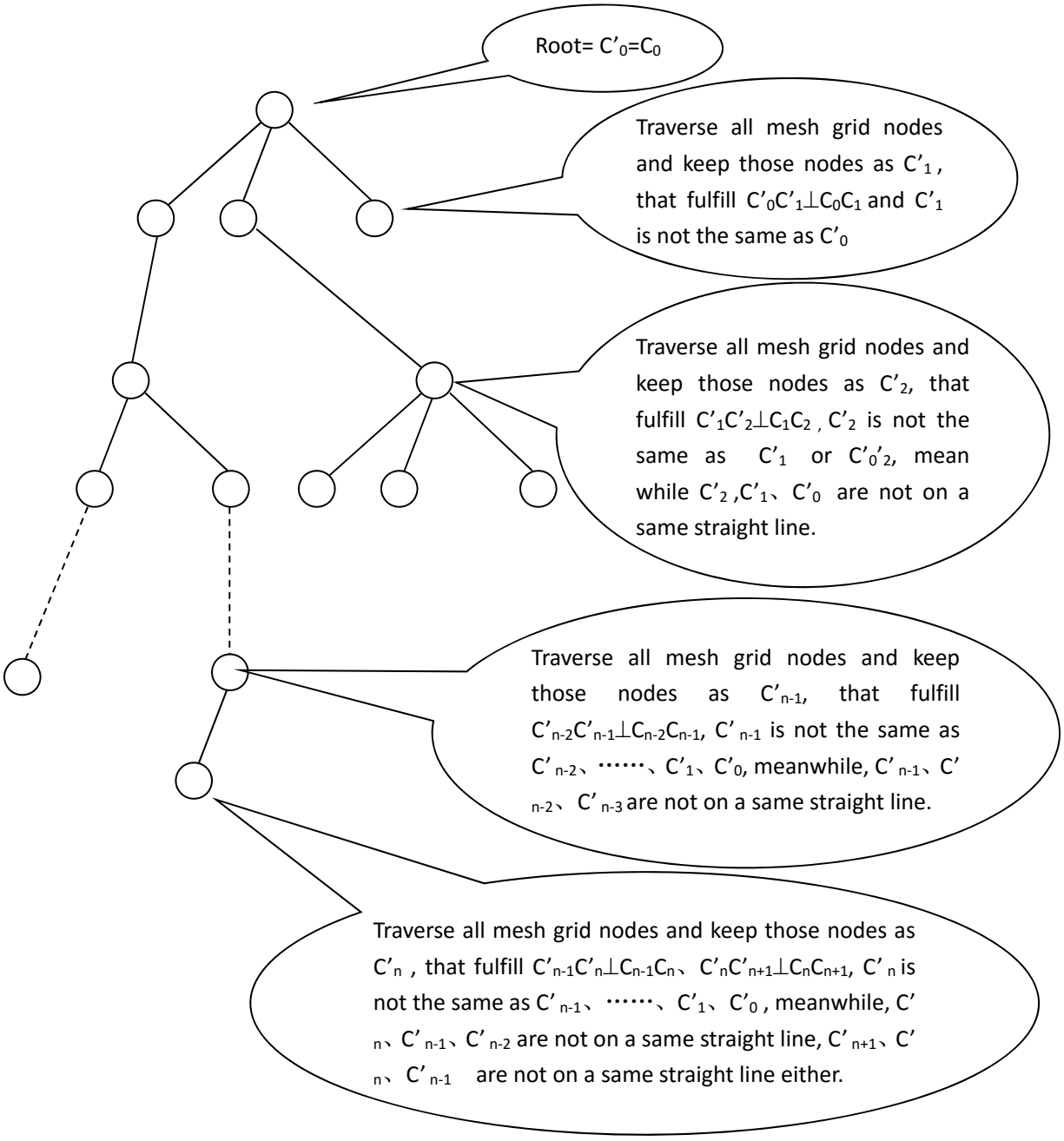


Figure 7. Search tree for CG graph and Xingdu problem solving and designing

From above analyses and examples, we still have to consider how to design a Xingdu problem practically. The original integer programming model described in (12) for Xingdu problem solving is an integer programming with nonlinear constraints, which usually can be solved by enumerating. The well-developed methods for integer programming, such as the branch and bound method, and the Gomory cut method, can be considered as improving the efficiency of enumerating by using different strategies [13][14]. Inspired from the branch and bound method for integer programming, we finally proposed an approach to solve and design CG graph and Xingdu problems by constructing and pruning a search tree. It is shown in Figure 7.

