

Article

Optimal Route Searching with Multiple Dynamical Constraints—A Geometric Algebra Approach

Dongshuang Li ¹ , Zhaoyuan Yu ^{1,2,3,*} , Wen Luo ^{1,2}, Yong Hu ^{1,4}, Xiaoyu Che ¹
and Linwang Yuan ^{1,2,3}

¹ Key Laboratory of Virtual Geographic Environment, Ministry of Education, Nanjing Normal University, Nanjing 210046, China; lds19921120@163.com or 09415@njnu.edu.cn (D.L.); luow1987@163.com (W.L.); huyong@njnu.edu.cn (Y.H.); chexy_mail@163.com (X.C.); yuanlinwang@njnu.edu.cn (L.Y.)

² State Key Laboratory Cultivation Base of Geographical Environment Evolution, Nanjing 210023, China

³ Jiangsu Center for Collaborative Innovation in Geographical Information Resource Development and Application, Nanjing 210023, China

⁴ Institute of Computer Science and Technology, Nanjing Normal University, Nanjing 210023, China

* Correspondence: yuzhaoyuan@njnu.edu.cn; Tel.: +86-135-1254-2434

Received: 2 March 2018; Accepted: 28 April 2018; Published: 4 May 2018



Abstract: The process of searching for a dynamic constrained optimal path has received increasing attention in traffic planning, evacuation, and personalized or collaborative traffic service. As most existing multiple constrained optimal path (MCOP) methods cannot search for a path given various types of constraints that dynamically change during the search, few approaches for dynamic multiple constrained optimal path (DMCOP) with type II dynamics are available for practical use. In this study, we develop a method to solve the DMCOP problem with type II dynamics based on the unification of various types of constraints under a geometric algebra (GA) framework. In our method, the network topology and three different types of constraints are represented by using algebraic base coding. With a parameterized optimization of the MCOP algorithm based on a greedy search strategy under the generation-refinement paradigm, this algorithm is found to accurately support the discovery of optimal paths as the constraints of numerical values, nodes, and route structure types are dynamically added to the network. The algorithm was tested with simulated cases of optimal tourism route searches in China's road networks with various combinations of constraints. The case study indicates that our algorithm can not only solve the DMCOP with different types of constraints but also use constraints to speed up the route filtering.

Keywords: geometric algebra; multiconstrained optimal path; network analysis; network coding; path extension; route planning; geographical information system

1. Introduction

Optimal path analysis, as one of the most fundamental methods of network analysis, has been used in multiple application fields (e.g., transportation, tourism, and evacuation) [1–7]. Searching for the optimal path that satisfies various kinds of constraints—such as a numerical value or range requirement of path weights (e.g., the time cost), attribute of part of the network elements (e.g., an exchange station as a node that must be visited), and topological structure of the path (e.g., noncyclic or cyclic visiting routes)—forms the problem of the multiple constrained optimal path (MCOP) [8–12]. If the network states or constraints change during the optimal route searching process, the problem becomes a dynamic multiple constrained optimal path (DMCOP) problem. Due to the complexity of constraints, both the MCOP and DMCOP problems constitute NP-hard problems [13,14].

Two different types of dynamics can increase the complexity of the DMCOP problem. With type I dynamics, the network condition, such as the weight or topology, changes during optimal path searching; the constraints are not altered. For example, in adaptive dynamic traffic navigation, the optimal path should be dynamically generated with the traffic conditions, where the weights are dynamically changed according to the current traffic status. With type II dynamics, the network conditions are not changed; however, the constraints used to restrict the searching of the optimal path change with time. One example is event-forced constraints, which are not determined unless a certain kind of event (e.g., emergency or user interactions) occurs. In this way, the constraints should be changed according to the specific schedule of the event dynamically occurring. Since these different constraints cannot be known in advance, these constraints should be added dynamically during route planning [15]. For instance, in a typical scenario of a long-distance self-driving tour, the navigation may first start from route planning with a start point and an end point. During the travel, users may add location constraints [16], reliability constraints [17], turn constraints [18] for intercity traffic and cycle routes [19], or satisfaction constraints [20]. Certain emergency cases may also exist that require the planned route to be dynamically adjusted in a short amount of time according to a series of constraints.

Previous initial studies have addressed the solution of DMCOP with type I dynamics (i.e., network status changes) [21]. Schott and Staples used geometric algebra (GA) to solve dynamical multiconstrained routing with numerical constraints [22]. Yuan et al. proposed a path matrix-based approach for route extension, which can support the optimal route generation for solving the MCOP problem [23]. Yu et al. further developed GA-based DMCOP methods for evacuation [24,25], which can support the DMCOP with dynamic updating of network weights and topologies. Nevertheless, for the DMCOP with type II dynamics, to our knowledge, no effective solution exists that can support different constraints that change during optimal path searching under a unified framework. One possible strategy is to split the problem into several subproblems. However, as different constraints are dynamically applied to the network, multiple MCOP algorithms may be combined to search for the optimal path, and this process results in high complexity or even conflicts that make the route combination fail.

One key issue that increases the complexity of solving DMCOP with type II dynamics is that a variety of constraints exists, including numerical constraints (e.g., the weight value or range requirement of path weights), network elements constraints (e.g., must-visit nodes), and the topological structure of the path (e.g., noncyclic or cyclic visiting routes). All of the various types of constraints cannot be uniformly represented and applied to the optimal route searching. For example, in most commonly used network analyses, the topology and the weights of the network are represented with certain types of data structures (e.g., table-based or tree-based data structures), but constraints in the network analysis are represented mainly as attributes or evaluation rules. As neither the network elements nor the constraints are uniformly represented, only rarely can a universal MCOP algorithm support multiple different types of constraints [26]. Therefore, how to use the normal mathematical language to uniformly describe the network elements and various types of constraints and integrate these constraints into the calculation process of optimal route filtering is the key challenge in solving the DMCOP problem with type II dynamics [27].

GA, founded on dimensional computation, provides an ideal tool for the unified network representation and analysis algorithm construction [23,28–30]. In our previous work, we introduced GA to the problem of multisource multisink optimal evacuation routing that partially provides the possibility to design the DMCOP algorithm to support different constraints. However, this method is performed from the perspective of data structure and lacks the integration of type II constraints to form a normative approach through a formal mathematical expression, and this method is also more focused on the weight and topological change of networks rather than the constraint change. If the constraints are complex and various, the efficiency of the method will be largely affected.

In this study, we discuss the development of a unified GA framework to address multiple types of constraints and propose a universal algorithm to solve the DMCOP problem with type II dynamics.

We first review the definition of the DMCOP problem with type II dynamics and the GA representation of various types of constraints, and then we create the GA coding of the network elements, including nodes, paths, routes, topologies, weights, and constraints. Network operations, such as the expansion of paths, link relationship computation, embedding weight, and filtering of routes with constraints, are developed to construct a DMCOP algorithm with type II dynamics. Then, a case study of tourism route planning is developed to test our algorithm.

2. Preliminary and Basic Ideas

2.1. Definition of the DMCOP with Type II Dynamics

Given the graph $G = (V, E)$, each path $P(i, j) \in G$ ($i, j \in V$) in the graph contains a main cost weight $\sum_{m=1}^M W_m(P(i, j))$, where m is each segmented element of the path, and a constraint set $W_c(P(i, j))$, which is abbreviated as W_c . The MCOP [31] can be defined as

$$\begin{cases} \text{obj. : } \sum_{m=1}^M W_m(p(s, t)) = \min\left\{\sum_{m=1}^M W_m(p(s, t))\right\} \\ \text{s.t. : } W_m(p(s, t)) \cap W_c \neq \emptyset, m = 1, 2, \dots, M \end{cases} \quad (1)$$

In Equation (1), the optimal route is defined as a path $P(s, t)$ with a start node s and an end node t that satisfy all of the constraints set with a minimum main cost weight $\sum_{m=1}^M W_m(P(s, t))$. Since various types of constraints exist, the constraints set W_c can be further split into multiple constraint sets, for example, $W_c = \{W_{nc}, W_{wc}, W_{tc}\}$, where W_{nc} , W_{wc} , and W_{tc} are nodes, weights, and topological constraints, respectively.

