International Journal of Geo-Information

MDPI

Article

Domain Constraints-Driven Automatic Service Composition for Online Land Cover Geoprocessing

Huaqiao Xing [1], Chang Liu [1], Rui Li [2,3,4], Haihang Wang [1], Jinhua Zhang [1] and Huayi Wu [2,3,4,*]

[1] School of Surveying and Geo-Informatics, Shandong Jianzhu University, Jinan 250101, China
[2] State Key Laboratory of Information Engineering in Surveying, Mapping and Remote Sensing, Wuhan University, Wuhan 430079, China
[3] Hubei Luojia Laboratory, Wuhan 430079, China
[4] Collaborative Innovation Center of Geospatial Technology, Wuhan 430079, China
* Correspondence: wuhuayi@whu.edu.cn

Abstract: With the rapid development of web service technology, automatic land cover web service composition has become one of the key challenges in solving complex geoprocessing tasks of land cover. Service composition requires the creation of service chains based on semantic information about the services and all the constraints that should be respected. Artificial intelligence (AI) planning algorithms have recently significantly progressed in solving web service composition problems. However, the current approaches lack effective constraints to guarantee the accuracy of automatic land cover service composition. To address this challenge, the paper proposes a domain constraints-driven automatic service composition approach for online land cover geoprocessing. First, a land cover service ontology was built to semantically describe land cover tasks, data, and services, which assist in constructing domain constraints. Then, a constraint-aware GraphPlan algorithm was proposed, which constructs a service planning graph and searches services based on the domain constraints for generating optimal web service composition solutions. In this paper, the above method was integrated into a web prototype system and a case study for the online change detection automatic geoprocessing was implemented to test the accuracy of the method. The experimental results show that with this method, a land cover service chain can generate automatically by user desire objective and domain constraints, and the service chain execution result is more accurate.

Keywords: land cover; automatic web service composition; GraphPlan; domain constraints; online geoprocessing

Citation: Xing, H.; Liu, C.; Li, R.; Wang, H.; Zhang, J.; Wu, H. Domain Constraints-Driven Automatic Service Composition for Online Land Cover Geoprocessing. *ISPRS Int. J. Geo-Inf.* **2022**, *11*, 629. https://doi.org/10.3390/ijgi11120629

Academic Editors: Wolfgang Kainz and Dev RAJ Paudyal

Received: 2 November 2022
Accepted: 14 December 2022
Published: 18 December 2022

1. Introduction

Efficient and reliable land cover information plays an essential role in environmental monitoring [1–3], resource management [4], urban studies [5], ecological assessment [6,7], and many other social-benefit areas. Traditionally, most land cover geoprocessing models have been proposed with desktop software systems in an offline manner, which is time-consuming and inefficient [8,9]. With the development of cloud computing and service computing technologies, more and more land cover data and geoprocessing models have been developed and published as web services over the web, and online geoprocessing has become a hot spot for current research [10]. Online land cover geoprocessing efficiently supports users to access data and computing resources from anywhere on the network and offers advantages in real-time spatial analysis, data and service resource sharing, or repetitive task automation [11].

In recent years, web service technology has been widely used in online land cover geoprocessing [12,13]. It is an effective way to address complex land cover tasks by composing several atomic services according to service descriptions. In general, the existing web service composition methods can be divided into two categories, i.e., manual service composition and automatic service composition [14]. During the manual service composition

process, the user must specify the workflow and manually select and bind specific services based on the matching relationship, which heavily relies on domain expert knowledge. In contrast, automatic services composition methods support the automatic discovery, selection, and binding of composite services with little human intervention [15]. AI planning has been an active area for automatic web service composition [16], and GraphPlan [17] is one of the most commonly-used algorithms in this approach. GraphPlan constructs service chains dynamically based on the semantic parameter matching relationships between services and changes in QoS values [18–20].

Compared with common web services, land cover services have distinctive domain characteristics of multi-sourcing data type and diversity geoprocessing services. First, the data involved in land cover are often acquired from various sensors with different spatial, temporal, spectral, and radiometric resolutions. Second, land cover geoprocessing services often have specific constraint requirements on the data, which cannot handle all types of remote sensing data. The coupling degree between the land cover services and their input data is high, and the business logic is closely related. Although GraphPlan has been successfully used in many computational domains, it has not been fully discussed in the online land cover geoprocessing service composition. This is mainly because that complex domain constraints information has not been considered, which is important to guarantee the accuracy of service composition.

To address the above challenge, we propose a domain constraints-driven automatic service composition approach for online land cover geoprocessing. First, a land cover service ontology was proposed to describe the land cover tasks, data, and services and assist in constructing land cover domain constraints. Second, a constraint-aware GraphPlan approach with service parameter constraints and logical process constraints was proposed. In the forward search process of GraphPlan, services were filtered and pruned by logical process constraints, which can accurately capture the true composition relationships between atomic services and improve the accuracy of service composition. In the GraphPlan backward search process, service parameter constraints were used to guide the direction of the service search in order to find the optimal service composition solution. Finally, the methods in this paper were integrated into a web system for online land cover geoprocessing and proved their validity.

The paper is organized as follows. Section 2 explores the existing methods for automatic geoprocessing service composition. Section 3 describes our proposed method. Section 4 designs an application case to illustrate the effectiveness and feasibility of the proposed approach, and Section 5 focuses on the problems encountered during the implementation of the service composition. The paper concludes with conclusions and recommendations for future research.

2. Related Work

2.1. Automatic Geoprocessing Service Composition Method

In early studies of automatic geoprocessing service composition, service chains were often composited by backlinking service input and output parameters [21–25]. However, those approaches have a significant drawback. If two different services have the same input and output parameters, it is not easy to distinguish between them. In recent years, many researchers have highlighted the importance of semantic information about service in geoprocessing service composition, arguing that an essential task during service composition is to discover services with appropriate functions [26–29]. The inputs and outputs between neighboring services must not only match in terms of parameters but also in terms of their syntax and semantics. Ontology has become a common tool for describing the semantic information of geoprocessing services, and ontological approaches are currently the most common methods for automatic geoprocessing service composition. In this direction, Li et al. (2019) proposed the concept of a spatial operation ontology to classify spatial data and services, and defined a series of spatial operations linking rules that automatically generate executable workflows through recursive algorithms [30]; Ulutas et al. (2021)

classified and represented different geospatial services through a defined geospatial service ontology, using semantic web technologies to find and match services, applying them to the automatic service composition for geographic siting [31]; Scheider et al. (2020) proposed a core concept data type ontology [32] and discussed its importance in the automatic GIS workflow construction [33].

