The Quadratic Shortest Path Problem and its Genetic Algorithm

Linzhong LIU[1,2], Bing WU[3] Yinzhen LI[1], Jianning YU[1]
1. Institute of System Engineering, Lanzhou Jiaotong University, Lanzhou 730070, China
2. Department of Mathematical Sciences, Tsinghua University, Beijing 100084, China
3. The West Logistic Corporation Limited of Gansu, Lanzhou 730000, China
Email: lzliu@orsc.edu.cn

Abstract: The quadratic shortest path (QSP) problem is to find a path from a node to another node in a given network such that the total cost includes two kinds of costs, say direct cost and interactive cost, is minimum. The direct cost is the cost associated with each arc and the interactive cost occurs when two arcs appear simultaneously in the shortest path. In this paper, the concept of the quadratic shortest path is initialized firstly. Then a spanning tree-based genetic algorithm is designed for solving the quadratic shortest path problem. Finally, a numerical example is given.

Keywords: graph; network; shortest path; genetic algorithm.

I. Introduction

The shortest path (SP) problem has a long history in the combinatorial optimization. Since the SP problem was initialized, it was studied by many researchers and has been applied in wide varieties. The classical SP is to find the shortest, namely the least cost, path from the origin node to the destination node in a given network, where the cost of the path is defined as the sum of the costs of the arcs in the path and the costs are assumed to be non-negative.

There are varieties of extensions both in models and algorithms of SP after is was initialized and thousands papers with respect to various SPs and algorithms in recent years can be retrieved by a quickly searching in the web. For example, Balachandhrm and Randan [2], Pettie [20] studied the all pairs SP; Pattanamekar and Park etc.[3], Frigioni and Marchetti-Spaccamela [5] investigated the dynamic and stochastic SP; Davies and Lingrascite [6] designed a genetic algorithms for the dynamic and stochastic SP; Bousono-Calzon and Figuiras-Vidal [7] explored an algorithm by using the Hopfield neural networks; Miller-Hooks and Mahmassanicite[8] studied SP in stochastic time-varying networks; Chen and Tang [9] introduced minimum time paths in a networks with mixed time constraints; Granat and Guerriero [10], Modesti and Sciomachen [11] gave the algorithms of the multi-criteria shortest path, respectively; Chen and Yang [19] introduce the algorithm for finding the first k shortest paths in a time-window network; Bose, Maheshwari and Narasimhan [21] initialized the bottleneck shortest paths; Okada [22] studied the SP with fuzzy arc lengths; Godor, Harmatos and Juttner [24] initialized the inverse SP and Daganzo [25] the time-dependent SP. Some of them are listed in the references [2]–[26]. Some of the extensions are NP-hard. As another extension of the SP, we initialize the quadratic shortest path (QSP) problem and formulate the QSP as a quadratic 0–1 programming in this paper.

This paper is organized as follows. Firstly, we initialize the concept of quadratic shortest path (QSP) problem in the Section . Then a spanning tree-based heuristic genetic algorithm is given in Section by using heuristic approach and a numerical example is given in Section **??**.

II. The Preliminaries and the Mathematical Model of QSP

Firstly, let us recall some preliminaries and define some notations. Throughout this paper, all the graphs are simple direct graph. Let G be a graph with $E(G)$ and $V(G)$ as its arcs set and nodes set, respectively. Assume that $E = E(G) = \{e_1, e_2, \cdots, e_{|E|}\}$, $V = V(G) = \{v_1, v_2, \cdots, v_{|V|}\}$, $m = |E|$, $n = |V|$ and the ordering pair (u, v), $u, v \in V(G)$ denote the arc from node u to node v. For a node $v \in V(G)$, $d_G(v)$ denotes its degree and $N^+[v]$, $N^-[v]$, $N_e^+[v]$ and $N_e^-[v]$ denote the sets $\{u|(u, v) \in E(G)\}$, $\{u|(v, u) \in E(G)\}$, $\{(u, v)|(u, v) \in E(G)\}$ and $\{(v, u)|(v, u) \in E(G)\}$, respectively. For a subset S of E or V, $G[S]$ and $|S|$ denote the subgraph induced by the subset S and the cardinality of the set S, respectively. Additionally, for a subgraph T of G and an arc $e = (u, v) \in E(G)$, $T + e$ or $T + (u, v)$ denotes the graph obtained by adding arc $e = (u, v)$ to T and $T - e$ or $T - (u, v)$ denotes the graph obtained by removing arc $e = (u, v)$ from T.