Since the constraints change according to the various schedules as the events dynamically occur in the DMCOP with type II dynamics, we can assume that the whole network search process can be split into several time frames $T_c = \{t_1, t_2, \dots, t_n\}$. For each t_i in T_c , a constraint change event occurs that necessitates a recalculation of the optimal path. Then, the optimal path search can be defined as follows:

$$\sum_{m=1}^{M_{i_{opt}}} W_m(P(s, t)_{opt}) = \min\left\{\sum_{m=1}^{M_{t_1}} W_m(P(s, r_1)) + \sum_{m=1}^{M_{t_2}} W_m(P(r_1, r_2)) + \dots + \sum_{m=1}^{M_{t_n}} W_m(P(r_n, t))\right\}, \quad (2)$$

Here, r_i ($i = 1, 2, \dots, n$) denotes the passing node in the route search, and $m = 1, 2, \dots, M_{t_i}$ ($i = 1, 2, \dots, n$) represents the segmented element of the path in time frame t_i . Although in DMCOP with type II dynamics, we do not initially know how the time frame should be split, for any selected route from the start point to the end point, there is a determined split of time frames. Assuming that various types of constraints should be processed during time frame sets $T_c = \{t_1, t_2, \dots, t_n\}$ and are expressed as $W_{t_{c_{T_c}}} = \{W_{t_{c_{t_1}}}, W_{t_{c_{t_2}}}, \dots, W_{t_{c_{t_n}}}\}$, then the feasible path at the time frame T_c should satisfy the constraints $W_m(P(s, r_i)) \cap W_{t_{c_{t_i}}} \neq \emptyset$, ($i = 1, 2, \dots, n$) ($m = 1, 2, \dots, M_{t_i}$). In this way, the DMCOP with type II dynamics can be defined as follows:

$$\begin{cases} \text{obj. : } \sum_{m=1}^{M_{i_{opt}}} W_m(P(s, t)_{opt}) = \min\left\{\sum_{m=1}^{M_{t_1}} W_m(P(s, r_1)) + \sum_{m=1}^{M_{t_2}} W_m(P(r_1, r_2)) + \dots + \sum_{m=1}^{M_{t_n}} W_m(P(r_n, t))\right\} \\ \text{s.t. : } W_m(P(s, r_i)) \cap W_{t_{c_{t_i}}} \neq \emptyset, (i = 1, 2, \dots, n) (m = 1, 2, \dots, M_{t_i}) \end{cases} \quad (3)$$

2.2. Basic Ideas

According to the problem definition above, the solution to the DMCOP problem with type II dynamics can be split into the following three steps: (1) Because the constraints are varied and often linked with the network elements, a formal description of constraints and network elements integrated

into a unified framework should be constructed to reduce the transformation complexity of different constraints; (2) Due to the dynamically added constraints, the route generation strategy, which can dynamically integrate constraints during feasible path generation and optimal path searching, should be constructed [32]; (3) As the actual network problems are often large in scale, an efficient solution strategy should be carefully designed. Therefore, combing through the GA properties and analysis above, the specific ideas regarding the three steps are as follows.

First, for the GA-based unified expression of network and constraints, the network elements, such as the nodes, edges, and paths, can be uniformly generated and stored in a blade structure. This blade-based expression form can represent the network elements already explicitly containing the nodes and path information of the route so that the representation of constraints can be directly constructed by rules that evaluate whether the blades are fitted for certain conditions or not. Second, for the route generation with dynamically added constraints, in the manner of GA route searching, the routes can be algebraically extended step-by-step with GA-based matrix multiplication operators [23]. Thus, the generation-refinement paradigm can be adopted to apply the constraints to the procedure of network route extension to filter out the unfeasible paths [33]. Third, for the optimizing strategy of the solution of the DMCOP problem with type II dynamics, the divide-and-conquer greedy algorithm can perform the local solution and updating of the path to reduce the problem scale. In addition, using the algebraic expression of constraints and the relation operators provided by GA, a strategy that is similar to recursive or pruning can be applied to reduce the search numbers and improve the ability and efficiency of path filtering.

Based on the GA network representation, route extension, and route filtering, a framework of GA-based DMCOP with type II dynamics was developed (Figure 1). In this framework, the network elements in the original network and the adjacency matrix were first represented by the GA coding. Then, the well-defined oriented join product was applied to realize the route extensions. Taking advantage of the unified network representation, the numerical, node-inclusion, and route-structure-related constraints were formally defined. By defining the operators that could support various types of constraints, a unified representation and integration strategy for various constraints was constructed. We then defined a dynamic optimal searching strategy of the feasible paths, which searches for the optimal path iteratively. During the path searching, the constraints were dynamically inserted and used to filter the feasible path. By iteratively sorting out the accumulated cost of all the feasible paths that were evaluated during the route extension, the optimal path was dynamically generated.

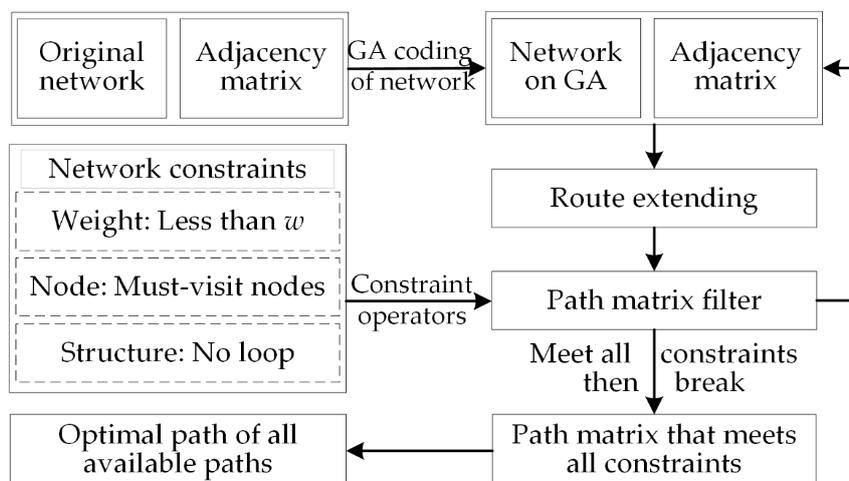


Figure 1. The framework of solving the DMCOP with type II dynamics.

3. Network Model Representation with GA

3.1. Basic Network Element Expression with GA

A unified representation of the network elements should integrate the representation of nodes, edges, paths, and weights of the network [23]. Thus, the following terms are first defined as follows.

Definition 1 (network node expression with GA). Consider a network node set $\{N_1, N_2, \dots, N_n\}$, which can be coded in the GA system as follows:

$$\text{Node}(\{N_1, N_2, \dots, N_n\}) = \{e_1, e_2, \dots, e_n\} = V, \tag{4}$$

where $\{e_1, e_2, \dots, e_n\}$ are the basis vectors in GA and correspond to the 1-blades that can be used as the fundamental construction elements of other network components (e.g., edges and paths).

Definition 2 (network edge and path expression with GA). The network edge E , which is normally interpreted as the connection between two adjacency nodes N_i and N_j , can be mapped to the GA space based on the outer product as follows:

$$E(N_i, N_j) = \text{Edge}(e_i, e_j) = e_i \wedge e_j, \tag{5}$$

Here, $e_i \wedge e_j$, the 2-blade, can be abbreviated as e_{ij} . By analogy, the path between the nodes N_i and N_j can be represented as the k-blades as $P(i, j) = e_i \wedge \dots \wedge e_k \wedge \dots \wedge e_j = e_{i\dots k\dots j}$, and the order of the vectors corresponds to the sequence of the passed nodes within the path. Then, the path set, which is used to describe the paths between the two given nodes, can be expressed as $S(i, j) = \{P(i, j)_1, P(i, j)_2, \dots, P(i, j)_n\}$.

Definition 3 (network weight expression with GA). The weights are scalars that indicate the cost of the edges or paths, which include the multiplicative and additive weights [26,34]. To integrate both the multiplicative weight w_{ij}^* and additive weights w_{ij}^+ to the edge e_{ij} , we can define the following expression:

$$W(i, j) = w_{ij}^* \exp(w_{ij}^+) e_{ij}, \tag{6}$$

Based on this expression, the weight calculation between the edges e_{ij} and e_{ik} can be expressed as $\text{Weight}(w_{ij}^* \exp(w_{ij}^+) e_{ij}, w_{jk}^* \exp(w_{jk}^+) e_{jk}) = (w_{ij}^* * w_{jk}^*) \exp(w_{ij}^+ + w_{jk}^+) P_{ik}$, where P_{ik} is the abbreviation of the path $P(i, k)$ that connects the edges e_{ij} and e_{jk} .

Basing on the above definitions, an example of the unified representation of the network with the GA is depicted in Figure 2.