More recent attempts have been made to automate the geoprocessing service composition using AI planning and program synthesis techniques. Farnaghi et al. (2018) explored an approach for automatic geospatial service composition under the geoportal, where the method uses an improved heuristic search algorithm to search for services and utilizes the computing power of CSW services to solve large-scale service composition questions [34]. Kruiger et al. (2021) used the automated pipeline explorer [35] (APE) tool to automate the synthesis of common geographic analysis task workflows through the concept of core conceptual data type ontology [36]. Some efforts have also been made to automate geoprocessing service composition with deep learning techniques [37–39]. However, most of the current research has focused on automatic service discovery and selection, while less attention has been paid to the accuracy of the generated service chains and the compatibility between adjacent services.

2.2. Constraints in Geoprocessing Service Composition

The constraints involved in service composition refer to the feature restrictions imposed on the service selection process to clarify the functionality of the service chain and guarantee its correct invocation. Constraints are interpreted from different perspectives in the study of service composition in computer science, such as QoS [40], context heterogeneity [41], and customer requirements [42].

Semantic constraints are increasingly considered critical in intelligent geoprocessing web service composition research. In this direction, Wiemann et al. (2018) utilize data constraints and operational constraints to dynamically search and bind geoprocessing services, facilitating spatial data analysis and decision-making in the environmental domain [43]. Xing et al. (2019) proposed an automatic service composition approach based on the semantic matching of constraint rules [14]. Hou et al. (2021) proposed a new method for parameter constraints of geoprocessing tools based on high-level and standard constraint language (SHACL), which ensures that sufficient semantic constraint information is obtained during the synthesis workflow [44]. Although some constraint information has been used in geoprocessing service composition, most approaches mainly focused on the effects of service parameter constraints. Logical process constraints were less considered in improving the efficiency and accuracy of service composition.

3. Methodology

The framework of the proposed method is shown in Figure 1, which mainly consists of two steps. In the first step, a land cover service ontology was constructed to represent the land cover service and domain constraints. In the second step, a constraint-aware GraphPlan algorithm was proposed to compose the land cover services automatically for generating an on-demand service chain.

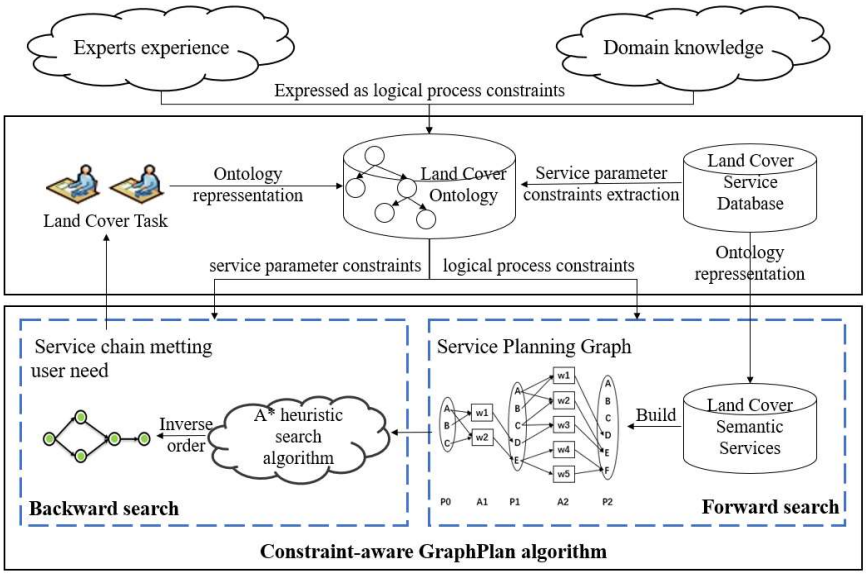


Figure 1. A methodology framework for the domain constraints-driven automatic land cover service composition approach.

3.1. Concept Definition for Land Cover Service Composition

This section provides a clear definition of the key concepts in the automatic land cover service composition.

Definition 1. *The land cover service ontology is defined as a tuple O = <C, R>, where: C denotes the concept of a class in ontology; R represents the hierarchical relationship between classes, which can be expressed as $C_2 \sqsubseteq C_1$ if a class is a subclass of another class.*

Definition 2. *The service parameter constraints represent a restricted relationship between the properties and values of service ontology individuals. For simplicity, we describe the service parameter constraints in the following form: Attr Operator Value, where: Attr represents a property of a service; Operator represents a mathematical relational operator; Value represents a property value. The following is an example of a complete service parameter constraint expression: "CoordinateSystem" = "WGS84".*

Definition 3. *The logical process constraints is defined as a tuple C = <N, T, R>, where: N and T represent the name and type of the constraint, and R represents the specific constraint rules that SWRL describes.*

Definition 4. *A land cover web service is defined as a tuple w = <Des, In, Out, Att>, where: Des denotes the description of the service; In and Out indicate the inputs and outputs of the service; Att denotes the data property of the service.*

Definition 5. *A land cover service composition problem can be used with a tuple <In_{req}, Out_{exp}, ω, O>, where: In_{req} denotes the request data provided by the user; Out_{exp} denotes the output data expected by the user; ω is denoted as a finite set of services in a service chain, with services linked sequentially, and a set of service chains can be denoted as ω = <{w_1}, {w_2, w_3} … … {w_n}>; O is the land cover service ontology, and domain constraints are constructed based on ontology.*

A land cover service composition problem is simply mapped to a Graphplan problem. The planning goal corresponds to the user's desired output goal, and the plan corresponds to a set of executable land cover service chains [45]. The process of Graphplan to find the shortest planning path can also be mapped to find the optimal service chain in the composition problem.

3.2. Land Cover Service Ontology and Domain Constraint Representation

Fully automatic service composition methods still need to be implemented, partly because of the lack of all the semantic information concerning the service composition [31]. Ontology is the formal modeling of knowledge, which defines the hierarchical classification of concepts and their interrelationships and facilitates the sharing and reusing of knowledge. Semantic description of web services through ontology models enables computers to understand the inputs and outputs of services and facilitates correct semantic links between web services [27].

