*Desalination and Water Treatment*
www.deswater.com

doi:10.5004/dwt.2019.24386

160 (2019) 214–218
August

# A new parallel algorithm to solve one classic water resources optimal allocation problem based on inspired computational model

Zuwen Ji[a], Zhaocai Wang[a,b,*], Anjun Deng[a], Wei Huang[c], Tunhua Wu[d,*]

[a]*State Key Laboratory of Simulation and Regulation of River Basin Water Cycle, China Institute of Water Resources and Hydropower Research, Beijing 100048, China, Tel. +86 10 68786631, Fax +86 10 68786631, email: jzw@iwhr.com, zuwenji70@163.com (Z. Ji), anjdeng@163.com (A. Deng)*
[b]*College of Information, Shanghai Ocean University, Shanghai 201306, China, Tel. +86 21 6190062, Fax +86 21 61900624, email: zcwang1028@163.com (Z. Wang)*
[c]*College of Mathematics and Information Science, Guiyang University, Guizhou, China, Tel. +86 851 85405556, Fax +86 851 85405556, email: 2533870668@qq.com (W. Huang)*
[d]*First Affiliated Hospital, Wenzhou Medical University, Wenzhou 325035, China, Tel. +86 577 86689913, Fax +86 577 86689913, email: appll188@163.com (T. Wu)*

Received 23 January 2019; Accepted 10 May 2019

**ABSTRACT**

The shortage of freshwater resources is a serious problem in the process of urbanization in the world, and even become the main constraint factor in some areas. In order to solve the problem, it is imperative to develop an effective, flexible and low-cost water resources management plan. Water resources optimal allocation is one of the hot topics in the field of water resources at present, such as water resources optimal *k*-edge cover problem. The *k*-edge cover problem aims to find an edge cover set with *k* edges in a given undirected graph. The efficient solution of this problem can play an important role in planning and setting up urban water resources network sites. Based on DNA molecular computing, the paper use a new parallel algorithm to solve *k*-edge cover problem with $O(n^2)$ time complexity, which greatly simplifies the computing complexity.

*Keywords:* DNA computation; Water resources optimal allocation problem; The *k*-edge cover problem; Adleman-Lipton model; NP-complete problem

## 1. Introduction

The problem of water resources optimal allocation is theoretically a multi-objective stochastic sequential decision making problem. It is to utilize the effective, fair and sustainable principles, to allocate limited and different forms of water resources scientifically and reasonably through a variety of measures in a specific basin or area. The water resources optimal allocation is the basis of the water resources rational utilization and the fundamental guarantee for the water resources sustainable utilization.

In fact, the water resources rational allocation, in a broad sense, is a study of how to use good water resources, including the development, utilization, protection and management of water resources. It is more urgent to carry out water resources rational allocation. The main reasons are as below: first, the natural spatial and temporal distribution of water resources does not adapt to the distribution of productive forces. Secondly, there have lots of water competitions between the regions and the various water use departments, and thirdly, the water resources development and utilization have caused many ecological environmental problems.

The rational allocation of water resources is a comprehensive system consisting of various measures. Its basic functions include two aspects: by adjusting the industrial structure, building a water-saving society, adjusting the

*Corresponding author.

distribution of productivity, restraining the momentum of water demand, and adapting to the more unfavorable conditions of water resources in demand, and coordinating competitive water use, strengthening management and changing water resources through engineering measures in the supply side. Natural space-time distribution is adapted to the distribution of productive forces. The two aspects complement each other in order to promote the sustainable development of the region. Rational allocation is the goal and desire when people allocate scarce resources. Generally speaking, the results of a reasonable allocation are not the best for the benefit or benefit of a certain individual, but the overall benefit or benefit of the whole resource allocation system is the best. The optimal allocation is the way and means people are looking for in reasonable allocation plan.