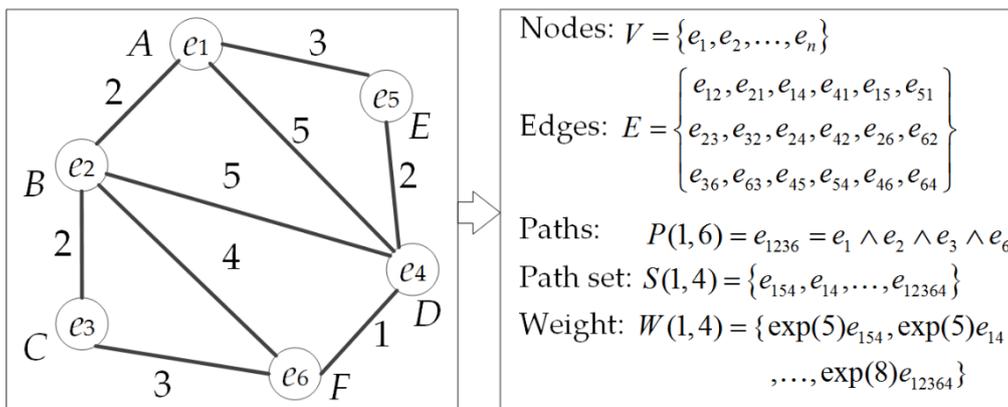


Figure 2. GA-based Network Representation.

3.2. Dynamic Route Generation with GA

In the DMCOOP problem with type II dynamics, because the constraints vary at different times, the feasible path must be dynamically generated and then filtered with the constraints. Thus, at first, the oriented join product \vee is used to support the route extension [23]. Given two paths $P(a, b)$ and $P(b, c)$, as the last node of the paths $P(a, b)$ has the same node b as the first node of path $P(b, c)$, their oriented join product is defined as follows:

$$P(a, b) \vee P(b, c) = P(a, b)' \wedge b \wedge P(b, c)', \quad (7)$$

where $P(a, b)'$ and $P(b, c)'$ are respectively the specific subspace of $P(a, b)$ and $P(b, c)$, from which their intersection b has been removed. For nonloop paths, $P(a, b) \vee P(c, d) = 0$ also means that the path has a loop by the oriented join product.

Because nodes, edges, and paths are all k -blades ($k = 1, 2, \dots, n$) that are caused by the geometric product of node vectors, the comprehensive expression and computation of nodes, edges, and paths in the framework can be realized. To make path generation possible, the following rules are constructed:

$$\left\{ \begin{array}{l} \text{path}(a, b) = e_{aij\dots kb} \\ \text{path}(a, b) \vee \text{path}(b, c) = \text{path}(a, c) \\ \text{path}(a, b) \vee \text{path}(d, f) = 0 \end{array} \right\} \begin{array}{l} \text{The expression rule} \\ \text{The adjacent rules} \end{array} \quad (8)$$

The expression rule provides a formal way to express the path for which a and b represent the subscripts of the start and end nodes, and i, j, \dots, k correspond to the subscripts of the visited nodes. The adjacent rules provide an operation and a determination for the path extension.

As a precondition of path analysis, the network graph directly affects the structure and complexity of the network analysis algorithms. Schott suggested using nilpotent adjacency matrices [35], which can generate an algebra system corresponding to the network topology. Slimane proposed operator calculus algorithms for MCOP path generation in which a path randomly passing n nodes can be represented as an n -order object (i.e., an n -blade) with n nodes connected in order [36]. Yuan established a universal network coding and a computing method based on the topological relationship between different nodes and developed the expression and algorithm structural framework based on the oriented join product [23]. Based on these studies, we can code the nodes using GA bases. The GA operation can then be applied to accomplish the generation, traversal, and selection of paths according to the definition of the topology and the space of the GA operation.

For a given network, the network-adjacency matrix M is initialized according to the GA-based coding as shown in Figure 2. The path of the network can be generated by applying the oriented join product to the matrix. In this way, the topological relationship and path weights can be obtained synchronously. Since the start nodes and end nodes are predefined in most cases and do not change during each iteration, it is preferred to construct the submatrix of the adjacent matrices by the start nodes and end nodes, which lessens the amount of computation for the algorithm. According to this idea, the start nodes matrix, end nodes matrix, and path matrix are constructed. Take the network in Figure 2 as an example. If we require all of the paths from nodes e_1, e_3 to nodes e_2, e_6 , the path matrix can be initialized as follows:

Figure 3 indicates that the path matrix selects the elements in the 1st and 3rd rows and in the 2nd and 6th columns. The path matrix can be represented as $\mathbf{M}_{F\{e_1, e_3\}T\{e_2, e_6\}}$, mainly because the 1st and 3rd rows represent all the edges starting with nodes e_1, e_3 and the 2nd and 6th columns represent all the edges ending with e_2, e_6 . Similarly, the start node and end node matrices can be expressed as $\mathbf{A}^1 = \mathbf{M}_{F\{e_1, e_3\}T\{\cdot\}}$ and $\mathbf{M}_{F\{\cdot\}T\{e_2, e_6\}}$. Because the start node matrix is the computing matrix of the next iteration of the loop, the computing matrix can be assigned with $A = \mathbf{M}_{F\{e_1, e_3\}T\{\cdot\}}$.

$$\mathbf{M} = \begin{pmatrix} 0 & 2e_{12} & 0 & 5e_{14} & 3e_{15} & 0 \\ 2e_{21} & 0 & 2e_{23} & 5e_{24} & 0 & 4e_{26} \\ 0 & 2e_{32} & 0 & 0 & 0 & 3e_{36} \\ 5e_{41} & 5e_{42} & 0 & 0 & 2e_{45} & 1e_{46} \\ 3e_{51} & 0 & 0 & 2e_{54} & 0 & 0 \\ 0 & 4e_{62} & 3e_{63} & 1e_{64} & 0 & 0 \end{pmatrix}$$

Start nodes matrix: \mathbf{A}^1
 $\mathbf{A}^1 = \mathbf{M}_{F\{e_1, e_3\} T\{\}}^1 = \begin{pmatrix} 0 & 2e_{12} & 0 & 5e_{14} & 3e_{15} & 0 \\ 0 & 2e_{32} & 0 & 0 & 0 & 3e_{36} \end{pmatrix}$
 End nodes matrix:
 $\mathbf{M}_{F\{\} T\{e_2, e_6\}}^1 = \begin{pmatrix} 2e_{12} & 0 & 2e_{32} & 5e_{42} & 0 & 4e_{62} \\ 0 & 4e_{26} & 3e_{36} & 1e_{46} & 0 & 0 \end{pmatrix}^T$
 Path matrix:
 $\mathbf{M}_{F\{e_1, e_3\} T\{e_2, e_6\}}^1 = \begin{pmatrix} 2e_{12} & 0 \\ 2e_{32} & 3e_{36} \end{pmatrix}$

Figure 3. Path matrix construction from the adjacency matrix.

According to Equation (7), the path is extended by the oriented join product. Similarly, the computing matrix can be extended by the oriented join product. Using the network and adjacency matrix in Figure 3, the computing matrix extension can be realized with the following equation:

$$\mathbf{A}^2 = \mathbf{A}^1 \vee \mathbf{M} = \begin{pmatrix} 0 & 25e_{142} & 4e_{123} & \{10e_{124}, 6e_{154}\} & 0 & \{8e_{126}, 5e_{146}\} \\ 4e_{321} & 12e_{362} & 0 & \{10e_{324}, 3e_{364}\} & 0 & 8e_{326} \end{pmatrix}, \quad (9)$$

4. GA Expression and Processing of Multiple Constraints

4.1. GA Expression of Multiple Constraints

For the DMCOP with type II dynamics algorithm, the key is the path selection for different constraints, which are recognized as certain conditions that paths should satisfy in the DMCOP with type II dynamics. Although it is possible to carry out direct searching and filtering of the route to fit the constraints using brute-force route extraction [23], the power of the algebraic representation cannot be fully used, and the solution to the problem is not elegant. Therefore, in this section, we formally define the algebraic representation of constraints and aim to integrate them into optimal route searching. Considering that network elements are uniformly expressed by a blade structure that explicitly contains the nodes and path information of the route, the expression of constraints can be directly constructed by rules that evaluate whether the blades are fitted for certain conditions or not. For example, the must-visit node constraints can be formulated as a certain basis vector that must exist in the blade, and the path cost constraints can be defined by filtering the sum of the scalar part of the blade. Therefore, according to the node, weight, and topological constraints, three corresponding operators, i.e., the node-inclusion constraint operator $C_{Node}()$, number constraint operator $C_{Num}()$, and linear structure constraint operator $C_{LStr}()$ are defined as follows.