According to the domain characteristics of land cover service, we developed a land cover service ontology (LCSOnt) to describe the data, land cover service, and the relationships between services in a unified way, which is the central basis of the whole work. The ontology consists of four parts (Figure 2), i.e., task ontology, data ontology, service ontology, and service chain ontology.

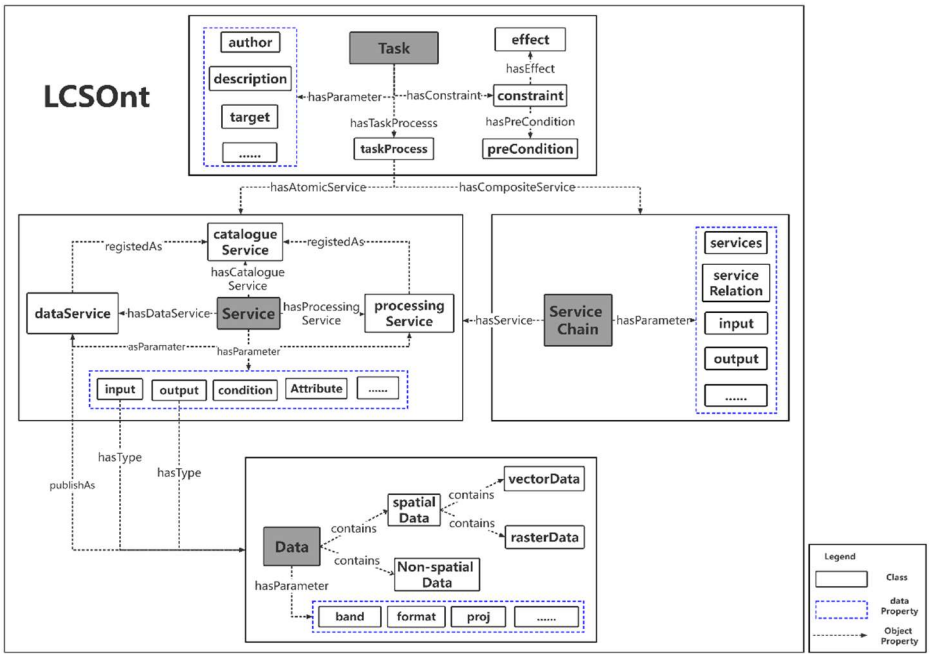


Figure 2. An ontology called LCSOnt describes land cover knowledge and facilitates the connection of land cover processes.

The task ontology describes the concepts within the task or reasoning behavior and the relationship between concepts, including task process, task constraints, and task property. The service ontology is defined for the description of the web services. In this study, the land cover services are divided into three categories: data service, catalogue service, and processing service, each of which contains the respective property information, such as input parameter, output parameter, type, etc. Among them, service parameters are represented using data ontology. The service chain ontology describes the workflow for solving complex task situations. It does not define subclasses and contains only some data properties, including the service chain description, service chain publisher, service collection of the service chain, and the relationship between services, etc. Data ontology is intended to describe the type of data, as well as the input and output of processing service and service chain, which can be divided into two main categories: spatial data and non-spatial data. Spatial data formats are the foundation of spatial data. Considering the complex and diverse parameters of the land cover service and that most of them are spatial data, we summarize all possible formats of its spatial data parameters. Spatial data is classified into vector and raster data, and the "format" data property describes their specific data format. In vector data, the possible data formats include KML/KMZ, WTK, WTB, GML, ShapeFile, DXF/DWG, GPX, GeoJSON, etc. In raster data, the possible

data formats include TIFF, GeoTIFF, NITF, Grid, JPEG, GRASS Raster, DEM, IMG, HDF, etc. In addition, an "algorithm-data" service relationship between the land cover service and the imagery data used is described using the data property of the ontology. The land cover service relationship is a functional and non-functional constraint association between services that help the automatic construction of processing workflows, which has already been proposed in our previous work [46].

In order to automate the composition of land cover services, the main challenge is to find appropriate semantic descriptions for domain constraints. In this paper, domain constraints include service parameter constraints and logical process constraints, and different domain constraints are represented in different ways. Service parameter constraints are represented by the data property of the service ontology individual, and logical process constraints are expressed using semantic web rule language (SWRL) based on the concept of LCSOnt. SWRL is an authoritative rule markup language proposed by the W3C that extends the expressiveness of OWL data through rule editing of elements defined in the ontology [47]. SWRL is intended to add additional information about relationships rather than acting as a programming language. The task of service chain generation cannot be satisfied by monotonic reasoning through SWRL rules alone. Therefore, we use OWLAPI [48], SWRLAPI [49], and SQWRL [50] for reasoning and querying land cover domain knowledge.

3.3. Constraint-Aware GraphPlan Algorithm

This section provides an outline of our planned constraint-aware GraphPlan approach. The service composition using the GraphPlan approach typically consists of the following two phases, i.e., forward search and backward search. During the forward search of GraphPlan, the land cover semantic service planning graph is generated based on the logic process constraints. Then, the backward search process uses the improved A* algorithm to generate land cover on-demand service chains by searching related web services.

3.3.1. Constructing the Land Cover Semantic Service Planning Graph

The service planning graph is a directed acyclic hierarchical graph containing two levels, parameter level P and service level A (Figure 3). The first level of the service planning graph is the parameter level P_0, whose nodes are composed of user request input parameters. The second level is service level A_1, where the input parameters for each service node contain the first-level parameters. The third layer is the parameter layer, which contains all the parameters of the P_0 layer and all the output parameters of the A_1 layer, and so on, alternating between P and A. The final layer is the parameter layer, consisting of the output parameters expected by the user.

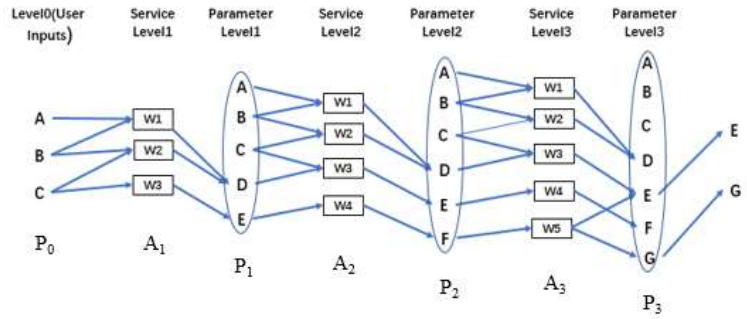


Figure 3. Example of the service planning graph model.