Since Adleman's pioneering work in Hamilton path problem [1], the research about DNA parallel computing has undergone rapid development process. Because of its advantages with parallelism and storage efficiency features, DNA parallel computing has been used to solve optimization NP-complete problems [2–18]. DNA computing refers to the process of solving the problems by producing and detecting a combination of a kind of mathematical process, which is based on DNA molecules parallel biochemical operations and processing techniques.

In general, DNA computing has three steps: (1) Generate DNA molecular data pools representing all possible solutions to the unsolved problem; (2) a series of biochemical experiments are carried out to eliminate the DNA solution chains which do not meet the logical requirements of the problem; (3) select the optimal solution chains for the problem. At present, DNA computing research involves many aspects, such as DNA computing capability, models and algorithms.

The *k*-edge cover problem, aiming to find the cover set with *k* edges, is a typical NP problem. For $G = (V,E)$, an edge cover means an edge set $S$ of $G$ such that $\forall\ v_i \in V$, then have at least one $e_{ij} \in S$ linking the vertex $v_i$. The *k*-edge cover problem is to find edge cover sets with *k* edges. For example, the undirected graph *G* in Fig. 1 is such a problem. There has no efficient algorithm to solve the problem by now. Based on the research of Adleman [1] and Lipton [2], we use a new bio-computing procedure to get the solutions of the *k*-edge cover problem.

The rest of the article is arranged as below. In the Section 2, we introduce the DNA computing Adleman-Lipton model and use the DNA molecular algorithm to solve *k*-edge coverage problem. Next, we describe feasibility and complexity of the algorithm in Section 3. Then, experimental results of simulated DNA computing are shown in Section 4. We come to the conclusion in Section 5.

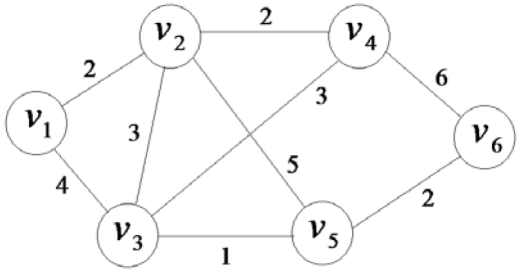

Fig. 1. An undirected graph G with 6 vertices and 9 edges.

## 2. DNA computing and DNA algorithm for the *k*-edge cover problem

Although biology and mathematics have their own complexity, in recent years, Mathematical Biology, as an interdisciplinary subject, has been intersecting from the two major fields of biology and mathematics. The operations in Adleman-Lipton model mainly include: *Merge* $(T_1,T_2)$, *Copy* $(T_1,T_2)$, *Separation* $(T_1,L,T_2)$, *Discard* $(T)$, *Read* $(T)$, *Sort* $(T_1,T_2,T_3)$ and *Append-tail* $(T,y)$ [19–22]. Meanwhile, the complexity of these operations is $O(1)$ [23–25].

The symbols $1$, $A_k$, $B_k$ $(k = 1,2,\ldots,n)$ denote distinct DNA singled strands with same length, say *t*-mer. We use distinct DNA singled strands $A_i1B_j$ $(1 \ i < j \ n)$ to denote the edge $e_{ij}$, with $A_i1B_j$ for $e_{ij}$ including in the edge set. In order to get the optimum solutions, we meantime design weight information string *X* with *t*-mer length, that the number of pasting is determined by the weight $w_{ij}$ of the edge $e_{ij}$. We use adjacency matrix of graph *G* denoting the connection relationship between different vertices, which $a_{ij} = 1$ means vertex $v_i$ links with $v_j$ and $a_{ij} = 0$ denotes connectionless. Taking Fig. 1 for example, the adjacency matrix $M(G)$ is as follows:

$$M(G) = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 \end{bmatrix} \tag{1}$$

As the simple undirected graph has no self-loops, so $a_{ii} = 0$ and $a_{ij} = a_{ji}$. In progress, we only need to check the numerical value of upper triangular matrix $M(G)$. Let $T_1 = \{\varnothing\}$.