The QSP is to find a path from a node to another

Proceedings of the Fifth International Conference on Electronic Business, Hong Kong, December 5-9, 2005, pp. 908 - 913.

node in a given network such that the total cost includes two kinds of costs, say direct cost and interactive cost, is minimum. The direct cost is the cost associated with each arc and the interactive cost occurs when two arcs appear simultaneously in the shortest path. Let d_i and w_{ij} be the direct cost of the arc $e_i \in E$ and the interactive cost of arcs $e_i, e_j \in E(G)$, respectively, $i, j = 1, 2, \cdots, |E(G)|$. Assume that the required path P is the path from the node 1 to the node $|V|$ in this paper. The decision variables $x_i = 0$ or 1, $i = 1, 2, \cdots, |E|$, indicate that each arc i is either in the path or not and $\boldsymbol{x}$ is the vector consists of x_i, $i = 1, 2, \cdots, |E|$. Then the path P can also be denoted by such a vector $\boldsymbol{x}$ and the cost function of the path $\boldsymbol{x}$ can be defined as

$$f(\boldsymbol{x}) = \sum_{i=1}^{|E|} d_i x_i + \sum_{i=1}^{|E|} \sum_{j=i+1}^{|E|} w_{ij} x_i x_j. \tag{1}$$

For convenience, we briefly take the index k of arc e_k or node v_k to denote the kth arc e_k in arcs set E or the kth node v_k in the nodes set V in the following process. Then the mathematical model of QSP can be formulated as follows:

$$\begin{aligned} \max\ & f(\boldsymbol{x}) = \sum_{i=1}^{|E|} d_i x_i + \sum_{i=1}^{|E|} \sum_{j=i+1}^{|E|} w_{ij} x_i x_j \\ \text{s.t.}\ & \sum_{i \in N_e^+[v]} x_i - \sum_{j \in N_e^-[v]} x_j = d, \quad v \in V(G), \\ & x_i = 0 \text{ or } 1, \quad i = 1, 2, \cdots, |E|, \end{aligned} \tag{2}$$

where

$$d = \begin{cases} 1, & \text{if } v = 1, \\ 0, & \text{if } v \neq 1 \text{ or } |V|, \\ -1, & \text{if } v = |V|. \end{cases}$$

It is clear that the model (2) of QSP is quadratic 0–1 programming. It is well-known that the quadratic programming is NP-hard, especially the quadratic 0–1 programming. So we will focus on the algorithm designing of QSP in the remain parts of this paper. It is also generally accepted that the genetic algorithm is an efficient ways to solve the NP-hard problems. For the SP, the related exposition related to SP is initialized by Cheng and Gen [23]. To design the genetic algorithm of QSP, we introduce a Lemma as follows.

Lemma 0.1 *[27] Let G be a connected simple graph and T be a spanning tree of the graph G. If $(u, v) \in E(G)$ satisfies $(u, v) \notin E(T)$, then there is a unique cycle in $T + (u, v)$.*

On the basis of Lemma 0.1, we design a spanning tree-based genetic algorithm for solving the model (2) in the next Section.

The encode method and evolution operation are the universal problems in the current genetic algorithms for the SP and its extensions. Firstly, different paths from the origin node to the destination node include a variable number of the nodes and a random arcs series usually does not correspond to a path. Furthermore, the evolution operation such as the crossover and mutation operations can't be performed efficiently because two paths usually haven't had any common nodes or arcs except the end node 1 and n of the paths. For example, Cheng and Gen [23] have designed a comprise approach-based genetic algorithm of the multi-criteria SP. In Cheng and Gen's algorithm, the chromosome is encoded as the path from the origin node to the destination node. In next Section, we explore a completely new encode method and evolution operators.