A forward search algorithm (Algorithm 1) is used to construct the service planning graph. In this step, logical process constraints are used along with the service parameter relationship to boost the service planning graph's semantic information and reduce redundancy in the service planning graph. The algorithm accepts as input the user request input parameter In_{req}, the user desired output parameter Out_{exp}, the service ontology O, and the

logical process constraints C, and the semantic service planning graph SG is given as output. The algorithm starts with the parameter level P_0 containing In_{req}, and the service level is empty. In the next step, the algorithm checks whether In_{req} requires data pre-processing, and if so, adds the pre-processing services that satisfy the inference results of the logical process constraint rules to the service level A_1 (Algorithm 1, lines 3–10). Next, lines 12 to 18 of the algorithm recursively analyze the accessibility of each service that can be added to the next service level, see whether its parameters match the services in the previous service level, and satisfies the logical process constraint rules inference result, then adds the filtered services and their parameters to the new service level and parameter level. The graph SG is iteratively expanded until the planning graph reaches a point where the parameter layer contains all the output parameters needed by the user and the logical process constraint rules inference result is satisfied, or the planning graph reaches a fixed point level, the graph construction algorithm then stops.

Algorithm 1. Forward Search

Input: In_{Req}, Out_{Exp}, O: Service Ontology, C: Logic Process Constraints
Output: SG: Semantic Service Planning Graph
1: A = ∅; P= {In_{Req}}; SG = {A, P};
2: S = GetOntService(O);
3: **if** In_{Req} is statisfied C **then**
4: **for** each w **in** S **do:**
5: **if** w is statisfied C **then**
6: A = A∪{w};
7: P = P∪{w.Out};
8: **end if**
9: **end for**
10: **end if**
11: **repeat**
12: **for** each w **in** S **do:**
13: **if** w.In ⊆ P and w is statisfied C **then**
14: A = A∪{w};
15: P = P∪{w.Out};
16: **end if**
17: **end for**
18: SG = SG∪{A, P};
19: **until** Out_{Exp}⊆ P ∧ G is statisfied C.process ∨ Fixedpoint(SG);

Figure 4 shows a simple diagrammatic representation of the service planning graph construction method. The algorithm in this section searches and filters services according to the inference results of logical process constraint rules under the premise of service parameter matching, grasps the real connection relationship between services, and optimizes the service search space. The generated service planning graph provides a unique search area where services from two adjacent service levels are also adjacent in the land cover service chain.

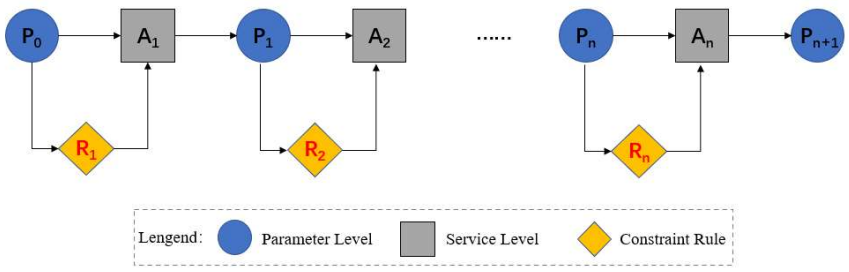


Figure 4. Illustration of the service planning graph construction method.

3.3.2. Service Composition Based on an Improved A* Algorithm

To find the optimal service composition solution in the service planning graph within the shortest time, we used the improved A* algorithm to reverse search the land cover service planning graph generated in the previous section. The basic idea of the A* algorithm is to determine an evaluation function and continuously search in the direction of the target point depending on the evaluation function from the starting point, while recording the points determined by each search until the target point is searched. Finally, return from the target point to the starting position to obtain the minimum cost path. The standard expression of the A* algorithm is f(n) = g(n) + h(n), where f(n) is the evaluation function of the current point, and g(n) and h(n) are the cost function and the heuristic function, respectively. In path planning, g(n) and h(n) usually refer to the physical distance between two points. This paper treats service composition as a pathfinding problem in a graph. Certain concepts of the A* algorithm must be redefined to adapt to the composition of land cover services with complex service parameter constraints.

The calculation of the function f(n) is redefined. The functions g(n) and h(n) are calculated as follows:

$$g(n) = \frac{1}{1 + e^{-(\#\ \text{node.step in servicechain})}} \quad (1)$$

In Equation (1), g(n) is used to evaluate the cost from the starting point to the current point, expressed in terms of the number of services in the service chain, whose value maps between 0 and 1—using the Sigmoid function. The fewer services used, the better, provided the accuracy of the service chain generation results is guaranteed;

$$h(n) = \frac{\#\ \text{Number of mismatched service parameter constraints}}{\#\ \text{Total number of service parameter constraints}} \quad (2)$$

In Equation (2), h(n) is represented as the degree of matching of the service parameter constraints between the current node and the target node. Table 1 describes examples of service parameter constraints and their matching meaning, where a parameter constraint matches between services if the forward service output parameter satisfies the backward service input parameter requirements. h(n) is calculated as the ratio between the number of unmatched service parameter constraints in the two service nodes and the total number of defined service parameter constraints. Its output value ranges between 0 to 1. To ensure the accuracy of the service composition result, the service node that best matches the service parameter constraints of the current service node must be selected, leading the search direction to the target service node.

Table 1. Examples of service parameter constraints and their matching meaning.

Constraint Content	Data Type	Matching Meaning
spatial resolution	Numeric	w_1.output.sr = w_2.input.sr
band	Int	w_1.output.band = w_2.input.band
scale	String	w_1.output.scale = w_2.input.scale
coordinate reference system	String	w_1.output.crs = w_2.input.crs
projection reference system	String	w_1.output.proj = w_2.input.proj
data format	String	w_1.output.format = w_2.input.format
data category	String	w_1. dataSource = w_2. dataSource
data size	Double	w_1.output.size $\leq$ w_2.input.size

Secondly, the path search strategy has been improved. In path planning, only one node is searched in the neighboring nodes. In contrast, in service composition, the number of nodes searched in adjacent nodes depends on the input parameter of the previous service node. The previous service node contains several input images, it searches for several service nodes in the next service level, and these service nodes then continue to search in parallel.

