Solving Maximum Clique Problems by Differential Evolution Algorithm

การแก้ปัญหากราฟบริบูรณ์ย่อยใหญ่สุดโดยขั้นตอนวิธีวิวัฒนาการเชิงผลต่าง

Jutamard Wongpen (จุฑามาศ วงษ์เพ็ญ)* Dr.Pikul Puphasuk (ดร.พิกุล ภูผาสุข)**

ABSTRACT

The maximum clique problem (MCP) is finding the maximum complete subgraphs of graph *G*. In this research, we solve the MCP which is transformed into constrained continuous optimization problem by using a modified differential evolution algorithm. We test its performance with 15 constructed graphs and 12 benchmark graphs. The experimental results show that the proposed algorithm can find the maximum cliques for all tested graphs, which indicates that it is effective for MCP.

บทคัดย่อ

ปัญหากราฟบริบูรณ์ย่อยใหญ่สุดคือการหากราฟบริบูรณ์ที่ใหญ่ที่สุดของกราฟ *G* งานวิจัยนี้จะแก้ปัญหากราฟบริบูรณ์ย่อยใหญ่สุดที่ถูกเปลี่ยนเป็นปัญหาการหาค่าเหมาะสมที่สุดแบบต่อเนื่องที่มีเงื่อนไขบังคับโดยใช้ขั้นตอนวิธีวิวัฒนาการเชิงผลต่างแบบปรับปรุง และทดสอบประสิทธิภาพของขั้นตอนวิธีด้วยกราฟที่สร้างขึ้นจำนวน 15 กราฟและกราฟมาตรฐานจำนวน 12 กราฟ ผลการทดลองแสดงว่าขั้นตอนวิธีที่นำเสนอสามารถหากราฟบริบูรณ์ย่อยใหญ่สุดได้สำหรับทุกกราฟทดสอบและเป็นวิธีที่มีประสิทธิภาพสำหรับปัญหากราฟบริบูรณ์ย่อยใหญ่สุด

Keywords: Maximum clique problems, Differential evolution algorithm, Optimization

คำสำคัญ: ปัญหากราฟบริบูรณ์ย่อยใหญ่สุด ขั้นตอนวิธีวิวัฒนาการเชิงผลต่าง การหาค่าเหมาะสมที่สุด

**Student, Master of Science Program in Applied Mathematics, Department of Mathematics, Faculty of Science, Khon Kaen University*

***Assistant Professor, Department of Mathematics, Faculty of Science, Khon Kaen University*

Introduction

Let $G = (V, E)$ be an undirected graph where $V = \{1, 2, \dots, n\}$ is the vertex set and $E \subseteq V \times V$ is the edge set. A graph is complete if all its vertices are pairwise adjacent. The complete graph with m vertices is denoted by K_m and a complete subgraph of G is called a clique. A maximum clique is a clique with the largest number of vertices. This number is called the clique number denoted by $\omega(G)$. The maximum clique problem (MCP) is to find a maximum clique of graph G (Harary, Ross, 1957). The MCP often occurs in many applications such as financial networks (Boginski et al., 2006), scheduling (Dorndorf et al., 2008; Weide et al., 2010), social network analysis (Balasundaram et al., 2011; Pattillo et al., 2012), telecommunication (Balasundaram, Butenko, 2006) and wireless networks (Butenko, 2003). Finding the maximum clique in an arbitrary graph is difficult because it is the NP-hard problem (Karp, 1972). The MCP can be solved by some exact algorithms (Carraghan, Pardalos, 1990) and heuristic methods (Bomze et al., 1999) but the best results can not be reached for all cases. The discrete MCP is transformed into the constrained continuous optimization problem using the following formulation (Motzkin, Straus, 1964):

$$\begin{aligned} &Maximize\ f(x) = \sum_{\{i,j\}\epsilon E, i<j} x_i x_j \\ &subject\ to \sum x_i = 1 \\ &0 \le x_i \le 1, i = 1, 2, \dots, n. \end{aligned} \tag{1}$$

For solving a constrained continuous optimization problem, we need an effective global optimization algorithm. The differential evolution algorithm (DE) is one of well-known optimization methods (Storn, Price, 1997). The DE algorithm is a simple and efficient algorithm which consists of three operators: mutation, crossover and selection. In this study, we propose a modified DE for solving the constrained continuous MCP.

Objectives of the study

The purpose of the study is to solve the MCP which is transformed into constrained continuous optimization problem by using modified DE algorithm and test the performance of algorithm with 15 constructed graphs and 12 benchmark graphs.