III. Genetic Algorithm

In the following process, we shall design the representation, crossover, mutation and selection operation of the spanning tree-based genetic algorithm for solving the QSP. As stated in the Section , the encode method and evolution operation are the universal problems in the current genetic algorithm for the SP. Aim at the purpose to solve these problems, we shall design the oriented tree-based crossover, mutation and evaluation operation by encoding the chromosome as a spanning tree. By such an encode method, there is a unique path from the origin node to the destination node and by using some spanning tree-based heuristic evolution operations, the algorithm completely overcome the defects of the current genetic algorithm with respect to SP. Additionally, the merits of such an encode method are that the chromosomes produced by the crossover and mutation are also feasible.

III. 1 Representation

Assume that $m = |E|$ and $n = |V|$. There are many ways to represent a solution of the optimization problem in genetic algorithm. In this paper, we employ a spanning tree-based encoding method to define the chromosomes of the QSP. We employ a spanning tree X which is denoted by an arcs set with fixed set cardinality $n - 1$ to encode a chromosome of the QSP, where $X = \{x_1, x_2, \cdots, x_n\}$ and each x_i represents an arc in the spanning tree and its value is equal to the index of the corresponding arc. It is clear that such a chromosome includes a unique path from the node 1 to the node n.

In the evolution process, we must perform some network search operation such as finding the cycle in the graph $T + e$ or the unique path in T from the node 1 to the node n, where T is a chromosome (tree) and $e \in E(G) \setminus E(T)$. So the chromosome is converted to an oriented tree when perform the crossover and mutation operations. Clearly, It is very convenient to find such a

cycle or path by using the oriented tree.

Definition 0.1 *The weight of the chromosome X, denoted by $W(X)$, is defined as*

$$\sum_{i\in P} d_i x_i + \sum_{i=1}^{|E|}\sum_{j=i+1}^{|E|} w_{ij}x_i x_j,$$

where P is the unique path from the node 1 to the node n in X. The optimal chromosome is such a chromosome X that the weight of $W(X)$ is optimal.

By such an encoding method, the crossover and mutation operations are really a set-based genetic operations and the decoding process are also convenient to be realized by finding the path from the node 1 to the node n in the chromosome X.

III. 2 Initialization Process

The initialization is an important process in genetic algorithm serving as the role of initializing the chromosomes. We take *pop_size* to denote the number of chromosomes. We initialize chromosomes X_1, X_2, $\cdots$, X_{pop_size} by repeating the following algorithm *pop_size* times, where T denotes a spanning tree of the graph $G(V, E)$.

The initialization algorithm:

Step 1. Set $T = \emptyset$. Randomly select nodes $u_0 \in V(G)$, $v_0 \in N(u_0)$. Set $T \leftarrow T + (u_0, v_0)$.

Step 2. Randomly select a node $u \in V(T)$ with $N(u) \setminus V(T) \neq \emptyset$. Randomly select a node $v \in N(u) \setminus V(T)$. Set $T \leftarrow T + (u, v)$.

Step 3. Repeat Step 2 until $|E(T)| = |V(G)| - 1$.

Step 4. Set $X_i \leftarrow T$ and stop.

Theorem 0.1 *The obtained tree in the initialization algorithm is a spanning tree.*

Proof: Because the arc (u, v) in the Step 2 has a common node u with $V(T)$, the obtained graph T is obviously a connected graph. Additionally, the node v in the Step 2 is selected from $N(u) \setminus V(T)$, this implies that T is cycle free. Therefore, a spanning tree is obtained when $|E(T)| = |V(G)| - 1$.