Algorithm 2 is the pseudo-code for the backward search algorithm (Algorithm 2). As shown in the table, the algorithm accepts as input the service planning graph G, the input

In_{req} and the composition targetOut_{exp}. It attempts to find the land cover service queue SrvSet that meets the user's requirements. The algorithm contains an openSet, a closeSet, and a currentSet, where the openSet holds the list of nodes to explore, the closeSet holds the generated list of nodes, and the currentSet stores the current node set. When openSet is not empty, the algorithm selects the same number of service nodes in openSet to add to the currentSet based on the number of input images of the previous service node (Algorithm 2, line 3). The current service nodes are removed from the openSet and added to the closeSet (Algorithm 2, lines 8–10). The neighbor nodes of the service nodes in the currentSet are added to the openSet, and their values f, g, and h are calculated afterward (Algorithm 2, lines 11–19). Until the openSet contains In_{req} or the openSet is empty, put In_{req} into the closeSet and the algorithm terminates (Algorithm 2, lines 4–7).

Algorithm 2. Backward Search

Input: G: Planning Graph, In_{Req}, Out_{Exp}
Output: SrvSet: Service Set
1: openSet = {Out_{Exp}}; closeSet = ∅; SrvSet = ∅; currentSet = {Out_{Exp}};
2: while openSet ≠ ∅ **do**
3: currentSet = openSet.getMinCostSevices();
4: **if** openSet.contain(In_{Req})
5: SrvSet.add(In_{Req});
6: return SrvSet;
7: **end if**
8: openSet.remove(currentSet);
9: closeSet.add(currentSet);
10: SrvSet.add(currentSet);
11: **for** each next in G.getNeibors(currentSet) **do:**
12: **if** next in closeSet **then**
13: continue;
14: **end if**
15: **if** next not in openSet **then**
16: openSet.add(next);
17: next.calValue();
18: **end if**
19: **end for**
20: end while

Figure 5 shows a simplified implementation of the backward search algorithm graphically, and one can see that the algorithm tends to select low-cost nodes to achieve the target. The improved A* algorithm can find the optimal service composition solution by guiding the search direction through the service parameter constraints. For the final output of the algorithm, the service queue must also be arranged in reverse order to form a service chain, which is returned as the final result of the land cover task.

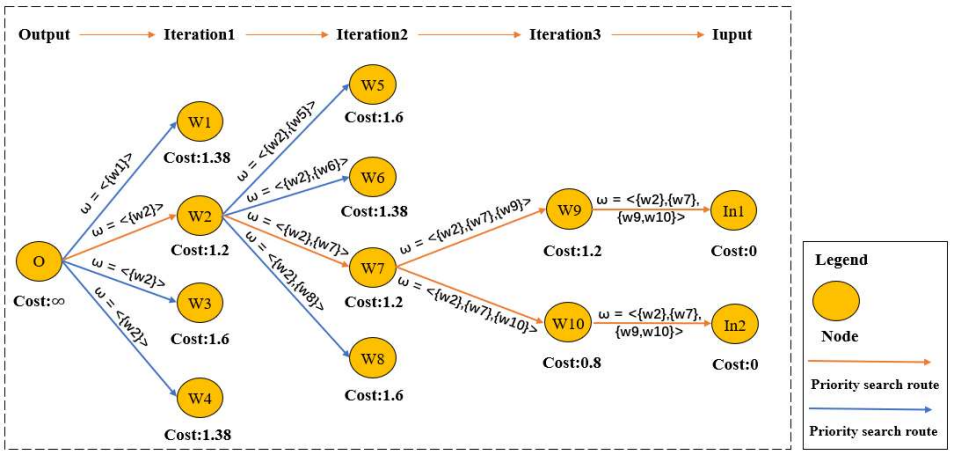


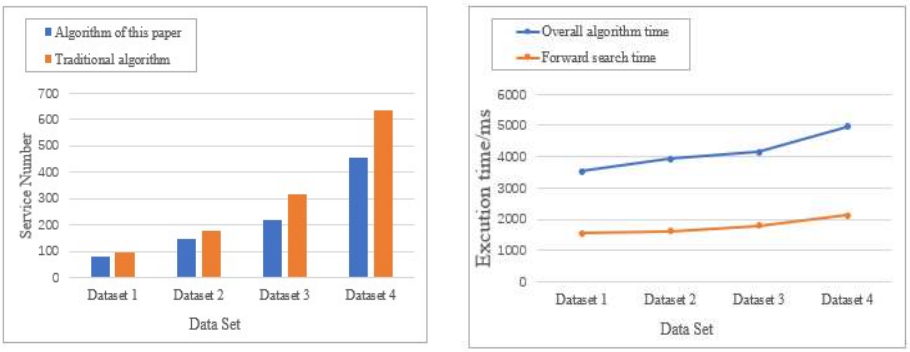
Figure 5. A simple example of the composition procedure.

4. Evaluation and Implementation

4.1. Evaluation of the Proposed Method

The performance of the constraint-aware GraphPlan algorithm was evaluated in this section. Since the number of available land cover services in the web space is limited and most of the service description information is incomplete, a land cover web service simulator was developed to generate land cover service. The land cover service generated by the simulator contains all the semantic information needed for service composition. We run experiments on a computer with the following configuration: (1) CPU: Intel(R) Core(TM) i7-4720HQ 2.60GHz RAM: 8.00GB DDR3L-1600. (2) Operating System: Windows 10 Professional 64-bit. Additionally, four land cover service data sets with the number of services 100, 200, 400, and 800 were simulated to evaluate two phases of the GraphPlan algorithm.

The results of our evaluation of this algorithm are presented in Figure 6. The first experiment was conducted using the traditional planning graph construction algorithm compared to the planning graph construction algorithm proposed in this paper (Figure 6a). In this figure, the horizontal axis represents the various data sets, while the vertical axis represents the number of services in the service planning graph. The results show that the algorithm which generates service planning graphs through logical process constraints reduces the redundancy of the graph. The more the number of services, the more pronounced the effect is. This can improve the efficiency of subsequent service recommendations, discovery, and composition. The second experiment shows the execution time results for the overall algorithm and the forward search algorithm with different data sets (Figure 6b). The test procedure was run 30 times for each data set, taking into account the unnecessary effects of the operating system and virtual software background processing, and the average of the algorithm execution times was evaluated. Results show that as the number of services increases, the service execution consumption time grows smoothly.