(1) Generate all vertex set strands of graph *G*.
For $i = 1$ to $i = n - 1$
  For $j = i + 1$ to $j = n$
(1-1) *If* $(a_{ij} = 1)$
Then
(1-2) *Copy* $(T_1,T_2)$;
(1-3) *Append–tail* $(T_2,A_i1B_j)$;
(1-4) *Merge* $(T_1,T_2)$;
(1-5) *Discard* $(T_2)$;
Else
(1-6) Continue
  End for
End for
(2) Each single chain in the tube $T_1$ represents a possible set of edges after algorithm (1). As *k*-edge cover problem requires the edge cover set should contain *k* edges. So we pick out these strands with the *3kt*-mer length since we denote each edge as *3t*-mer length.
(2-1) *Selection* (*T1,3kt,T3*).

(3) Every singled chain in tube $T_3$ denotes a possible $k$ edges set so far. Meanwhile, the edge cover problem requires that arbitrary vertex should connect at least one edge of set. Therefore, we should check whether all sets satisfy the constraints. For $\forall v_i \in V$ in graph, we should keep the strands with edge symbols $A_i1$ and $1B_i$ synchronously.

For $i = 1$ to $i = n$
(3-1) *Copy* ($T_3$, $T_4$);
(3-2) *Separation* ($T_3$, {$A_i1$}, $T_5$);
(3-3) *Separation* ($T_4$, {$1B_i$}, $T_6$);
(3-4) *Merge* ($T_5$, $T_6$);
(3-5) *Copy* ($T_5$, $T_3$);
(3-6) *Discard* ($T_4$);
(3-7) *Discard* ($T_5$);
(3-8) *Discard* ($T_6$).
End for

(4) The $k$-edge cover problem should be an edge cover set with minimum weight value. If edge $e_{ij}$ in the edge cover set, we append additional weight strand $X$ with $w_{ij}$ times in order to get the corresponding strand solutions.

For $i = 1$ to $i = n - 1$
  For $j = i + 1$ to $j = n$
(4-1) *Separation* ($T_3$, {$A_i1B_j$}, $T_7$);
(4-2) *Append-tail* ($T_7$, *XX…X*);
(4-3) *Merge* ($T_3$, $T_7$);
(4-4) *Discard* ($T_7$).
  End for
End for

(5) We find the shortest DNA chains by sorting, which indicate the optimal solutions of the problem.
(5-1) *Sort* ($T_3$, $T_8$, $T_9$);
(5-2) *Read* ($T_8$).

## 3. The complexity and feasibility of the proposed DNA algorithm

### 3.1. Theorem

The solutions of $k$-edge cover problem having $n$ vertices can be got in $O(n^2)$ time complexity by biological molecules parallel computing.

### 3.2. Proof

We first generate arbitrary edges sets strands in $T_1$ at Algorithm (1). Then, after Algorithm (2), all the singled strands in tube $T_3$ denote all possible $k$-edges set. Then the strands can be described:

$$A_{i_1}1B_{j_1}A_{i_2}1B_{j_2}...A_{i_k}1B_{j_k} \qquad i_k, j_k \in \{1,2,...,n\} \tag{2}$$

Next, we choose the edge cover set strands and append the weight strands at the end of corresponding strands at Algorithm (3) and Algorithm (4). S can be supposed as the strands after the Algorithm (4). Then they can be described:

$$A_{i_1}1B_{j_1}A_{i_2}1B_{j_2}...A_{i_k}1B_{j_k}XX...XX \tag{3}$$

And we reasonably design the length of $A_k$, $B_k$, $1$ and $X$, for $||A_k|| = ||B_k|| = ||1|| = ||\#|| = t$. We suppose $\max\left(w_{i_p j_p}\right) = L$, so