After the search tree is constructed, we can determine whether the problem related to the constructed search tree is a CG graph or a Xingdu. If the tree has at least one branch which depth is the same as the line segments, the problem is a CG graph. Furthermore, if the number is one, it is a Xingdu. Otherwise, the problem is neither a CG graph nor a Xingdu when the number is zero. Checked by this algorithm, none of the examples shown in Figure 6 is Xingdu. Using the similar strategy as that in Figure 5, we presented in Fig.8 the strategy of designing CG graph and Xingdu problems by random simulation together with search tree.

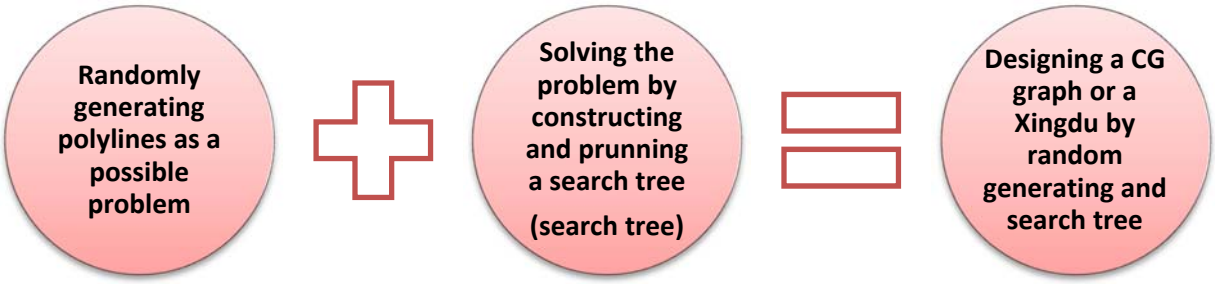


Figure 8. The strategy for CG graph and Xingdu problem designing by random generating and search tree

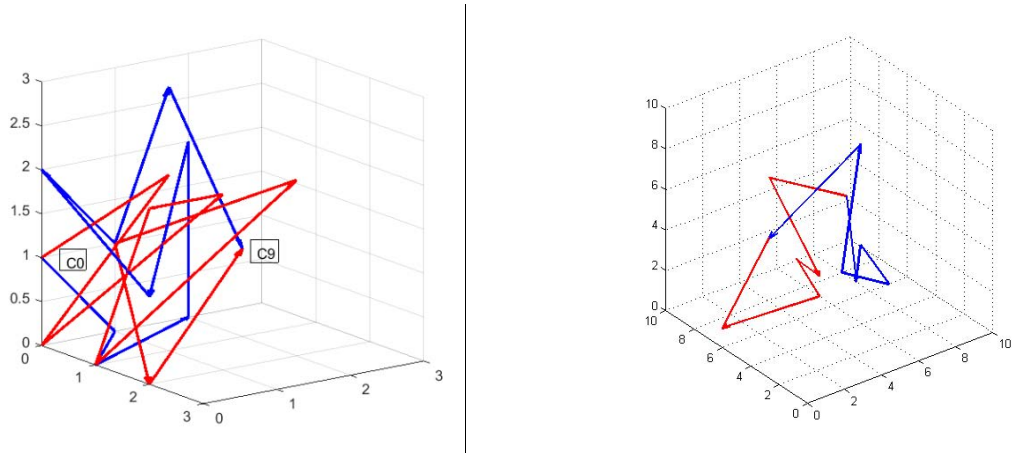
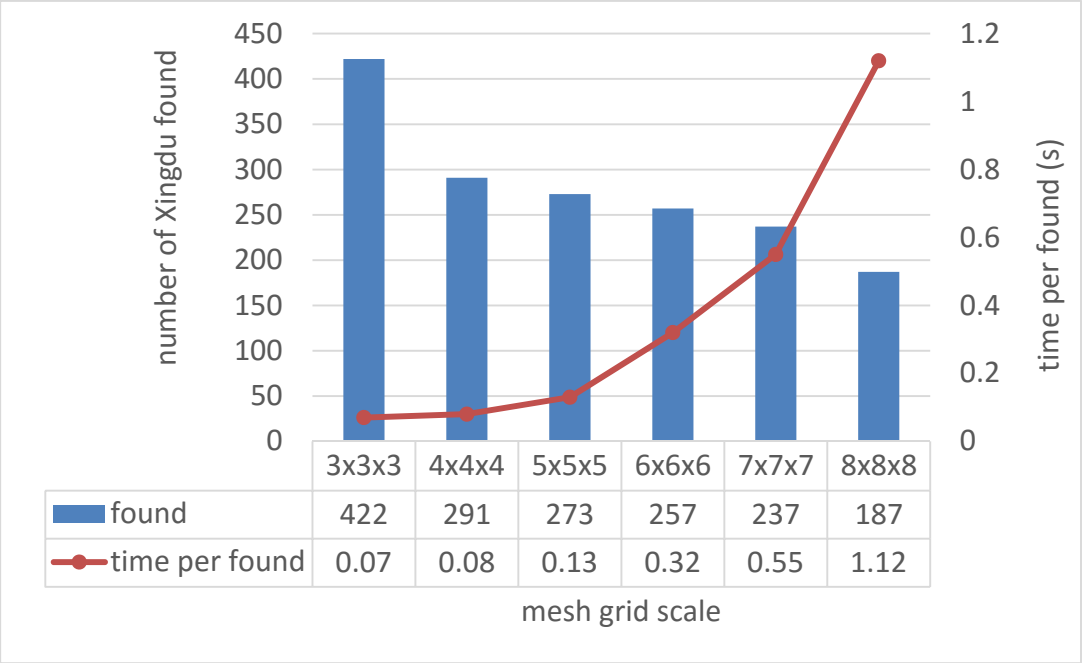