(**a**) (**b**)

Figure 6. Evaluation of constraint-aware GraphPlan algorithm. (**a**) The number of services in the planning graph; (**b**) The execution time of the overall algorithm and the forward search algorithm.

The focus of the algorithm in this paper is to improve the generation capability and generation quality of the land cover service chain under domain constraints. Therefore, our evaluation experiments only verified the non-redundancy of the constructed service planning graph and algorithm time complexity, with no comparison with other geoprocessing service composition methods to verify its efficiency.

4.2. Implementation in a Web-Based System

To prove the effectiveness of the approach in this paper, we developed a prototype system and integrated the LCSOnt and the constraint-aware GraphPlan algorithm into the service composition system module. The system platform is based on the B/S model and service-oriented architecture (SOA). The publication and execution of land cover services

are provided by GeoServer and 52° North, while SWRLAPI provides the querying and reasoning of ontology.

A change detection problem was chosen as a system case study. A simple scenario was considered: if a user is an engineer who is not in the land cover change detection domain and has no knowledge or experience in this domain. He has two periods of image data and expects to obtain data on land cover change areas online. We chose two different study areas and used different resolution spectral images to validate the approach (Figure 7). The area study details are shown in Table 2. Study area A uses Landsat 8 image data. The Landsat 8 images were acquired in Dezhou City, Shandong Province, which has a different coordinate system. Study area B uses Sentinel-2 image data. The Sentinel-2 images were acquired in Liaocheng City, Shandong Province, which has a different radiation resolution.

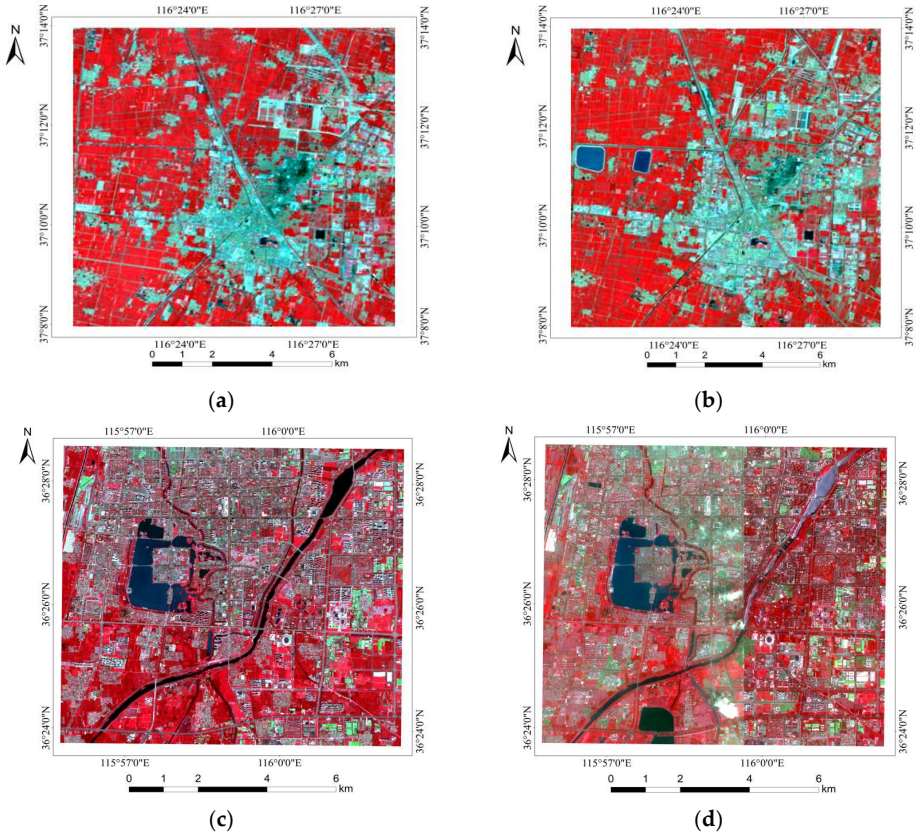


Figure 7. (**a**) The Landsat 8 image acquired in 2013 in Dezhou City, Shandong Province, China. (**b**) The Landsat 8 image acquired in 2018 in Dezhou City, Shandong Province, China. (**c**) The Sentinel-2 image acquired in 2016 in Liaocheng City, Shandong Province, China. (**d**) The Sentinel-2 image acquired in 2021 in Liaocheng City, Shandong Province, China.

LCSOnt was built using the open-source software Protégé. Figure 8 shows the main classes and properties of the ontology in Protégé. The left part shows the relationship between the classes and their subclasses, the middle part shows a visualization of the class relationships, and the right part shows the object properties between the classes.

Table 2. A description of user demand and information on images from different study areas.

User Demand	Study Area	Data Category	Description
Input	A	Landsat-8	Remote sensing image data, with a spatial resolution of 2 m, acquisition time of 21 May 2013, and pixels of 372 × 372, coordinate system WGS84.
		Landsat-8	Remote sensing image data, with a spatial resolution of 2 m, an acquisition time of 26 August 2018, and pixels of 372 × 372, coordinate system CGCS2000
	B	Sentinel-2	Remote sensing image data, with a spatial resolution of 10 m, radiation resolution 12 bits, an acquisition time of 16 May 2016, and pixels of 1225 × 890, coordinate system WGS84
		Sentinel-2	Remote sensing image data, with a spatial resolution of 10 m, radiation resolution 8 bits, an acquisition time of 26 October 2021, and pixels of 1225 × 890, coordinate system WGS84
Expected Output		Change area	Land cover change data