Methodology

To solve the MCP which is transformed into constrained continuous optimization,

$$\begin{aligned} &Maximize\ f(x) = \sum_{\{i,j\}\epsilon E, i<j} x_i x_j \\ &subject\ to \sum x_i = 1 \\ &0 \le x_i \le 1, i = 1, 2, \dots, n. \end{aligned}$$

The basic differential evolution algorithm, the proposed algorithm and the tested graphs are described as follows.

1. Differential evolution algorithm

The basic DE algorithm is described as follows:

Step 1. Initialization

The initial population of real vectors $X = [x_i]$ where $x_i = [x_{i1}, x_{i2}, \dots, x_{iN}], L \leq x_{ij} \leq U$ is randomly generated, where $i = 1, 2, \dots, NP$ and $j = 1, 2, \dots, N$; N is the dimension or number of parameters and NP is the number of population vectors.

Step 2. Mutation

For each target vector x_i^G of the generation G, the mutant vector v_i^G is computed by

$$v_i^G = x_{r1} + F(x_{r2} - x_{r3}) \quad (2)$$

where r_1, r_2 and r_3 are randomly chosen in the set $\{1, 2, \dots, NP\}$ such that $r_1 \neq r_2 \neq r_3 \neq i$ and $F \in [0,1]$ is a scaling factor.

Step 3. Crossover

The trial vector u_i^G is then formed to increase the diversity of the mutant vectors. It is computed by

$$u_{ij}^G = \begin{cases} v_{ij}^G & ;\ rand \leq CR \text{ or } j = I_{rand} \\ x_{ij}^G & ;\ \text{otherwise} \end{cases} \quad (3)$$

where $CR \in [0,1]$ is a crossover rate, I_{rand} is a random integer in the set $\{1, 2, \dots, N\}$, $j = 1, 2, \dots, N$ and $rand$ is a random number in range of $(0,1)$.

Step 4. Selection

The target vector x_i^{G+1} of next generation is selected by comparing fitness values of the trial vector u_i^G and the target vector x_i^G as follows

$$x_i^{G+1} = \begin{cases} u_i^G & ;\ f(u_i^G) > f(x_i^G) \\ x_i^G & ;\ f(u_i^G) \leq f(x_i^G). \end{cases} \quad (4)$$

Step 5. Repeat step 2-4 until the maximum number of generations (NG) is reached.

2. Modified DE algorithm for solving maximum clique problem

Our modified DE algorithm for solving the MCP is described as follows:

Step 1. Initialization

Initial population of real vectors $X = [x_i]$, where $x_i = [x_{i1}, x_{i2}, \dots, x_{iN}], 0 \leq x_{ij} \leq 1$ is randomly generated and is normalized to satisfy $\sum_{j=1}^{n} x_{ij} = 1$.

Step 2. Mutation

Use the same mutation operator of the basic DE with the random F values in the range of [0.5,0.7]. We need to adjust the negative components of v_i^G by

$$v_{ij}^G = \begin{cases} 0 & ;\ v_{ij}^G < 0 \\ v_{ij}^G & ;\ \text{otherwise.} \end{cases} \quad (5)$$

Then normalize v_i^G by $v_{ij}^G = \frac{v_{ij}^G}{\sum v_{ij}^G}$.

Step 3. Crossover

Use the same crossover operator of the basic DE with the fixed value $CR = 0.9$. Normalize u_i^G by $u_{ij}^G = \frac{u_{ij}^G}{\sum u_{ij}^G}$.

Step 4. Selection

Use the same selection operator of the basic DE. If $f(x_i^{G+1})$ cannot be improved until the limitation number (*limit*), then generate a new vector x_i^{G+1} as in the initialization.

Step 5. Repeat step 2-4 until the maximum number of generations is reached.

In this study, we set the parameters for constructed graph as $NP = 100, NG = 1000, limit = 100$. For benchmark graphs, if $|V| \leq 30$ we set $NP = 100, NG = 1000, limit = 100$; otherwise $NP = 200, NG = 2000, limit = 100$. Each tested graph is performed 20 runs.

3. Tested graphs

We test the performance of the proposed algorithm with the tested graphs consisting of two types of graphs as follows.

3.1 Constructed graphs

For the first group of tested graphs, we divide the vertex set $\{1, 2, \ldots, N\}$ into 3 parts where each part forms a clique and any two consecutive parts form the complete bipartite subgraphs. Let n_1, m, n_2 be the numbers of vertices of the 3 parts respectively as shown in Figure 1.