1. Node-inclusion constraints:

Node-inclusion constraints mean that for a special-node issue, the optimal route should contain a special node, i.e., the blade set of the path should contain the basis vector that indicates the special node. Therefore, the special-node issue can be realized by searching for the special basis vector of the blade set. According to Equations (2) and (3), we use the meet operator “ \cap ” to detect the must-visit nodes. If any path in the route does not satisfy the node-inclusion constraints, the route is filtered out from the feasible path sets. Supposing that arbitrary node e_v is in the must-visit node set V' , we have

$$C_{Node}(p(i, j), e_v) = \begin{cases} 0, & \text{if } p(i, j) \cap e_v = 0 \\ 1, & \text{otherwise} \end{cases}, \quad (10)$$

For each step of route searching, we check the constraint to remove the invalid path that is unable to create an appropriate path in the next steps. Since the route is dynamically generated, the paths

that do not satisfy the constraints are not processed further in the next step. In this way, the additional constraints do not rapidly increase the computational complexity. In addition, the large number of constraints greatly reduces the scale of the feasible path set during traversal, making it possible to support a large degree of constraint filtering.

2. Numerical constraints:

The numerical constraints can be transformed into the threshold constraints represented as g_{\max} . Since the arbitrary path $p(i, j)$ is expressed using the blade structure, the norm operator $\|p(i, j)\|$, which can extract the scalar part of the blade, can be used to extract the weight part of the path. Thus, $C_{Num}()$ can be defined as follows:

$$C_{Num}(p(i, j), g_{\max}) = \begin{cases} 0, & \text{if } \|p(i, j)\| > g_{\max} \\ 1, & \text{otherwise} \end{cases}, \quad (11)$$

The numerical constraints can be integrated during the optimal route searching by directly evaluating the value of the weight. Comprehensively considering the influence of the weights, we establish the function f that can embed the numerical constraints into the search path; thus, the updated k -order path $p(i, j)'$ under the GA framework is

$$p(i, j)' = f(p(i, j)) = C_{Num}(p(i, j))p(i, j), \quad (12)$$

where $C_{Num}(p(i, j))$ means that the numerical constraints can be embedded in the route extension procedure. Since the path in the GA framework is dynamically generated using matrix multiplication, if any path in a route has not satisfied the numerical constraints, then the route is filtered out from the feasible path set. This process is also helpful for optimal path searching when the number of constraints is large.

3. Route-structure-related constraints:

The structure constraints include the open route and the circular route (loop route) [37]. A loop is a kind of special path structure where the start node and end node are the same. If a certain path has a circuit, the path must have at least one repeating node [38]. The outer product can be available to decide the linear dependence, i.e., the result is 0 when the same elements between the edges are involved in the outer product performance. Furthermore, in an adjacent matrix, all the diagonal elements represent the loop path based on the current location [29]. In this way, the loop and nonloop paths not only are distinguished but also can be processed independently. Since the loop route we consider here is a constraint, if the constraint exists, only algebraic elements in the diagonal line are required to be considered. Otherwise, if no circular constraints exist, then the circular paths can be skipped such that the structure-related constraining operator $C_{LStr}()$ for the given path $p(i, j)$ can be defined as follows:

$$C_{LStr}(p(i, j)) = \begin{cases} 1, & \text{if } i \neq j \\ 0, & \text{otherwise} \end{cases} \quad (13)$$

Thus, the $C_{LStr}()$ operator eliminates all of the paths with the same subscripts, which means that the start point and the end point are the same. Therefore, by using the linear structure constraint operator $C_{LStr}()$, the estimated routes are all open paths. In contrast, because the k -order circular route directly corresponds to the diagonal elements of the adjacent matrix, the circular constraints can be achieved by retrieving the diagonal elements.

4.2. Dynamic Filtering of the Path by Constraints

During the route-searching process, the path extension is guaranteed to satisfy the three types of constraints introduced in the previous section. If the path satisfies the constraints, this path is one of

the candidates for the optimal path. To screen out suboptimal paths, an updated method of the path matrix is defined, which is also helpful for dynamic updating and tracing of the path with constraints. To make the path filtering clear and efficient, we can define the following three route filter operators:

Definition 4. *NodeFilterOp* is defined with the meet operator, which is used to calculate whether a certain node e_i is included in the generated path:

$$\text{NodeFilterOp}(p(s,t), e_i) = p(s,t) \cap e_i = \begin{cases} 0, & \text{if } e_i \text{ is not included in } p(s,t) \\ 1, & \text{if } e_i \text{ is included in } p(s,t) \end{cases}, \quad (14)$$

The route is generated by the computing matrix \mathbf{A} , which contains several different routes during the route generation stage. Directly applying the *NodeFilterOp* operator to each route in the matrix \mathbf{A} may not be computationally efficient. Thus, we can further optimize the *NodeFilterOp* operator by the set relations. Initially, the must-visit nodes passed in \mathbf{A}^n can be obtained by \cap relative to those in \mathbf{A}^{n-1} at the last step. The corresponding elements in \mathbf{A}^{n-1} must be updated if the path in \mathbf{A}^n contains new must-visit nodes. The relationship of must-visit nodes between these two path matrices can be determined by the standard join operator “ \cup ” because this operator is a set algebra relation. Supposing that the must-visit node sets are V^n and V^{n-1} , the update principles of the computing matrix are as follows:

$$\begin{cases} \mathbf{A}^n = \mathbf{A}^n \oplus \mathbf{A}^{n-1} & \text{grade}(V^n \cup V^{n-1}) > \max(\text{grade}(V^n), \text{grade}(V^{n-1})) \\ \mathbf{A}^n = \mathbf{A}^{n-1} & \text{grade}(V^n \cup V^{n-1}) = \text{grade}(V^{n-1}) \\ \mathbf{A}^n = \mathbf{A}^n & \text{grade}(V^n \cup V^{n-1}) = \text{grade}(V^n) \end{cases}, \quad (15)$$

In the first pattern, the grade of $V^n \cup V^{n-1}$ is greater than the maximum grade of V^n and V^{n-1} (Figure 4a,b). The solved path should be merged with the original path matrix; in the second case, when the grade of $V^n \cup V^{n-1}$ equals the grade of V^{n-1} , the path matrix can be generated according to the end nodes in the commuting matrix \mathbf{A}^{n-1} , and then the path in \mathbf{M}^{n-1} is the optimal one (Figure 4c). Finally, when the grade equals the grade of V^n , the newly solved path is optimal (Figure 4d).

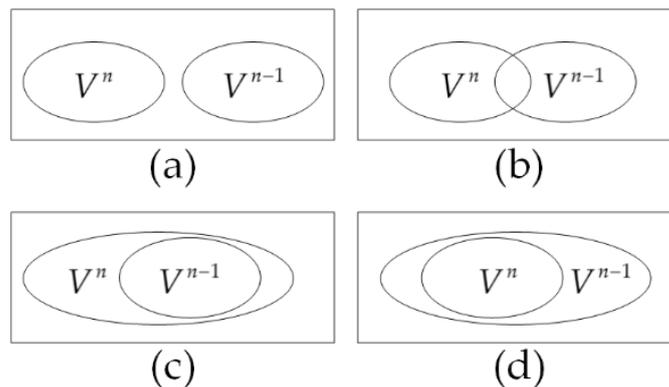


Figure 4. Inclusion relations of the must-visit nodes. (a) the grade of $V^n \cup V^{n-1}$ is greater than maximum grade of V^n and V^{n-1} ; (b) the grade of $V^n \cup V^{n-1}$ is greater than maximum grade of V^n and V^{n-1} ; (c) the grade of $V^n \cup V^{n-1}$ equals the grade of V^{n-1} ; (d) the grade of $V^n \cup V^{n-1}$ equals the grade of V^n .

Definition 5. *NumericalFilterOp* is the numerical constraints filter operator, which is used to calculate whether certain network elements' weights or costs fit a certain threshold. Thus, we define the *NumericalFilterOp*($p(s, t)$, *type*, *threshold*) as follows:

$$\begin{cases} \text{NumericalFilterOp}(p(s, t), \text{weight}, g_{\text{threshold}}) = \begin{cases} 0, & \text{if } \|p_{st}\| > g_{\text{threshold}} \\ 1, & \text{otherwise} \end{cases} \\ \text{NumericalFilterOp}(p(s, t), \text{cost}, g_{\text{threshold}}) = \begin{cases} 0, & \text{if } f(\|p_{st}\|) > g_{\text{threshold}} \\ 1, & \text{otherwise} \end{cases} \end{cases}, \quad (16)$$

According to Equation (8), we can further extend *NumericalFilterOp* to a multiplicative weight or cost. In *NumericalFilterOp*, the numerical constraints for weights can be directly filtered by the w_i^* and w_j^+ of a certain path. The filtering of such weights is straightforward and direct during path generation, i.e., in the update procedure of the computing matrix \mathbf{A}^n , the path element that does not fit the constraint can be replaced by 0. For the cost constraints, the cost function should be applied to each path in the generated path matrix \mathbf{M}^n , and the path that does not fit the cost constraints should be filtered.