$$\begin{aligned}\|S\| &= \left\|A_{i_1}\right\| + \|1\| + \left\|B_{j_1}\right\| + \left\|A_{i_2}\right\| + \|1\| + \left\|B_{j_2}\right\| + \\ &... + \left\|A_{i_k}\right\| + \|1\| + \left\|B_{j_k}\right\| + \|X\| + ... + \|X\| = \\ &\sum_{p=1}^{k}\left\|A_{i_p}\right\| + \sum_{p=1}^{k}\|1\| + \sum_{p=1}^{k}\left\|B_{j_p}\right\| + \left(w_{i_1 j_1} + w_{i_2 j_2} + ... + w_{i_k j_k}\right)t \\ &= 3kt + \left(w_{i_1 j_1} + w_{i_2 j_2} + ... + w_{i_k j_k}\right)t\end{aligned} \tag{4}$$

$$\max\left(w_{i_p j_p}\right) = L \tag{5}$$

$$3kt \le \|S\| \le 3kt + kL \tag{6}$$

So the solutions strands of Algorithm (5) can be found in appropriate length range between $3kt$ and $3kt + kL$.

In addition, the calculation of DNA algorithm time complexity $T$ is as below:

$T$(*Algorithm* (1)) = $O(6n(n-1)/2) = O(n^2)$;
$T$(*Algorithm* (2)) = $O(1)$;
$T$(*Algorithm* (3)) = $O(8n) = O(n)$;
$T$(*Algorithm* (4)) = $O(4n(n-1)/2) = O(n^2)$;
$T$(*Algorithm* (5)) = $O(2) = O(1)$;
$T = T$(*Algorithm* (1)) + $T$(*Algorithm* (2)) + $T$(*Algorithm* (3)) + $T$(*Algorithm* (4)) + $T$(*Algorithm* (5))
$= O(n^2) + O(1) + O(n) + O(n^2) + O(1)$
$= O(n^2)$

## 4. Experimental results of simulated DNA computing

DNA-based calculations depend on the biochemical operations of molecules, which may lead to errors in the application of these biochemical operations. Therefore, how to design DNA sequences is an important issue to ensure the reliability of DNA computation. In order to make good performance in hybridization reactions, we learn from the sequence design methods in reference [34–40].

In this paper, we use BioPython, a computational molecular biology tool, as our development platform to generate good DNA sequences suitable for performing our algorithms in the laboratory. The Braich's programs run on Windows XP systems, with Intel kernel CPU and 8 GB main memory, and Visual C++ compiler. The coding programs are used to generate DNA sequences for solving the k-edge cover problem, and construct the DNA sequences of each bit in the library. Meanwhile, when a new DNA sequence is added, the program determines whether the DNA strand meets the constraints listed in the reference [34–41]. If the generated DNA sequence cannot satisfy the constraints, the program will regenerate the new DNA sequence. If the restriction condition is satisfied, the DNA sequence is accepted. When all DNA strands satisfy these constraints, the program stops, and these sequences are the output.

Taking 3-edge cover problem in Fig. 1 for an example, the program generates 4-base random sequences, consisting of $A_i$, $B_i$, $1$ and $X$ shown in Table 1. Edge DNA sequences

Table 1
Sequences chosen to represent $A_i$, $B_i$, *1* and *X* ($i \in \{1,2,\ldots,6\}$) for the 3-edge cover proble

| Bit | 3'–5'DNA sequence | Bit | 3'–5'DNA sequence |
|---|---|---|---|
| $A_1$ | AGAT | $B_1$ | CGCA |
| $A_2$ | TGAT | $B_2$ | ATTG |
| $A_3$ | GTGT | $B_3$ | TCTG |
| $A_4$ | TGAG | $B_4$ | GGCT |
| $A_5$ | AGCG | $B_5$ | CAGT |
| $A_6$ | TGCT | $B_6$ | AGTC |
| *1* | GACA | *X* | CCTA |

generated by Braich's program are shown in Table 2. And we also calculate the enthalpy, entropy, and free energy for binding of each probe to its corresponding region on a library strand using the Braich's program, while the energy used is shown in Table 3.

We also calculate the mean and standard deviation of enthalpy, entropy and free energy in all probe/library interactions. The energy level is shown in Table 4. Table 5 gets solutions strands of the 3-edge cover problem.