III. 3 Crossover Operation

Let $P_c \in (0, 1)$ be the crossover probability. In order to determine the parents for the crossover operation, we repeat the following process *pop_size* times: randomly generating a real number r from interval $(0, 1]$, the chromosome X_i is selected to crossover if $r < P_c$. We denote the selected parents by X'_1, X'_2, $\cdots$ and divide them into the following pairs:

$$(X'_1, X'_2), \quad (X'_3, X'_4), \quad (X'_5, X'_6), \ \cdots$$

Based on Lemma 0.1, we take the chromosomes pair (X'_1, X'_2) as the example to illustrate the crossover process. The Figure 1 is a numerical example. Randomly generate an integer k from interval $[1, t]$, where

$$t = \left|E(X'_1) \setminus (E(X'_1) \bigcap E(X'_2))\right| = \left|E(X'_2) \setminus (E(X'_1) \bigcap E(X'_2))\right|.$$

After that, randomly select two sets $S_1 \subset E(X'_1) \setminus (E(X'_1) \bigcap E(X'_2))$ and $S_2 \subset E(X'_2) \setminus (E(X'_1) \bigcap E(X'_2))$ with cardinality k. Follows from Lemma 0.1, for each arc $e \in S_1$ (or $e \in S_2$), there is a unique cycle, says C_e, in $X'_2 + \{e\}$ (or $X'_1 + \{e\}$) with the arc e on it. Then we can perform crossover operation as follows: randomly select an arc $e' \in E(C_e)$ and set $X'_2 \leftarrow X'_2 - \{e'\} + \{e\}$ (or $X'_1 \leftarrow X'_1 - \{e'\} + \{e\}$)(Figure 2, where (a) and (b) are X'_1 and X'_2 before crossover, (c) and (d) are these after crossover).

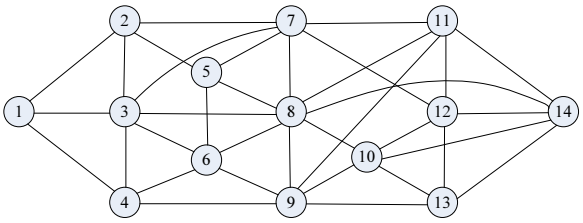

Figure 1: A numerical example with 14 nodes and 35 arcs

Crossover algorithm:

Step 1. Convert the chromosome X'_1 and X'_2 to the oriented trees T_1 and T_2.

Step 2. Set $t = |E(X'_1) \setminus (E(X'_1) \bigcap E(X'_2))| = |E(X'_1) \setminus (E(X'_2) \bigcap E(X'_2))|$.

Step 3. Randomly generate an integer k from interval $[1, t]$.

Step 4. Randomly select two sets S_1 and S_2:

$$S_1 \subset E(X'_1) \setminus (E(X'_1) \bigcap E(X'_2)),$$

$$S_2 \subset E(X'_2) \setminus (E(X'_1) \bigcap E(X'_2)),$$

$$|S_1| = |S_2| = k.$$

Step 5. For every arc $e \in S_1$, Find the unique cycle C_e in $T_2 + \{e\}$. Randomly select an arc $e' \in E(C_e)$, $e' \neq e$. Set $X'_2 \leftarrow X'_2 - \{e'\} + \{e\}$.

Step 6. Perform the same operations as Step 5 between T_1 and S_2.

For a spanning tree T, if we indicate a node as the root of T, then the spanning tree can be converted to an oriented spanning tree by using the following method. Let P be a path from the root to a node v. The direction of the path P is defined as the direction go away from the root to the node v in the path P and the direction of an arc in $E(P)$ is defined as the same direction of the path P. By such a way, an oriented spanning tree is defined. Based on such an oriented tree T, we design an algorithm to find the cycle C_e in the Step 5 of the crossover algorithm. For arc $e = (u, v)$, $P_{[s,t]}$ denotes the segment in the path P between the vertices s and t.

The algorithm to find the cycle C_e:

Step 1. Start from the node u, along the predecessor node of u, find the path P^u from the root to the node u. If $v \in V(P^u)$, the cycle C_e is found and set $C_e = P^u_{[u,v]} + \{e\}$, stop. Otherwise, go to next step.