Definition 6. *TopologicalFilterOp* is the topological constraint filter operator. *TopologicalFilterOp* can filter straight or circular paths. As in the definition of $C_{LStr}(p(s, t))$, we can use the elements located at the diagonal of the path matrix \mathbf{M}^n to filter the cyclers. In this way, we define *TopologicalFilterOp* as

$$\begin{cases} \text{TopologicalFilterOp}(p(s, t), \text{cycle}) = p(s, t) \in \mathbf{M}_{ij}^n, i = j \text{ and } i, j \text{ is connected with } n - \text{th orders} \\ \text{TopologicalFilterOp}(p(s, t), \text{nocycle}) = p(s, t) \in \mathbf{M}_{ij}^n, i \neq j \text{ or } i, j \text{ is not connected with } n - \text{th orders} \end{cases} \quad (17)$$

Here, \mathbf{M}_{ij}^n represents the element in the i -th row and j -th column of the path matrix \mathbf{M}^n . By using the update method, we can remove many unnecessary traversals by deleting paths that break rules, which can significantly decrease computational complexity. In addition, based on the simultaneous path extension from the beginning to the ending nodes, the efficiency of the algorithm can be better enhanced. Relative to the traditional step-by-step analysis algorithm based on the greedy method, the shortest path of GA adjacency is clearer and simpler.

4.3. Greedy Multiconstraint Route-Searching Algorithm

Based on the definitions above, we define a greedy multiconstraint route-searching algorithm on the foundation of the multiplication by the adjacent matrix and computation matrix and use the constraint filtering operators to filter the feasible path. Since the constraint integration is realized by evaluating simple conditions in searching, increasing the number of constraints does not necessarily increase the computational complexity. Therefore, the algorithm proposed here can efficiently support the optimum path analysis over a large range of constrained conditions. The complete algorithm is depicted in Figure 5.

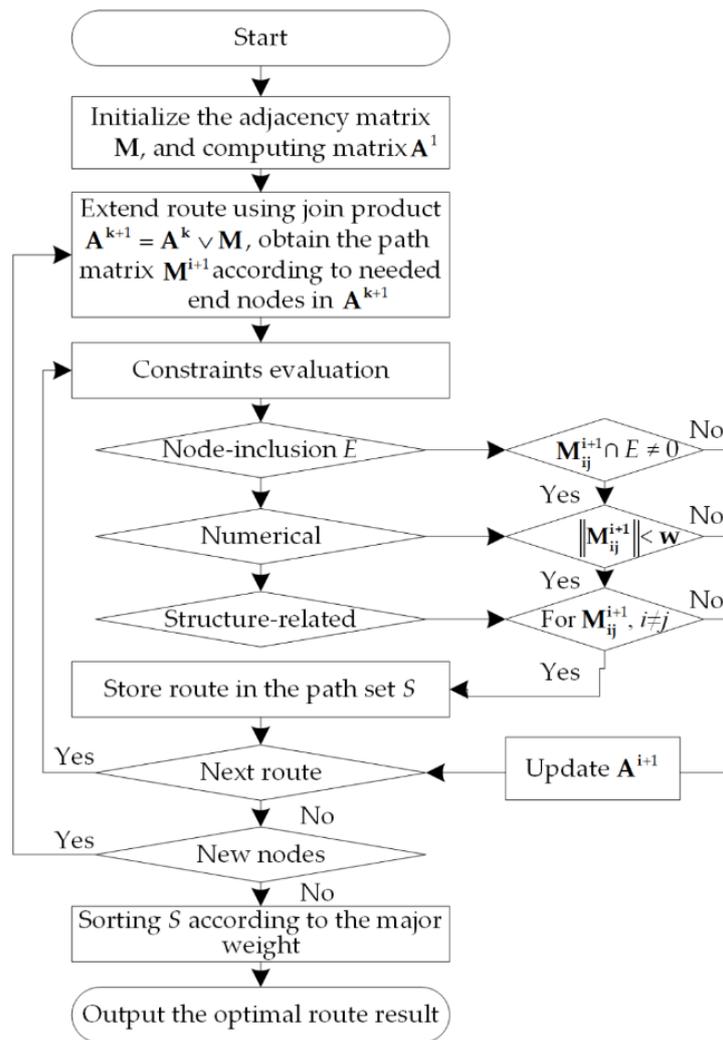


Figure 5. The flow of the multiconstrained optimal path algorithm based on GA.

The implementation process of the greedy multiconstraint route-searching algorithm is provided in detail as follows:

- (1) Construct the adjacent matrix \mathbf{M} and the computing matrix (start node matrix) \mathbf{A}^1 via the network data and the start nodes. We selected a subset of elements in the matrix \mathbf{A}^1 to form the simplified path matrix \mathbf{M}^1 , which was adopted to store the path that connects the end nodes.
- (2) For the given start node e_a and end node e_b , the simplified start node matrix \mathbf{A}^1 is $\mathbf{M}_{F\{e_a\}T\{\cdot\}}$. Extend the first route by using the formula $\mathbf{A}^2 = \mathbf{A}^1 \vee \mathbf{M}$, which extends the route from time frame t_1 to t_2 . The final computing matrix \mathbf{A}^2 can be updated according to the above Formula (15), as the computing matrix \mathbf{A}^2 has the same start node as $\mathbf{M}_{F\{e_a\}T\{\cdot\}}$, and then the path matrix \mathbf{M}^2 can be generated according to the end node matrix $\mathbf{M}_{F\{\cdot\}T\{e_b\}}$ and defined as $\mathbf{M}_{F\{e_a\}T\{e_b\}}^2$.
- (3) Apply constraints with the constraining operators and filter the path matrix.
- (4) Extend the computing matrix by using the formula $\mathbf{A}^{k+1} = \mathbf{A}^k \vee \mathbf{M}$ and Formula (15).
- (5) Repeat the third and fourth steps until all paths that meet the constraints are found.

5. Case Study: Application in Dynamic Tourism Planning

We evaluated a real-world case study of the planning of a tourism route in Jiangsu Province with the road network data. In this case study, optimal tourism planning routes that should

dynamically meet several different inserted constraints were studied based on Jiangsu road network data downloaded from the OpenStreetMap website (<http://www.openstreetmap.org>). After the topological correction, the final experiment data contained 9783 nodes and 23,338 road segments. To test the various constraints, we added several different attributes to the road segments of the data, including the lengths, velocity, and pass time of the road segments. The data were first imported into the CAUSTA system [30] and then represented by GA.

To illustrate the dynamic scenarios, a simulated tourism event was raised, which could be separated by 4 steps as shown in Table 1. First, the initial goal of the tourist was to drive from Site 1 to Site 6; second, after some deep investigation, four interest points, i.e., Site 2, Site 3, Site 4, and Site 5, were added; third, after arriving at Site 2, because of the time limitation, Site 3 was omitted from the interest list; fourth, upon arriving at Site 4, because of a budget shortage, the remainder of the tour had to circumvent the highway to avoid the road toll.

Table 1. Table of constraints in the case study.

Steps	Demands	Constraints
Step 1	Site 1 to Site 6	The shortest path from Site 1 to Site 6
Step 2	Site 1 to Site 6; pass Site 2, Site 3, Site 4, and Site 5	The shortest path from Site 1 to Site 6 with the node constraint of Site 2, Site 3, Site 4, and Site 5
Step 3	Site 2 to Site 6; pass Site 4 and Site 5	The shortest path from Site 2 to Site 6 with the node constraint of Site 4 and Site 5
Step 4	Site 4 to Site 6; pass Site 5; cannot use the highway road	The shortest path from Site 4 to Site 6 with the node constraint of Site 5, and with the attribute-related constraint that cannot use highway road

The 4-step dynamic constraints listed in Table 1 were formulated in the CAUSTA system and were interactively applied to filter the result path. For the first step, the filter was set to NULL so that the shortest path without any other constraints was evaluated (Figure 6a); during the second step, the node constraint was dynamically added to obtain a more winding path and obtain all the interest points (Figure 6b); after arriving at the third step, one of the constraint nodes was omitted to update the road cost in less time (Figure 6c); finally, the attribute-related constraint of the highway road was incorporated to obtain a more affordable path (Figure 6d). As shown in the case study, according to the dynamic insertion of the constraints, the feasible path sets are dynamically reduced, and the algorithm can filter out the final optimal path.