Table 2
Sequences chosen to represent the elements $A_i1B_j$ ($1 \le i < j \le 6$) for the 3-edge cover problem

| Bit | 3′–5′DNA sequence | Bit | 3′–5′DNA sequence |
|---|---|---|---|
| $A_{11}B_2$ | AGATGACAATTG | $A_{11}B_3$ | AGATGACATCTG |
| $A_{21}B_3$ | TGATGACATCTG | $A_{21}B_4$ | TGATGACAGGCT |
| $A_{21}B_5$ | TGATGACACAGT | $A_{31}B_4$ | GTGTGACAGGCT |
| $A_{31}B_5$ | GTGTGACACAGT | $A_{41}B_6$ | TGAGGACAAGTC |
| $A_51B_6$ | AGCGGACAAGTC | | |

Table 3
The energies for of binding each probe to its corresponding region on a library strand

| Edges | Enthalpy energy H | Entropy energy S | Free energy G |
|---|---|---|---|
| $A_{11}B_2$ | 108.7 | 278.5 | 24.6 |
| $A_{11}B_3$ | 104.6 | 267.7 | 23.1 |
| $A_{21}B_3$ | 103.5 | 261.2 | 22.7 |
| $A_{21}B_4$ | 101.3 | 254.3 | 22.3 |
| $A_{21}B_5$ | 107.4 | 270.1 | 24.5 |
| $A_{31}B_4$ | 99.7 | 250.6 | 23.3 |
| $A_{31}B_5$ | 105.8 | 270.8 | 23.9 |
| $A_{41}B_6$ | 107.1 | 272.4 | 24.2 |
| $A_{51}B_6$ | 102.2 | 261.8 | 22.8 |

Table 4
The energies over all probe/library strand interactions

| | Enthalpy energy H | Entropy energy S | Free energy G |
|---|---|---|---|
| Average | 104.478 | 265.267 | 23.489 |
| Standard Deviation | 3.0429 | 9.0128 | 0.8388 |

## 5. Conclusions

This paper introduces a new DNA computing algorithm to solve *k*-edge cover problem. The DNA algorithm has two merits. First of all, the aforementioned algorithm has a lower hybrid error rate, as we utilize reasonable DNA sequences to produce solutions. Secondly, the DNA algorithm can deal with the *k*-edge cover problem having *n* vertices in the $O(n^2)$ time complexity and compare exponential time complexity with the electronic computer, such as $2^{O(k)}n^{O(1)}$ [29], $O(2.3147^k)$ [30], $O(n^3)$ [32] and so on. Many scholars believe that the study of DNA computing lays the foundation for human development of molecular computers. The intelligent system based on DNA computing will set up a bridge between DNA computing and intelligent system research, and will play a stepping stone role in DNA intelligent computers and other related research. The integration of DNA computing and soft computing will provide a good method to realize DNA intelligent computers. DNA intelligent computer can completely solve the intelligent computing function that the existing computer can't achieve [26–28].

## Acknowledgements

It was supported by National Key Research and Development Program of China (grant No. 2017YFC0405501) and the National natural science foundation of China (grant No. 51879282) and the Open Research Fund of State Key Laboratory of Simulation and Regulation of Water Cycle in River Basin, China Institute of Water Resources and Hydropower Research (grant No. SKL2018ZY07, IWHR-SKL-KF201905). The project was also financially supported by the Education Department of Guizhou province science and technology talents support programs (KJH KY[2016]088).

## References

[1] L.M. Adleman, Molecular computation of solution to combinatorial problems, Science, 266 (1994) 1021–1024.
[2] R.J. Lipton, DNA solution of HARD computational problems, Science, 268 (1995) 542–545.