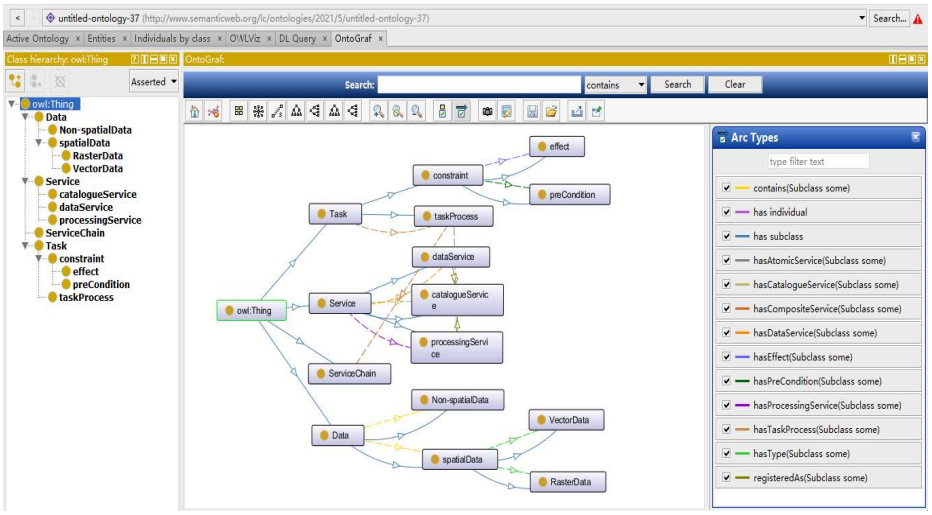


Figure 8. The schematic structure of the LCSOnt in protégé.

Change detection knowledge was represented as logical process constraints using SWRL and stored in the LCSOnt ontology. Change detection is a complex task requiring different workflows for different tasks, and a change detection algorithm that detects multiple features works better than detecting a single feature change detection algorithm [51–55]. The multi-feature change detection task workflow is generally divided into the following four steps: (1) image pre-processing, (2) multi-feature fusion change detection algorithm, (3) threshold selection, and (4) post-processing. Based on this knowledge, the logical process constraints of change detection were defined. Table 3 shows the formal representation of the SWRL rules for logical process constraints. In logical process constraint rules, the pretreatment constraint rules aim to check whether the image needs pre-processing services such as radiation calibration, atmospheric correction, cropping, coordinate conversion, and format conversion; the logical constraint rules are a constraint for service selection, limiting the connection relationships between services.

Table 3. A detailed list of defining logical process constraint rules.

SWRL Type	Name	SWRL Definition
Pretreatment Constraint Rules	AC_Rule	ServiceChain(?sc) ^ Service(?s) ^ label(?s, "AtmosphericCorrection") ^ hasInput(?sc, ?in1) ^ radiometricResolution(?in1, ?rr1) ^ hasInput(?sc, ?in2) ^ radiometricResolution(?in2, ?rr2) ^ notEqual (?rr1, ?rr2) -> hasPreService(?sc, ?s)
Logical Constraint Rules	Trans_Rule	ServiceChain(?sc) ^ Service(?s) ^ label(?s, "CoorTrans") ^ hasInput(?sc, ?in1) ^ coordinateSystem(?in1, ?c1) ^ hasInput(?sc, ?in2) ^ coordinateSystem(?in2, ?c2) ^ notEqual(?c1, ?c2) -> hasPreService(?sc, ?s)
	Clip_Rule	ServiceChain(?sc) ^ Service(?s) ^ label(?s, "Tailoring") ^ hasInput(?sc, ?in1) ^ hasInput(?sc, ?in2) ^ extent(?in1, ?e1) ^ extent(?in2, ?e2) ^ notEqual(?e1, ?e2) -> hasPreService(?sc, ?s)
	CD_Rule	ServiceChain(?sc) ^ hasPreService(?ps) ^ Service(?s) ^ type(?s, "ChangeDetectionService") ^ output(?ps, out) ^ input(?s, in) ^ equal(?in, ?out) ^ label(?s, ?la) ^ need(?sc, ?nd) ^ equal(?nd, ?la) -> hasCDService(?sc, ?s)
	Fusion_Rule	ServiceChain(?sc) ^ Service(?s) ^ label(?s, "fusion") -> hasFusionService(?sc, ?s)
	TS_Rule	ServiceChain(?sc) ^ hasFusionService(?fs) ^ Service(?s) ^ type(?s, "ThresholdSelection") ^ output(?fs, out) ^ input(?s, in) ^ equal(?in, ?out) -> hasTSService(?sc, ?s)
	PP_Rule	ServiceChain(?sc) ^ hasTSService(?ts) ^ Service(?s) ^ type(?s, "PostProcessing") ^ output(?ts, out) ^ input(?s, in) ^ equal(?in, ?out) -> hasPPService(?sc, ?s)

Based on the constraint rules defined above, the service level in the change detection service planning graph was divided into five levels: pre-processing service level, change detection service level, fusion service level, threshold selection service level, and post-processing service level. The improved A* algorithm searches layer by layer based on the service parameter constraints in this planning graph. If a service level is empty, it skips that level and continues searching, finally finding the optimal service composition solution.

In the system, the service composition module consists of three steps: data preparation, dynamic generation of service chains, and visualization of service chain execution. The data preparation step requires uploading the input image data and then filling in the necessary relevant information about the remote sensing image and the expected data product type. After the information is completed, the system can generate a land cover change detection service chain automatically. Finally, execute the service chain and visualize its results.

Uploading the study area A image data into the system, the generated service chain and its execution result are shown in Figure 9. The left part of the figure shows a visualization of the change detection service chain generated by the algorithm. It can be seen that the algorithm selects the coordinate transformation processing service (Trans) for image pre-processing, fuses the Sandwich Angle Cosine (SAM) and Euclidean Distance (ED) services, i.e., fuses the two features of spectral magnitude and spectral shape. Finally, the EM service is used for threshold selection, and the Sleve service for post-processing. The right part of the figure shows the service chain execution result. The resulting image is published as a data service using GeoServer and the user can export and download the resulting image. After exporting the result, it was found that the result was accurate and met the user's desired objective.

After uploading the study area B image data into the system and filling in the relevant information, the generated service chain and the service chain execution result are shown in Figure 10. The left part of the figure shows a visualization of the change detection service chain generated by the algorithm. It can be seen that the service chain uses the Flaash service for image pre-processing compared to the service chain generated by study area A. The reason is that the two Sentinel-2 images have the same coordinate system but different radiometric resolutions. According to the logical process constraint rules, the spatial operation Flaash was selected for atmospheric correction pre-processing. The right part of the figure shows the service chain execution result. After exporting the result, it was also found that the result was accurate and met the user's desired objective.