The experiment demonstrated that our method obtains the optimal path with numeric, node, and structure constraints. However, the applicability of the method in real tourism route planning should also integrate more realistic factors and provide several alternative paths to fit customer requirements. The support for various kinds of constraints in the optimal routing can certainly be beneficial for real-world tourism routing.

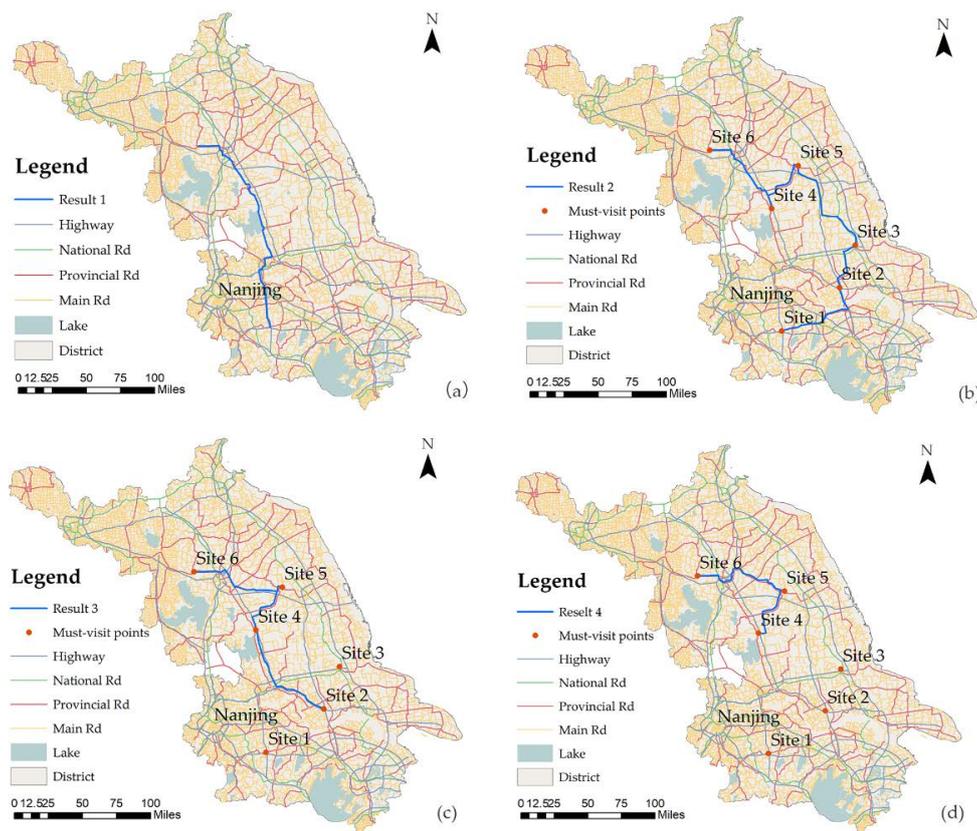


Figure 6. Demonstration of optimal path searching with dynamic addition of different constraints. (a) the shortest path without any constraints; (b) the winding path with the dynamically added node constraint; (c) the updated path when one of the constraint nodes was omitted; (d) the affordable path when the attribute-related constraint of the highway road was incorporated.

6. Comparisons with other Methods

6.1. Support for the Various Constraints

For the DMCOP methods with type II dynamics, the abundance of types of constraints can increase compatibility with different dynamic application scenarios. Table 2 lists the most commonly used constraints, including node-inclusion constraints (Constraints ID 1), additive weights (ID 2, ID 3), multiplicative weights (ID 4), and route type constraints (ID 5, ID 6). Other types of attributes of the path, including road type, road grade, and road status (ID 7), were considered. These constraints were then formulated as GA expressions and processed interactively and dynamically in the Clifford Algebra-based Unified Spatial–Temporal Analysis (CAUSTA) system with the DMCOP analysis modules using Visual C++, which is sourced from Microsoft Corporation Redmond, Washington, USA [30]. All experiments were tested on a Windows 7 system using Visual Studio 2010, which is sourced from Microsoft Corporation Redmond, Washington, USA. The hardware platform was a Lenovo T440s notebook with an Intel Core i5-4200U CPU (1.60 GHz, 4 GB RAM).

Since there are no commonly used DMCOP methods with type II dynamics, the four existing MCOP algorithms, namely, the Lagrange relaxation traverse algorithm (LRT), Yen’s *K*-Path algorithm, Staples’s *K*-cycle algorithm [29], and Lozano’s exact method [39], are used as references to test the accuracy and efficiency of our approach. As none of the four mentioned algorithms can support all of the constraints we provide in Table 2, we first compared the applicability of different algorithms with various types of constraints (Table 3). Clearly, our method is the only one that can accommodate a variety of constraints in a single algorithm.

Table 2. Table of constraints in the case studies.

Constraints ID	Constraints Type	Examples
1	Node-inclusion constraints	Node ID 1106 must be included in the final route
2	Additive numerical constraints	The length of the path should be less than 200 km
3	Additive numerical constraints	The velocity of the path should be greater than 60 km/h
4	Multiplicative numerical constraints	The tourism cost (distance·time) should be less than 300 km·hour
5	Route structure-related constraints	The tourism route should be circular
6	Route structure-related constraints	The tourism route should not be circular
7	Route attribute-related constraints	The route must use the highway

Table 3. Constraints supporting comparison between different algorithms.

Constraints	Algorithms				
	LRT	Yen's K-Path Algorithm	Modified Staples's K-Cycle Algorithm	Lozano's Exact Method	Our Algorithm
Node-inclusion	+	-	+	-	++
Additive numerical	++	++	++	++	++
Multiplicative numerical	-	-	++	++	++
Multiple weight	-	-	++	++	++
Route structure-related	-	-	+	-	++
Route attribute-related	+	-	-	-	+

++, the algorithm can support such constraints directly; +, the algorithm can support such constraints with modification or with additional filtering with the output results; -, the algorithm cannot support such constraints.

6.2. Efficiency Comparison

To test the performance of our algorithm, the various types of constraints in Table 2 were applied randomly to perform 100 experiments. As Staples's K-cycle algorithm is originally implemented in Mathematica but is unavailable in C++, this algorithm is not contrasted here. The average computation times were logged and are compared in Table 4. From Table 4, our algorithm was the only one that calculated the optimal path within an acceptable computation time. Although the existing mature algorithms were faster in some cases (e.g., with only the numerical additive constraints), this may be because these methods store the network topology and constraints in advance by the adjacency matrix or adjacency list. But these methods are mainly designed for the problems with specific constraints that are difficult to deal with and the DMCOP problems with type II dynamics. In our algorithm, the route structure constraints in particular greatly increased the computational cost because the filtering of the route structure in most cases is expected to eliminate all of the possible paths. Since the GA-based DMCOP algorithm with type II dynamics is only in the early stages of development, we expect it to be possible to fully optimize the algorithm to make it considerably more efficient, on par with most of the current MCOP algorithms.

Table 4. Average time costs of different algorithms with different constraints (unit: second/s) *.

Algorithm	LRT	Yen's K-Path Algorithm	Lozano's Exact Method	Our Algorithm
With Constraints ID 1	/	/	/	114.5
With Constraints ID 2	17.6	23.6	12.1	34.5
With Constraints ID 3	15.7	21.8	11.6	33.2
With Both Constraints ID 3 and 4	/	/	25.2	36.7
With Constraints ID 5	/	/	/	299.7
With Constraints ID 6	/	/	/	214.7
With Constraints ID 7	/	/	/	32.1
With Constraints ID 1–5 + 7	/	/	/	298.9
With Constraints ID 1–4 + 6, 7	/	/	/	278.7

* The average time cost of Staples' K-cycle Algorithm with constraints 2, 3, 3 and 4, 5, and 6 in Mathematica were 217.2, 280.8, 223.1, 256.3, and 345.6 s, respectively.