Table 5
DNA sequences chosen to represent the solutions to the 3-edgecover problem in Fig. 1

| $e_{ij}$ | $e_{12}$,$e_{34}$,$e_{56}$ |
|---|---|
| Solutions strands | 3′–AGATGACAATTGGTGTGACAGGCTAGCGGACAAGTCCCTACCTACCTACCTACCTACCTACCTA– 5′ |

[3] S. Roweis, E. Winfree, R. Burgoyne, N.V Chelyapov, M.F. Goodman, P.W.K. Rothemund, L.M. Adleman, A sticker based model for DNA computation, J. Comput. Biol., 5(4) (1998) 615–629
[4] Q. Ouyang, P.D. Kaplan, S. Liu, A. Libchaber, DNA solution of the maximal clique problem, Science, 278 (1997) 446–449.
[5] E. Winfree, F. Liu, L.A. Wenzler, N.C. Seeman, Design and self assembly of two dimensional DNA crystals, Nature, 394 (1998) 539–544.
[6] J.F. Ren, Y.Z. Zhang, G. Sun, The np-hardness of minimizing the total late work on an unbounded batch machine, Asia Pac. J. Oper. Res., 26(3) (2009) 351–363.
[7] D.M. Xiao, W.X. Li, Z.Z. Zhang, L. He, Solving maximum cut problems in the Adleman-Lipton model, BioSystems, 82 (2005) 203–207.
[8] W.X. Li, D.M. Xiao, L. He, DNA ternary addition, Appl. Math. Comput., 182 (2006) 977–986.
[9] D.M. Xiao, W.X. Li, J. Yu, X.D. Zhang, Z.Z. Zhang, L. He, Procedures for a dynamical system on $\{0,1\}^n$ with DNA molecules, BioSystems, 84 (2006) 207–216.
[10] C. Wang, J. Zhou, X. Xu, Saddle points theory of two classes of augmented Lagrangians and its applications to generalized semi-infinite programming, Appl. Math Opt., 59(3) (2009) 413–434.
[11] Z. Wang, D. Huang, H. Meng, C. Tang, A new fast algorithm for solving the minimum spanning tree problem based on DNA molecules computation, BioSystems, 114(1) (2013) 1–7.
[12] M.Y. Guo, W.L. Chang, M. Ho, J. Lu, J.N. Cao, Is optimal solution of every NP-complete or NP-hard problem determined from its characteristic for DNA-based computing, BioSystems, 80 (2005) 71–82.
[13] H. Zhang, Y. Wang, A new CQ method for solving split feasibility problem, Front Math China, 5(1) (2010) 37–46.
[14] L. Qi, X. Tong, Y. Wang, Computing power system parameters to maximize the small signal stability margin based on min-max models, Optim Eng., 10(4) (2009) 465–476.
[15] Z.C. Wang, J. Tan, D.M. Huang, Y.C. Ren, Z.W. Ji, A biological algorithm to solve the assignment problem based on DNA molecules computation, Appl. Math Comput., 244 (2014) 183–190.
[16] Z. Wang, D. Huang, J. Tan, T. Liu, K. Zhao, L. Li, A parallel algorithm for solving the n-queens problem based on inspired computational model, BioSystems, 131(5) (2015) 22–29.
[17] X.C. Liu, X.F. Yang, S.L. Li, Y. Ding, Solving the minimum bisection problem using a biologically inspired computational model, Theor. Comput. Sci., 411 (2010) 888–896.
[18] Z. Wang, J. Pu, L. Cao, J. Tan, A parallel biological optimization algorithm to solve the unbalanced assignment problem based on DNA molecular computing, Int. J. Mol Sci., 16(10) (2015) 25338–25352.
[19] G. Wang, X.X. Huang, J. Zhang, Levitin-Polyak well-posedness in generalized equilibrium problems with functional constraints, Pac. J. Optim., 6(2) (2010) 441–453.
[20] M.R. Garey, D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-completeness., W. H. Freeman and Company, 1979.
[21] M. Yamamura, Y. Hiroto, T. Matoba, Solutions of shortest path problems by concentration control, Lecture Notes Comp. Sci., 2340 (2002) 231–240.