Step 2. Start from the node v, along the predecessor node of v, find the path P^v from the root to the node v. If $u \in V(P^v)$, the cycle C_e is found and set $C_e = P^v_{[u,v]} + \{e\}$, stop. Otherwise, go to next step.

Step 3. Start from the root node, along the path P^v or P^u to find the the last common node s of the path P^v and P^u, set $C_e = P^u_{[u,s]} + P^v_{[s,v]} + \{e\}$.

It is clear that the new chromosome obtained by such crossover operation is also a feasible.

III. 4 Mutation Process

In this section, we will design an oriented tree and knowledge-based heuristic mutation operator. Let $P_m \in (0, 1)$ be the mutation probability. We employ the following operator to select the chromosome to be mutated: for $i = 1$ to *pop_size*, randomly generate a random number r from interval $(0, 1)$; if $r \leq P_m$, then the chromosome X_i is selected to be mutated. The idea of the mutation operation is also originated from Lemma 0.1. Firstly, randomly select an integer k from interval $[1, |E(G) \setminus E(X_i)|]$. Then randomly select k different arcs $e_1, e_2, \cdots, e_k$ from set $E(G)\setminus E(X_i)$. For each arc e_j, $j = 1, 2, \cdots, k$, it follows from Lemma 0.1 that there is a unique cycle C_{e_j} in $X_i + \{e_j\}$. The mutation is performed on C_{e_j} as follows: randomly select an arc $e'_j \in E(C_{e_j})$ such that $e'_j \neq e_j$, set $X_i \leftarrow X_i - \{e'_j\} + \{e_j\}$. The mutation operation is summarized as follows.

Mutation algorithm:

Step 1. For $i = 1$ to *pop_size*, repeat Step 2 to Step 3.

Step 2. Randomly generate a random number r from interval $(0, 1]$. If $r \leq P_m$, then go to Step 3. Otherwise, go to Step 1.

Step 3. Randomly select an integer k from interval $[1, |E(G) \setminus E(X_i)|]$ and randomly select k different arcs $e_1, e_2, \cdots, e_k$ from set $E(G)\setminus E(X_i)$. For $j = 1$ to k, repeat Step 4.

Step 4. Find the unique cycle C_{e_j} in $X_i + \{e_j\}$. Randomly select an arc $e'_j \in E(C_{e_j})$ such that $e'_j \neq e_j$, set $X_i \leftarrow X_i - \{e'_j\} + \{e_j\}$.

Clearly, such a mutation can also ensure that the new chromosome is also a spanning trees.

In order to accelerate the convergence speed of the designed genetic algorithm, some heuristic methods can be used in the mutation operator. The base idea is to give each arc in the set $S = E(G) \setminus E(X_i)$ a selecting probability according to the weight of the arcs when we perform mutation operation on X_i. Assume that $S = E(G) \setminus E(X_i) = \{e'_i | i = 1, 2, \cdots, |S|\}$. The selecting probability of the arc $e'_i \in S$ is defined as $f(e'_i) = (1/w'_i) \Big/ \sum_{e'_j \in S} (1/w'_j)$. Let $P_i = \sum_{k=1}^{i} f(e'_i)$, $i = 1, 2, \cdots, |S|$, and $P_0 = 0$. Then the arcs in $S = E(G) \setminus E(X_i)$ is selected by spanning roulette wheel when we select the arc e_j in the above process operation. The detail operation is similar as that in the selection operation and we omitted here.

III. 5 Evaluation

The evaluation process is to calculate the fitness of chromosomes so as to evaluate the chromosomes. In the designed algorithm, the fitness f_i of the ith chromosome X_i is defined as the reciprocal of the weight $W(X_i)$ (see Definition 0.1), $i = 1, 2, \cdots, pop\_size$. The evaluating function is defined as

$$Eval(X_i) = f_i \Big/ \sum_{k=1}^{pop\_size} f_k \ , i = 1, 2, \cdots, pop\_size.$$

III. 6 Selection Process

The selection process is to select the offsprings of the chromosomes in the genetic algorithm. Let $P_i = \sum_{k=1}^{i} Eval(X_i)$, $i = 1, 2, \cdots, pop\_size$, and $P_0 = 0$, then we employ the spanning roulette wheel to select the chromosomes: randomly generate a number $p \in (0, 1]$; if $p \in (P_{i-1}, P_i]$, then the chromosome X_i is selected. The following is the algorithm.