7. Discussion

7.1. Potential Application of this Template-Based Approach

The diversity and heterogeneity of the constraints pose major challenges to the construction of the GA-based network analysis algorithm. The algebraic formulation of the wider range of constraints may be the key issue for extending the GA-based DMCOP algorithm with type II dynamics. In this study, we mathematically represented three different kinds of constraints and integrated them into the DMCOP with type II dynamics. However, more complex constraints should also be considered in realistic traffic routing applications [40,41]. For example, in real tourism route planning, issues such as road form, max-flow, and objective constraint should be modeled and integrated into the solution to the DMCOP problem with type II dynamics. Other updates of the components, such as attributions, should also be considered. A large percentage of such constraints can be transformed into one of the other types of constraints or a combination thereof. Therefore, the optimal path can be solved directly.

For other kinds of constraints that cannot be converted into the three types examined here, the dynamic generation properties of solving the GA-based network path may be helpful in solving such problems according to the generation-refinement paradigm (e.g., multiconstraint transportation network design [42,43] and transit network analysis [12,44]). With the network represented in the GA framework, the route in the network can be directly generated and refined according to the properties of the algebra system. With the generation-refinement paradigm, certain kinds of constraints (e.g., routes with a certain number of nodes) can be directly solved without traversing all of the network routes.

In the GA-based DMCOP framework with type II dynamics, the path matrix method concentrated on GA offers a unified expression for both the connective relationship of the network graph and the node information. The method directly calculates the path based on the adjacency matrix. Since all of the path evaluations and queries are based on GA, which has unified mathematical and geometrical significance, the method may aid the construction of distinct network analysis algorithms and then unify the analysis algorithm structures of various networks. In addition, the weight expression and embedding methods can support path traversal, including weights. The separation of weight and topology enables the quick indexing of the path and supports the path searching process even if the weight and the constraints are changed dynamically. Moreover, the independence and uniformity of the oriented join product operation provide the network analysis based on GA with a computational infrastructure for parallel computing. All features above contribute to the large-scale network analysis in the multicondition, multiconstraint, and multiuser environment.

7.2. Directions for Future Improvements

In our GA-based DMCOP algorithm with type II dynamics, optimizing paths generated by the oriented join product of adjacency matrices was the key to enhancing the algorithm efficiency. Further optimization could still be achieved to improve the efficiency of the algorithms. The optimization approaches include the following: (1) constructing general data models and structures that are suitable for large-scale networks; for example, a sparse matrix, which can reduce the time and space occupancy for this kind of network, could be used to increase the storage and query efficiency [45]; (2) constructing quick parallel algorithms and applying them to the oriented join product operation of the adjacency matrix; for example, we can introduce matrix partitioning or premultiplied tables to accelerate the path-searching process; (3) optimizing the constrained integration and path selection principles when paths are generated; path querying using the oriented join product can sharply reduce the computing complexity and support the network analysis with more complex constraints; and (4) optimizing the method with decomposition and hierarchical representation of routes [46,47].

8. Conclusions

The DMCOP algorithm with type II dynamics is vital to traffic path planning analysis. It is challenging for the current algorithms to solve problems under a unified framework due to the diversity

and complexity of conditional constraints. In this study, we used rigorous mathematical definition, the multidimensional expression of GA, to construct a unified network expression of network elements and constraints and then achieved an algorithm under the generation-refinement paradigm to solve the DMCOP problem with type II dynamics. The algorithm can support numerical, nodal, and structural constraints that are dynamically inserted during the path searching. Relative to most existing MCOP and DMCOP algorithms, our method is more unified and has comparable efficiency, especially when addressing a large range of constrained conditions. Although our method results in mathematical construction complexity and possibly small affection in efficiency in engineering realization, based on this uniform representation, calculation, and algorithm construction framework, our method is flexible, complete, and elegant.

Author Contributions: All six authors have contributed to the work presented in this paper. D.L., Z.Y. and L.Y. formed the original idea. D.L., W.L., and Y.H. worked on the experiment and data analysis. D.L. and W.L. worked on the literature search and production of figures and tables. D.L. and X.C. wrote the paper. All authors worked collaboratively on writing the manuscript.

Funding: This research was funded by [National Natural Science Foundation of China] grant number [41571379, 41625004] and the PAPD program.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

	meaning	meaning of subscript and superscript
G	Network	None
V	Nodes	None
E	Edges	None
$P(s, t)$	Path from node s to node t	s, t : certain nodes
W_m	Main cost weight	m : the first letter of "main"
W_c	Constraints set	c : the first letter of "Constraint"
W_{nc}	Node constraints	n : the first letter of "Node"
W_{wc}	Weight constraints	w : the first letter of "Weight"
W_{tc}	Topological constraints	t : the first letter of "Topological"
T_c	Time frame	c : the first letter of "Constraint"
t_i	Inserting time frames of constraints	i : the numerical order, in italics
e_1, e_2, \dots, e_n	Basis vector	n : numerical order
$S(s, t)$	Path set from node s to node t	s, t : certain nodes
w_{ij}^*	Multiplicative weights	w : the first letter of "Weight", in italics, * indicates multiplicative weights, i, j : certain nodes
w_{ij}^+	Additive weight	w : the first letter of "Weight", in italics, + indicates additive weights i, j : certain nodes
$\mathbf{M}_{F\{s\}T\{t\}}$	Path matrix with the start node s and end node t	s, t : certain nodes
A^i	The computing matrix	i : power based on oriented join product
g_{\max}	The maximum acceptable threshold	max: the first three letters of "Maximum"
$g_{\text{threshold}}$	The weight threshold	none
\wedge	Outer product	none
\vee	Oriented join product	none
$C_{\text{Node}}()$	Node-inclusion constrain operator	Node: node, in italics
$C_{\text{Num}}()$	Number constrain operator	Num: number, in italics
$C_{\text{LStr}}()$	Linear structure constrain operator	LStr: linear structure, in italics

\cap	Meet operator	none
\cup	Join	none
$\ \ \ $	Norm operator	none
<i>grade()</i>	Dimension calculating operator	none
<i>NodeFilterOp()</i>	Node filter operator	none
<i>NumericalFilterOp()</i>	Numerical constraints filter operator	none
<i>TopologicalFilterOp()</i>	Topological constraints filter operator	none

References

- Zhang, X.; Chang, G.L. A dynamic evacuation model for pedestrian–vehicle mixed-flow networks. *Transp. Res. Part C Emerg. Technol.* **2014**, *40*, 75–92. [[CrossRef](#)]
- Inada, Y.; Izumi, S.; Koga, M.; Matsubara, S. Development of Planning Support System for Welfare Urban Design—Optimal Route Finding for Wheelchair Users ☆. *Procedia Environ. Sci.* **2014**, *22*, 61–69. [[CrossRef](#)]
- Papinski, D.; Scott, D.M. A GIS-based toolkit for route choice analysis. *J. Transp. Geogr.* **2011**, *19*, 434–442. [[CrossRef](#)]
- Liao, F.; Arentze, T.A.; Timmermans, H.J.P. Constructing personalized transportation networks in multi-state supernetworks: A heuristic approach. *Int. J. Geogr. Inf. Sci.* **2011**, *25*, 1885–1903. [[CrossRef](#)]
- Nepal, K.P.; Park, D. Solving the Median Shortest Path Problem in the Planning and Design of Urban Transportation Networks Using a Vector Labeling Algorithm. *Transp. Plan. Technol.* **2005**, *28*, 113–133. [[CrossRef](#)]
- Bast, H.; Funke, S.; Sanders, P.; Schultes, D. Fast Routing in Road Networks with Transit Nodes. *Science* **2007**, *316*, 566. [[CrossRef](#)] [[PubMed](#)]
- Curtin, K.M. Network Analysis in Geographic Information Science: Review, Assessment, and Projections. *Cartogr. Geogr. Inf. Sci.* **2007**, *34*, 103–111. [[CrossRef](#)]
- Van Der Zijpp, N.J.; Catalano, S.F. Path enumeration by finding the constrained *K*-shortest paths. *Transp. Res. Part B Methodol.* **2005**, *39*, 545–563. [[CrossRef](#)]
- Xie, C.; Waller, S.T. Parametric search and problem decomposition for approximating Pareto-optimal paths. *Transp. Res. Part B Methodol.* **2012**, *46*, 1043–1067. [[CrossRef](#)]
- Mooney, P.; Winstanley, A. An evolutionary algorithm for multicriteria path optimization problems. *Int. J. Geogr. Inf. Sci.* **2006**, *20*, 401–423. [[CrossRef](#)]
- Zeng, W.; Church, R.L. Finding shortest paths on real road networks: The case for A*. *Int. J. Geogr. Inf. Sci.* **2009**, *23*, 531–543. [[CrossRef](#)]
- Liu, P.; Liao, F.; Huang, H.J.; Timmermans, H. Dynamic Activity-Travel Assignment in Multi-State Supernetworks ☆. *Transportmetrica* **2015**, *12*, 572–590. [[CrossRef](#)]
- Carlyle, W.M.; Royset, J.O.; Kevin Wood, R. Lagrangian relaxation and enumeration for solving constrained shortest-path problems. *Networks* **2010**, *52*, 256–270. [[CrossRef](#)]
- Di, X.; Liu, H.X. Boundedly rational route choice behavior: A review of models and methodologies. *Transp. Res. Part B Methodol.* **2016**, *85*, 142–179. [[CrossRef](#)]
- Chang, G.; Caneday, L. Web-based GIS in tourism information search: Perceptions, tasks, and trip attributes. *Tour. Manag.* **2011**, *32*, 1435–1437. [[CrossRef](#)]
- Coutinho-Rodrigues, J.; Tralhão, L.; Alçada-Almeida, L. Solving a location-routing problem with a multiobjective approach: The design of urban evacuation plans. *J. Transp. Geogr.* **2012**, *22*, 206–218. [[CrossRef](#)]
- Chen, B.Y.; Lam, W.H.K.; Sumalee, A.; Li, Z. Reliable shortest path finding in stochastic networks with spatial correlated link travel times. *Int. J. Geogr. Inf. Sci.* **2012**, *26*, 365–386. [[CrossRef](#)]
- Vanhove, S.; Fack, V. Route planning with turn restrictions: A computational experiment. *Oper. Res. Lett.* **2012**, *40*, 342–348. [[CrossRef](#)]
- Milakis, D.; Athanasopoulos, K. What about people in cycle network planning? Applying participative multicriteria GIS analysis in the case of the Athens metropolitan cycle network ☆. *J. Transp. Geogr.* **2014**, *35*, 120–129. [[CrossRef](#)]
- Skov-Petersen, H.; Zachariassen, M.; Kefaloukos, P.K. Have a nice trip: An algorithm for identifying excess routes under satisfaction constraints. *Int. J. Geogr. Inf. Sci.* **2010**, *24*, 1745–1758. [[CrossRef](#)]