[22] R.S. Braich, C. Johnson, P.W.K. Rothemund, D. Hwang, N. Chelyapov, L.M. Adleman, Solution of a satisfiability problem on a gel-based DNA computer, in: Proceedings of the Sixth International Conference on DNA Computation (DNA 2000), Lecture Notes Comp. Sci., 2054 (2001) 27–42.
[23] H. Chen, Y. Wang, A Family of higher-order convergent iterative methods for computing the Moore–Penrose inverse, Appl. Math Comput., 218(8) (2011) 4012–4016.
[24] R.B.A. Bakar, J. Watada, W. Pedrycz, DNA approach to solve clustering problem based on a mutual order, BioSystems, 91 (2008) 1–12.
[25] Z. Wang, Z. Ji, Z. Su, X. Wang, K. Zhao, Solving the maximal matching problem with DNA molecules in Adleman-Lipton model, Int. J. Biomath., 9(2) (2016) 1650019.
[26] Y. Ni, Fuzzy minimum weight edge covering problem, Appl. Math Model, 32(7) (2008) 1327–1337.
[27] H. Fernau, F.V. Fomin, G. Philip, On the parameterized complexity of vertex cover and edge cover with connectivity constraints, Theor. Comput. Sci., 565 (2015) 1–15.
[28] Z. Wang, Z. Ji, X. Wang, A new parallel DNA algorithm to solve the task scheduling problem based on inspired computational model, BioSystems, 162 (2017) 59–65.
[29] F.V. Fomin, D. Lokshtanov, V. Raman, S. Saurabh, Subexponential algorithms for partial cover problems, Inform. Process Lett., 111(16) (2011) 814–818.
[30] M. Xiao, T. Kloks, S.H. Poon, New parameterized algorithms for the edge dominating set problem, Theor. Comput. Sci., 511 (Complete) (2013) 147–158.
[31] C.P. Wei, P. Wang, Y.Z. Zhang, Entropy similarity measure of interval-valued in tuition is tic fuzzy sets and their applications, Inform Sciences, 181(19) (2011) 4273–4286.
[32] J. Chen, L. Kou, X. Cui, An approximation algorithm for the minimum vertex cover problem, Procedia Engineering, 137 (2016) 180–185.
[33] B. Liu, B. Qu, N. Zheng, A successive projection algorithm for solving the multiple-sets split feasibility problem, Numer. Func. Anal Opt., 35(11) (2014) 1459–1466.
[34] N. Zhao, C. Wei, Z. Xu, Sensitivity analysis of multiple criteria decision making method based on the OWA operator, Int. J. Intell. Syst., 28(11) (2013) 1124–1139.
[35] Q. Liu, A. Liu, Block SOR methods for the solution of indefinite least squares problems, Calcolo., 51(3) (2014) 367–379.
[36] J. Ren, G. Sun, Y. Zhang, The supplying chain scheduling with outsourcing and transportation, Asia Pac. J. Oper. Res., 34(2) (2017) 1750009.
[37] B. Wang, A. Iserles, X. Wu, Arbitrary-order trigonometric fourier collocation methods for multi-frequency oscillatory systems, Found Comput Math., 16(1) (2016) 151–181.
[38] C. Miao, Y. Zhang, Z. Cao, Bounded parallel-batch scheduling on single and multi machines for deteriorating jobs, Inform. Process Lett., 111(16) (2011) 798–803.
[39] G. Wang, Levitin–Polyak Well-Posedness for optimization problems with generalized equilibrium constraints, J. Optimiz Theory App., 153(1) (2012) 27–41.
[40] W. Liu, C. Wang, A smoothing Levenberg–Marquardt method for generalized semi-infinite programming, Comput. Appl. Math, 32(1) (2013) 89–105.
[41] Z. Ji, Z. Wang, X. Bao, X. Wang, T. Wu, Research on water resources optimal scheduling problem based on parallel biological computing, Desal. Water Treat., 111 (2018) 88–93.