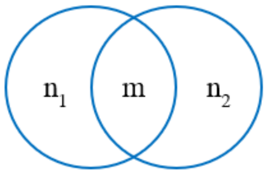

Figure 1 The 3 parts of the constructed graphs for the first group

From Figure 1, the constructed graphs consist of cliques $K_{n_1}, K_{n_1+m}, K_m, K_{m+n_2}, K_{n_2}$ where the clique number is

$$\omega = max\{n_1, n_1 + m, m, m + n_2, n_2\}.$$

For example, the graph with 10 vertices such that $n_1 = 3, m = 3, n_2 = 4$ is shown as Figure 2.

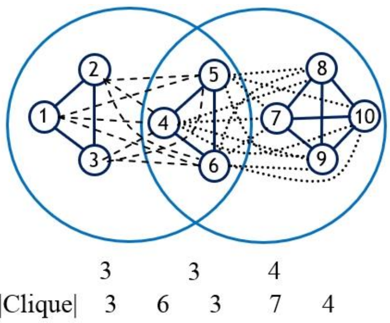

Figure 2 A graph of the first group with 10 vertices where $n_1 = 3, m = 3, n_2 = 4$.

From Figure 2, the graph contains 5 cliques $(K_3, K_6, K_3, K_7, K_4)$ and the clique number is $\omega = max\{3,6,3,7,4\} = 7$.

The constructed graphs used in this study are shown in Table 1.

Table 1 The constructed graphs of the first group

No.	n_1, m, n_2	**|Cliques|**	**|V|**	**|E|**	ω
1	3, 3, 4	3, 6, 3, **7**, 4	10	33	7
2	2, 3, 5	2, 5, 3, **8**, 5	10	35	8
3	7, 7, 6	7, **14**, 7, 13, 6	20	148	14
4	5, 7, 8	5, 12, 7, **15**, 8	20	150	15
5	10, 9, 11	10, 19, 9, **20**, 11	30	325	20
6	8, 12, 10	8, 20, 12, **22**, 10	30	355	22

For the second group of tested graphs, we divide the vertex set $\{1, 2, \ldots, N\}$ into 5 parts where each part forms a clique and any two consecutive parts form the complete bipartite subgraphs. Let n_1, m_1, n_2, m_2, n_3 be the numbers of vertices of the 5 parts respectively as shown in Figure 3.

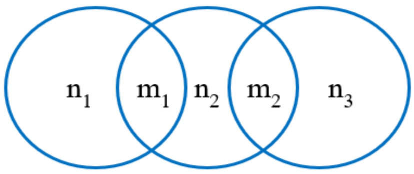

Figure 3 The 5 parts of the constructed graphs for the second group

From Figure 3, the graph consists of cliques $K_{n_1}, K_{n_1+m_1}, K_{m_1}, K_{m_1+n_2}, K_{n_2}, K_{n_2+m_2}, K_{m_2}, K_{m_2+n_3}, K_{n_3}$ where the clique number is $\omega = max\{n_1, n_1 + m_1, m_1,\ m_1 + n_2, n_2, n_2 + m_2, m_2, m_2 + n_3, n_3\}$.

For example, the graph with 10 vertices such that $n_1 = 3, m_1 = 2, n_2 = 1, m_2 = 2, n_3 = 2$ is shown as Figure 4.

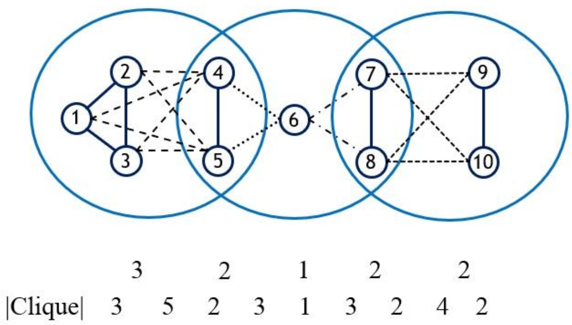

Figure 4 A graph of the second group with 10 vertices where $n_1 = 3, m_1 = 2, n_2 = 1, m_2 = 2, n_3 = 2$.

From Figure 4, the graph contains 9 cliques $(K_3, K_5, K_2, K_3, K_1, K_3, K_2, K_4, K_2)$ and the clique number is $\omega = max\{3, 5, 2, 3, 1, 3, 2, 4, 2\} = 5$.

In this work, we use the constructed graphs as shown in Table 2.