Figure 9. Two Xingdu examples designed by using random generating together with search tree. (a) mesh grid scale: 4×4×4, number of line segments of problem:9; (b) mesh grid scale: 10×10×10, number of line segments of problem:6.

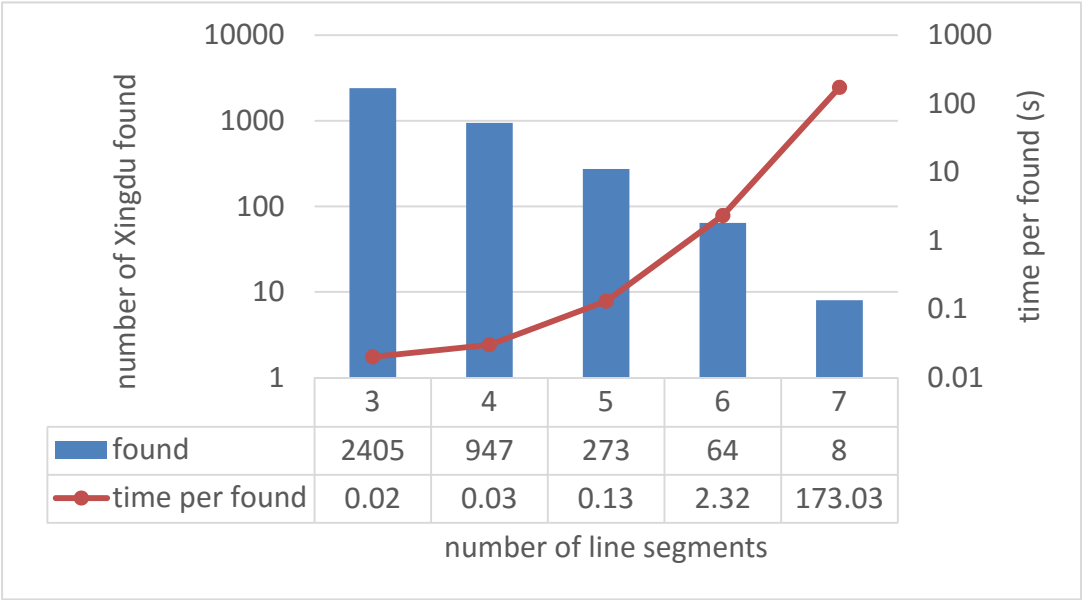
Figure 9 demonstrates two Xingdu examples to designed by using above strategy. The problem and solution corresponding to the examples in Figure 9 are given below.