Selection algorithm:

Step 1. *Let $k = 1$, repeat Step 2 until $k = pop\_size$.*

Step 2. *Randomly generate a number $p \in (0, 1)$. If $p \in [P_{i-1}, P_i)$, then chromosome X_i is selected and let $k = k + 1$.*

III. 7 Genetic Algorithm

Step 1. *Randomly initialize pop_size chromosomes.*

Step 2. *Update the chromosomes by crossover process and mutation process.*

Step 3. *Calculate the objective values of the chromosomes.*

Step 4. *Calculate the fitness of each chromosome according to the objective values.*

Step 5. *Select the chromosomes by spanning the roulette wheel.*

Step 6. *Repeat Step 2 to Step 5 for a given number times.*

Step 7. *Report the best chromosome as the optimal solution.*

IV. Conclusions

In this paper, an oriented spanning tree-based heuristic genetic algorithm is designed for solving the quadratic shortest path problem. Finally, a numerical example is given.

Acknowledgements

This research is supported by National Natural Science Foundation of China (No.60174049) and the Qinglan talent Funds of Lanzhou Jiaotong University.

References

[1] E. W. Dijkstra. A note on two problems in connection with graphs. Numerical Mathematics, 1(1959), 269-271.

[2] V.Balachandhrm, C.P.Randan. All-pairs-shrotest-length on strongly chordal graphs. Discrete Applied Mathematics, 69(1996), 169-182.

[3] P.Pattanamekar, D.Park, L.R.Rilett etc.. Dynamic and stochastic shortest path in transportation networks with two components of travel time uncertainty. Transportation Research Part C, 11(2003), 331-354.

[4] A. Azaron, F. Kianfar. Dynamic shortest path in stochastic dynamic networks: Ship routing problem. European Journal of Operational Research, 144(2003), 138-156.

[5] D. Frigioni, A. Marchetti-Spaccamela. Fully dynamic shortest paths in digraphs with arbitrary arc weights. Journal of Algorithms, 49(2003), 86-113.

[6] C. Davies, P. Lingras. Genetic algorithms for routing shortest paths in dynamic and stochastic networks, European Journal of Operational Research, 144(2003), 27-38.

[7] C. Bousono-Calzon, A.R.Figuiras-Vidal. A bank Hopfield neural networks for the shortest path problrm, Signal Processing, 61(1997), 157-170.

[8] E. D. Miller-Hooks, H.S. Mahmassani. Least possible time paths in stochastic time-varying networks. Computer Operational Research, 25(1998), 1107-1125.

[9] Y. L. Chen, K. Tang. Minimum time paths in a networs with mixed time constraints. Computer Operational Research, 25(1998), 793-805.

[10] J. Granat, F. Guerriero. The interactive analysis of the multi-criteria shortest path problem by the reference point method. European Journal of Operational Research, 151(2003), 103-118.

[11] P. Modesti, A. Sciomachen. A utility measure for finding multi-objective shortest paths in urban multimodal transportation networks. European Journal of Operational Research, 111(1998),495-508.

[12] A.L. Roginsky, K.J. Christensen & V. Srinivasan. New methods for shortest path selection for multimedia traffic with two delay constraints. Computer Communications, 22(1999), 1531-1539.

[13] S. Banerjee, R. K. Ghosh & A. P. K. Reddr. Parallel algorithm for shortest pairs of edge-disjoint paths. Journal of parallel and distributed computing, 33(1996), 165-171.