21. Qi, X.; Liu, L.; Liu, S. An Algorithm for Dynamic Optimal Path Selection with Constraint. *J. Comput. Inf. Syst.* **2008**, *5*, 25–30. [[CrossRef](#)]
22. Schott, R.; Staples, G.S. *Operator Calculus on Graphs: Theory and Applications in Computer Science*; World Scientific: Singapore, 2012; ISBN 9781848168763.
23. Yuan, L.; Yu, Z.; Luo, W.; Zhang, J.; Hu, Y. Clifford algebra method for network expression, computation, and algorithm construction. *Math. Methods Appl. Sci.* **2014**, *37*, 1428–1435. [[CrossRef](#)]
24. Yu, Z.; Li, D.; Zhu, S.; Luo, W.; Hu, Y.; Yuan, L. Multisource multisink optimal evacuation routing with dynamic network changes: A geometric algebra approach. *Math. Methods Appl. Sci.* **2017**. [[CrossRef](#)]
25. Yu, Z.; Wang, J.; Luo, W.; Hu, Y.; Yuan, L.; Lü, G. A dynamic evacuation simulation framework based on geometric algebra. *Comput. Environ. Urban Syst.* **2016**, *59*, 208–219. [[CrossRef](#)]
26. Chen, P.; Nie, Y. Bicriterion shortest path problem with a general nonadditive cost \star . *Transp. Res. Part B Methodol.* **2013**, *57*, 419–435. [[CrossRef](#)]
27. Pouly, M.; Kohlas, J. *Generic Inference: A Unifying Theory for Automated Reasoning*; Wiley Publishing: Hoboken, NJ, USA, 2011; ISBN 9780470527016.
28. Schott, R.; Staples, G.S. Dynamic Geometric Graph Processes: Adjacency Operator Approach. *Adv. Appl. Clifford Algebras* **2010**, *20*, 893–921. [[CrossRef](#)]
29. Staples, G.S. A new adjacency matrix for finite graphs. *Adv. Appl. Clifford Algebras* **2008**, *18*, 979–991. [[CrossRef](#)]
30. Yuan, L.; Yu, Z.; Chen, S.; Luo, W.; Wang, Y.; Lü, G. CAUSTA: Clifford algebra-based unified spatio-temporal analysis. *Trans. GIS* **2010**, *14*, 59–83. [[CrossRef](#)]
31. Garroppo, R.G.; Giordano, S.; Tavanti, L. A survey on multi-constrained optimal path computation: Exact and approximate algorithms. *Comput. Netw.* **2010**, *54*, 3081–3107. [[CrossRef](#)]
32. Liu, L.; Mu, H.; Yang, X.; He, R.; Li, Y. An oriented spanning tree based genetic algorithm for multi-criteria shortest path problems. *Appl. Soft Comput. J.* **2012**, *12*, 506–515. [[CrossRef](#)]
33. Yu, Z.; Yuan, L.; Luo, W.; Feng, L.; Lv, G. Spatio-Temporal Constrained Human Trajectory Generation from the PIR Motion Detector Sensor Network Data: A Geometric Algebra Approach. *Sensors* **2016**, *16*, 43. [[CrossRef](#)] [[PubMed](#)]
34. Agdeppa, R.P.; Yamashita, N.; Fukushima, M. The traffic equilibrium problem with nonadditive costs and its monotone mixed complementarity problem formulation. *Transp. Res. Part B Methodol.* **2007**, *41*, 862–874. [[CrossRef](#)]
35. Schott, R.; Staples, G.S. Nilpotent adjacency matrices, random graphs and quantum random variables. *J. Phys. A Math. Theor.* **2008**, *41*, 155205. [[CrossRef](#)]
36. Slimane, J.B.; René, S.; Song, Y.; Staples, G.S.; Tsiontsiou, E. *Operator Calculus Algorithms for Multi-Constrained Paths*; Southern Illinois University: Edwardsville, IL, USA, 2015; ISSN 1814-04.
37. Mohri, M. Semiring Frameworks and Algorithms for Shortest-Distance Problems. *J. Autom. Lang. Comb.* **2013**, *7*, 321–350. [[CrossRef](#)]
38. Nash, E.; Cope, A.; James, P.; Parker, D. Cycle Network Planning: Towards a Holistic Approach Using Temporal Topology. *Transp. Plan. Technol.* **2005**, *28*, 251–271. [[CrossRef](#)]
39. Lozano, L.; Medaglia, A.L. *On an Exact Method for the Constrained Shortest Path Problem*; Elsevier Science Ltd.: New York, NY, USA, 2013.
40. Yang, B.; Luan, X.; Zhang, Y. A Pattern-Based Approach for Matching Nodes in Heterogeneous Urban Road Networks. *Trans. GIS* **2015**, *18*, 718–739. [[CrossRef](#)]
41. Li, Q.; Chen, B.Y.; Wang, Y.; Lam, W.H.K. A Hybrid Link-Node Approach for Finding Shortest Paths in Road Networks with Turn Restrictions. *Trans. GIS* **2015**, *19*, 3059–3068. [[CrossRef](#)]
42. Tong, L.; Zhou, X.; Miller, H.J. Transportation network design for maximizing space–time accessibility. *Transp. Res. Part B Methodol.* **2015**, *81*, 555–576. [[CrossRef](#)]
43. Zheng, N.; Geroliminis, N. Modeling and optimization of multimodal urban networks with limited parking and dynamic pricing. *Transp. Res. Part B Methodol.* **2016**, *83*, 36–58. [[CrossRef](#)]
44. Arbex, R.O.; Cunha, C.B. Da Efficient transit network design and frequencies setting multi-objective optimization by alternating objective genetic algorithm. *Transp. Res. Part B Methodol.* **2015**, *81*, 355–376. [[CrossRef](#)]
45. Hildenbrand, D. Foundations of Geometric Algebra Computing. *AIP Conf. Proc.* **2012**, *1479*, 27–30. [[CrossRef](#)]

46. Jafari, E.; Pandey, V.; Boyles, S.D. A decomposition approach to the static traffic assignment problem. *Transp. Res. Part B Methodol.* **2017**, *105*, 270–296. [[CrossRef](#)]
47. Sanders, P.; Schultes, D. *Highway Hierarchies Hasten Exact Shortest Path Queries*; Springer: Berlin/Heidelberg, Germany, 2005.



© 2018 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).