Table 2 The constructed graphs of second group

No.	n_1, m, n_2	|Cliques|	|V|	|E|	ω
1	3, 2, 1, 2, 2	3, **5**, 2, 3, 1, 3, 2, 4, 2	10	20	5
2	2, 1, 3, 3, 1	2, 3, 1, 4, 3, **6**, 3, 4, 1	10	24	6
3	1, 1, 1, 4, 3	1, 2, 1, 2, 1, 5, 4, **7**, 3	10	27	7
4	4, 4, 3, 5, 4	4, 8, 4, 7, 3, 8, 5, **9**, 4	20	94	9
5	4, 6, 2, 4, 4	4, **10**, 6, 8, 2, 6, 4, 8, 4	20	94	10
6	4, 2, 7, 4, 3	4, 6, 2, 9, 7, **11**, 4, 7, 3	20	99	11
7	6, 7, 5, 6, 6	6, **13**, 7, 12, 5, 11, 6, 12, 6	30	219	13
8	5, 7, 8, 5, 5	5, 12, 7, **15**, 8, 13, 5, 10, 5	30	235	15
9	5, 5, 6, 7, 7	5, 10, 5, 11, 6, 13, 7, **14**, 7	30	223	14

2.2 Benchmark graphs

We select 12 benchmark graphs to test the performance of the proposed DE algorithms. These graphs are shown in Table 3.

Table 3 The benchmark graphs

No.	Benchmark	|V|	|E|	ω
1	myciel3	11	20	2
2	chvátal	12	24	2
3	myciel4	23	71	2
4	queen5_5	25	160	5
5	1-Fulllns_3	30	100	3
6	queen6_6	36	290	6
7	2-Insertions_3	37	72	2
8	myciel5	47	236	2
9	queen7_7	49	476	7
10	3-Insertions_3	56	110	2
11	queen8_8	64	728	8
12	1-Insertions_4	67	232	2

Results and Discussion

The performances of the proposed DE algorithm for both constructed and benchmark graphs are presented as follows.

1. Constructed graphs

For the first group, the performance of the proposed DE algorithm is shown in Table 4. This table shows the minimum (Min) and maximum (Max) numbers of vertices in cliques, and the number of successful runs (NS) for each graph. The experimental results show that the proposed algorithm can find the maximum cliques of graphs with 10 vertices for all 20 runs. For graphs with 20 vertices, NS = 20 for graph (5, 7, 8) whereas NS = 19 for graph (7, 7, 6). For graphs with 30 vertices, NS = 20 for graph (8, 12, 10) whereas NS = 17 for graph (10, 9, 11). For the few unsuccessful runs, the algorithm gives the second best maximum cliques for all graphs.

Table 4 The performances of DE algorithm of the first group

No.	n_1, m, n_2	|Cliques|	|V|	|E|	ω	Min	Max	NS
1	3, 3, 4	3, 6, 3, **7**, 4	10	33	7	7	7	20
2	2, 3, 5	2, 5, 3, **8**, 5	10	35	8	8	8	20
3	7, 7, 6	7, **14**, 7, 13, 6	20	148	14	13	14	19
4	5, 7, 8	5, 12, 7, **15**, 8	20	150	15	15	15	20
5	10, 9, 11	10, 19, 9, **20**, 11	30	325	20	19	20	17
6	8, 12, 10	8, 20, 12, **22**, 10	30	355	22	22	22	20

For the second group, the performance of the proposed DE algorithm is shown in Table 5. The experimental results show that NS = 20 for all graph with 10 vertices. For graphs with 20 vertices, the algorithm can perform NS = 20 for all graphs except graph (4, 4, 3, 5, 4). For graphs with 30 vertices, the algorithm gives NS = 19, 12, 11 for graphs (5, 7, 8, 5, 5), (6, 7, 5, 6, 6) and (5, 5, 6, 7, 7), respectively. For the few unsuccessful runs, the algorithm gives the second best maximum cliques for all graphs.

Table 5 The performances of DE algorithm of the second group

No.	n_1, m_1, n_2, m_2, n_3	|Cliques|	|V|	|E|	ω	Min	Max	NS
1	3, 2, 1, 2, 2	3, **5**, 2, 3, 1, 3, 2, 4, 2	10	20	5	5	5	20
2	2, 1, 3, 3, 1	2, 3, 1, 4, 3, **6**, 3, 4, 1	10	24	6	6	6	20
3	1, 1, 1, 4, 3	1, 2, 1, 2, 1, 5, 4, **7**, 3	10	27	7	7	7	20
4	4, 4, 3, 5, 4	4, 8, 4, 7, 3, 8, 5, **9**, 4	20	94	9	8	9	15
5	4, 6, 2, 4, 4	4, **10**, 6, 8, 2, 6, 4, 8, 4	20	94	10	10	10	20
6	4, 2, 7, 4, 3	4, 6, 2, 9, 7, **11**, 4, 7, 3	20	99	11	11	11	20
7	6, 7, 5, 6, 6	6, **13**, 7, 12, 5, 11, 6, 12, 6	30	219	13	12	13	12
8	5, 7, 8, 5, 5	5, 12, 7, **15**, 8, 13, 5, 10, 5	30	235	15	13	15	19
9	5, 5, 6, 7, 7	5, 10, 5, 11, 6, 13, 7, **14**, 7	30	223	14	13	14	11