[14] C. Barbieri, J. Swait. New dominance criteria for the generalized permanent labelling algorithm for the shortest path problem with time windows on dense graphs. International Transaction in Operational Research, 7(2000), 139-157.

[15] P. Avella, M. Boccia, A. Sforza. A penalty function heuristic for the resource constrained shortest path problem. European Journal of Operational Research, 142(2002), 221-230.

[16] I. Murthy, S. Sarkar. Exact algorithm for the stochastic shortest path problem with a decreasing deadline utility function. European Journal of Operational Research, 103(1997), 209-229.

[17] K. A. Rink, E. Y. Rondin, V. Sundarpandian. A simplification of the double-sweep algorithm to solve the k-shortest path problem. Applied Mathematics Letters, 13(2000),77-85.

[18] Y. Han. Improved algorithm for all pairs shortest paths. Information Processing Letters, 91(2004), 245-250.

[19] Y. Chen, H. Yang. Finding the first k shortest paths in a time-window network. Computers & Operations Research, 31(2004), 499-513.

[20] S. Pettie. A new approach to all-pairs shortest paths on real-weighted graphs. Theoretical Computer Science, 312(2004), 47-74.

[21] P. Bose, A. Maheshwari and,G. Narasimhan, etc.. Approximating geometric bottleneck shortest paths. Computational Geometry, 29(2004), 233-249.

[22] S.Okada. Fuzzy shortest path problems incorporating interactivity among paths. *Fuzzy Sets and Systems*, 142(2004), 335–357.

[23] M. Gen, R. Cheng . Genetic Algorithms & Engineering Optimization, Wiley-Interscience Publication, New York, 2000.

[24] I. Godor, J. Harmatos and A. Juttner. Inverse shortest path algorithms in protected UMTS access networks, Computer Communications, 28(2005), 765-772.

[25] C. F. Daganzo. Reversibility of the time-dependent shortest path problem. Transportation Research Part B, 36(2002), 665-668.

[26] M. Garey, D. Johnson. Computers and Intractability: A guide to the NP-completeness. W.H. Freeman, New York, 1979.

[27] J.A.Bondy, U.S.R.Murty. *Graph Theory with Application*, The Macmillan Press Ltd.,1976.

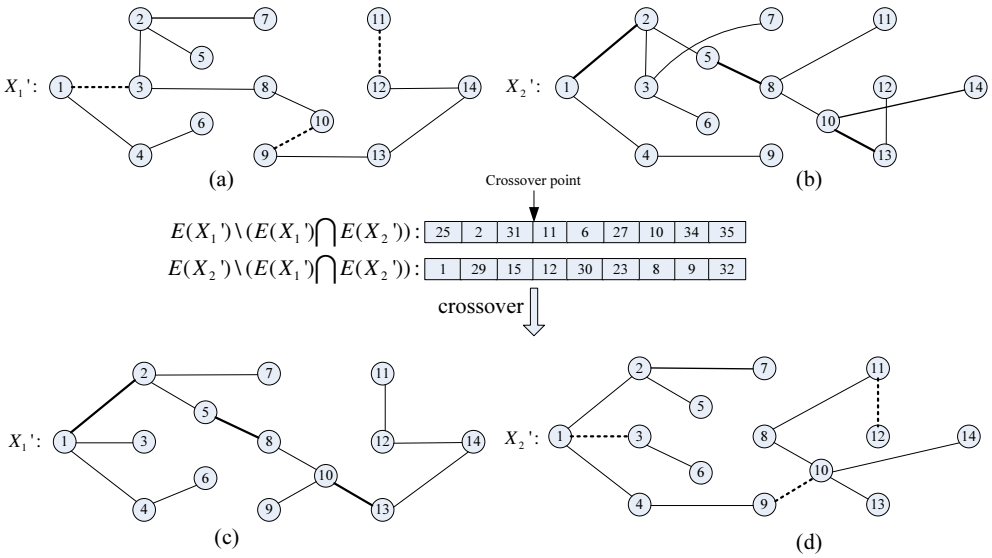

Figure 2: The crossover operation of X_1 and X_2.