Figure 9(a): C=[0 0 1; 1 1 2; 0 0 0; 2 1 2; 2 0 2; 1 0 0; 2 2 2; 0 1 1; 2 0 0; 1 2 1];
C'=[0 0 1; 0 1 0; 1 0 0; 0 2 0; 0 2 2; 2 0 1; 0 0 2; 0 1 1; 1 1 3; 1 2 1];

Figure 9(b): C=[9 9 3; 1 4 9; 6 7 1; 4 6 3; 6 7 0; 0 6 1; 1 4 6];
C'=[9 9 3; 8 7 0; 9 8 1; 9 6 0; 8 8 0; 9 8 6; 1 4 6];



(a)



(b)

Figure 10. Statistics of 10000 experiments related to Xingdu designing. (a) comparison of effect of different mesh grid scales while the number of line segments is 5, (b) comparison of effect of different line segments while the mesh grid scale is 5x5x5.

For comparison purpose, the statistics of 10000 experiments related to Xingdu problem designing with the same number of line segments in different mesh grid scales and the statistics of 10000 experiments related to Xingdu problem designing with different number of line segments in the same mesh grid scale are demonstrated in Figure 10(a) and 10(b), respectively. It is found that with the increase of problem scale, particularly the number of line segments, it becomes more and more difficult, and takes more and more time to find a Xingdu problem.

In practice, we also can generate new Xingdu problems from known Xingdu problems by using geometry transform. For example, suppose that we already have a Xingdu problem $\mathbf{C}$ in a mesh grid of $M \times P \times Q$, we can generate a new Xingdu problem $\mathbf{C}_{transformed}$ by easily finding the mirror image of $\mathbf{C}$ with respect to the plane $x = \frac{M}{2}$. Furthermore, the solution of $\mathbf{C}_{transformed}$ is also the mirror image of the solution of $\mathbf{C}$ with respect to the plane $x = \frac{M}{2}$.

5. Summary

Referring to the achievements in mathematical methods for Sudoku, we studied preliminarily the possibility of using optimization theory to solve the problem of Xingdu. We proposed two integer programming models for Xingdu problem, which are the model using only line segments perpendicular property as constraints and the binary integer programming model using line segments perpendicular property together with no repeated points as constraints. The test examples show that the proposed binary integer programming model has a very good performance in finding the solution of Xingdu problem. We also discussed the possibility of using the proposed binary integer programming model together with random simulation to design a CG graph or a Xingdu problem. Due to the challenge of discriminating the uniqueness of Xingdu solution, we finally presented an approach to solve and design the Xingdu problem by constructing and pruning a search tree.

References

[1] Q. M. Yang and R. Xu: “点可点非常点——格点”, Peking University Press, 2011.7.

[2] Students from the first middle school in Xi’an high technology district: “On existence of Xingdu and CG graph problems” in Chinese, 2011.8.

[3] Y. W., J. W. Zheng, Y. F. Gong, F. H. Liang, X. Y. Xu, K. F. Gu：“Extension of CG graph in 3-Dimensional space” in Chinese，2011.8.

[4] Q. M. Yang，“Xingdu” in Chinese，Tsinghua University Press, 2012.9.

[5] W. H. Li, Y. X Liao, W. W Liu, J. Q. Xia, W. R. Yang, Z. Q. Zhou: “Preliminary discussion on Xingdu problem and its properties ” in Chinese，2011.8.

[6] T. Yato, “Complexity and completeness of finding another solution and its application to puzzles,” Master’s thesis, University of Tokyo, January 2003.
[7] F. Simons, “Solving a sudoku puzzle with Mathematica”, Mathematica in Education and Research, Vol. 10, No 4, 2005, 1 - 24.
[8] M. Chlond, “Classroom exercises in IP modelling: sudoku and the log pile”, INFORMS Transactions on Education, Vol. 5, No 2, January 2005.
[9] A. Herzberg and M. Murty, “Sudoku squares and chromatic polynomials,” Notices of the AMS, vol. 54, no. 6, pp. 708–716, June/July 2007.
[10] A. Bartlett, T. Chartier, A. Langville, and T. Rankin, “An integer programming model for the sudoku problem,” The Journal of Online Mathematics and Its Applications, 2008.
[11] Sudoku as a constraint program. 4th International Workshop on Modelling and Reformulating Constraint Satisfaction Problems, 2005.
[12] M. Thein, “An in depth analysis of Sudoku with focus on integer and constraint programming”, Master’s thesis, University of the Pennsylvania State University, January 2014.
[13] T. C. Hu and A. B. Kahng, “Linear and Integer Programming Made Easy”, Springer, 2016.
[14] M. Conforti, G. Cornuejols, G. Zambelli, “Integer Programming”, Springer, 2014.