2. Benchmark graphs

For the benchmark graphs, the performance of the proposed DE algorithm is shown in Table 6. The experimental results show that the algorithm can find the maximum cliques of graphs with $|V| \leq 30$ for all 20 runs. For graphs with $|V| > 30$, the algorithm gives NS = 20 for 2-Insertions_3, myciel5, 3-Insertions_3, and 1-Insertions_4 whereas it gives NS = 18 for queen7_7 and queen8_8. The algorithm gives NS = 15 for queen6_6.

Table 6 The performances of DE algorithm of the benchmark graphs

No.	Benchmark	|V|	|E|	ω	Min	Max	NS
1	myciel3	11	20	2	2	2	20
2	chvátal	12	24	2	2	2	20
3	myciel4	23	71	2	2	2	20
4	queen5_5	25	160	5	5	5	20
5	1-FullIns_3	30	100	3	3	3	20
6	queen6_6	36	290	6	5	6	15
7	2-Insertions_3	37	72	2	2	2	20
8	myciel5	47	236	2	2	2	20
9	queen7_7	49	476	7	5	7	18
10	3-Insertions_3	56	110	2	2	2	20
11	queen8_8	64	728	8	7	8	18
12	1-Insertions_4	67	232	2	2	2	20

Conclusions

In this study, we solve the MCP which is transformed into constrained continuous optimization problem by using the modified DE algorithm. We test the performance of the algorithm with 15 constructed graphs and 12 benchmark graphs. The experimental results show that the proposed algorithm can find the maximum cliques for all 20 runs for 4 and 5 graphs of first and second constructed groups, and for 9 graphs of benchmark graphs. For the few unsuccessful runs, the algorithm gives the second best maximum cliques for all graphs. This shows that the proposed DE algorithm is effective for MCP.

Acknowledgements

The authors would like to thank the research capability enhancement program through graduate student scholarship, Faculty of Science, Khon Kaen University, for the financial support.

References

Balasundaram B and Butenko S. Graph domination, coloring and cliques in telecommunications. In Handbook of Optimization in Telecommunications. Springer US. 2006; 865-890.

Balasundaram B, Butenko S and Hicks IV. Clique relaxations in social network analysis: The maximum k-plex problem. Operations Research 2011; 59(1): 133-142.

Boginski V, Butenko S and Pardalos PM. Mining market data: a network approach. Computers & Operations Research 2006; 33(11): 3171-3184.

Bomze IM, Budinich M, Pardalos PM and Pelillo M. The maximum clique problem. In Handbook of combinatorial optimization Springer US 1999; 1-74.

Butenko S. Maximum independent set and related problems with applications. PhD thesis University of Florida USA. 2003.

Carraghan R and Pardalos PM. An exact algorithm for the maximum clique problem. Operations Research Letters 1990; 9: 375-382.

Dorndorf U, Jaehn F and Pesch E.Modelling robust flight gate scheduling as a clique partitioning problem. Transportation Science 2008; 42(3): 292-301.

Harary F and Ross IC. A Procedure for Clique Detection Using the Group Matrix. journal article American Sociological Association 1957; 20(3): 205-215.

Karp RM. Reducibility among combinatorial problems. Complexity of Computer Computations Plenum Press 1972; 85-103.

Motzkin TS and Straus EG. Maxima for graphs and a new proof of a theorem of turan. American Mathematical Society 1964; 11(382): 533-540.

Pattillo J, Youssef N and Butenko S. Clique relaxation models in social network analysis. In: M.T. Thai, P.M. Pardalos (eds), Handbook of Optimization in Complex Networks: Theory and Applications. Springer Optimization and Its Applications 2012; 58: 143-162.

Storn R and Price K. Differential Evolution - A simple and efficient adaptive scheme for global optimization over continuous spaces. Journal of Global Optimization 1997; 11: 341-59.

Weide O, Ryan D and Ehrgott M. An iterative approach to robust and integrated aircraft routing and crew scheduling. Computers & Operations Research 2010; 37(5): 833-844.