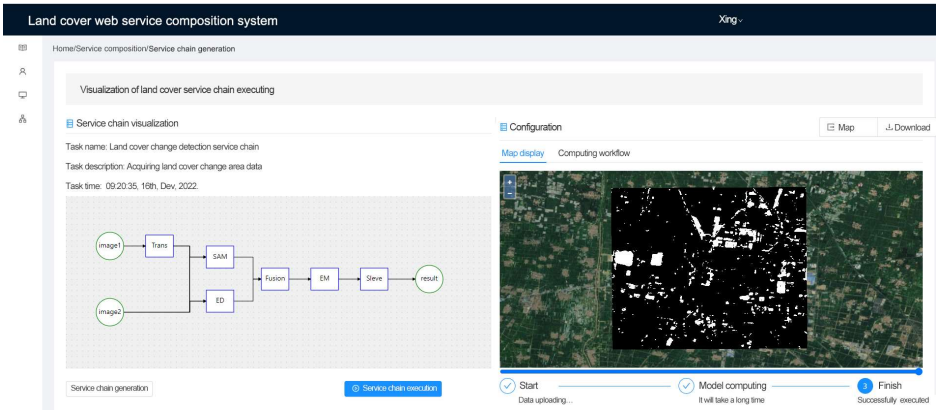


Figure 9. Diagram of the service chain and the service chain execution result for study area A.

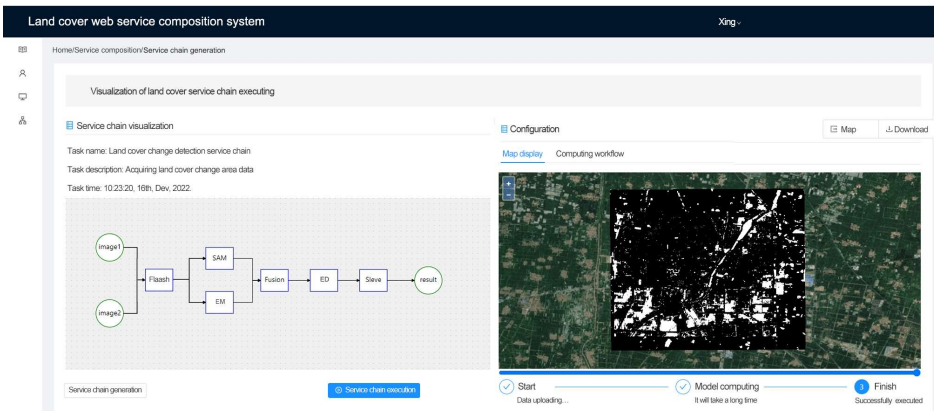


Figure 10. Diagram of the service chain and the service chain execution result for study area B.

In order to evaluate the generation ability of the change detection chain and the generation quality of the algorithm, a comparative experiment was also conducted based on study area A. The experiment was carried out between the constraint-aware GraphPlan algorithm and the traditional GraphPlan algorithm. Figure 11 shows the results of the service chains generated by these two methods. As can be seen that the service chain generated by the former approach incorporates two features (Figure 11a). In contrast, the service chain generated by the latter method detects only one single feature, and there is no coordinate conversion process (Figure 11b). In addition, we sent the service chains generated by these two different methods to the execution engine, and only the service chain in Figure 11a was successfully executed. The service chain in Figure 11b failed to execute because of a mismatch in the service parameter constraints between the CVAPS service and the EM service.

The reasons for these results are as follows. First, the service planning graphs constructed by both are different. The traditional GraphPlan algorithm does not consider the influence of expert knowledge on service planning graph construction. The service planning graph based on the traditional method contains only three service levels. There is no correct connection between service levels, leading to wrong service chain results from the backward search. Second, the constraint-aware GraphPlan algorithm considers expert knowledge to ensure the correct logical relationship between service levels. Additionally, the constraint-aware GraphPlan algorithm uses an improved A* algorithm in the backward search process to search for services based on service parameter constraints, which ensures compatibility among adjacent services in the service chain.

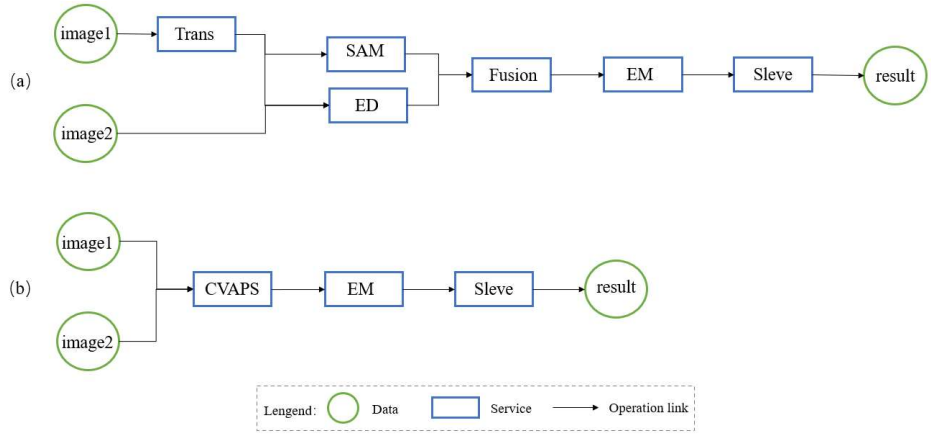


Figure 11. Two generated change detection service chains. (**a**) A service chain based on the constraint-aware GraphPlan algorithm; (**b**) A service chain bases on the traditional GraphPlan algorithm.

5. Discussion

LCSOnt provides a unified semantic description for land cover tasks, data, and services. The constraint-aware GraphPlan algorithm restricts the service composition search space by logical process constraints when constructing a service planning graph, which results in the service planning graph with real composition relationships. The backward search process uses the improved A* algorithm, which guides the path search direction through service parameter constraints and finally produces the optimal service composition scheme. Despite the advantages of the method described above, some critical issues must be improved.

Since there is no ready-made service composition ontology for the land cover domain, we have constructed an ontology to serve the land cover service composition. One of the essential elements of the land cover service composition is to capture sufficient semantic constraint information in this ontology. However, the completeness of the LCSOnt concept requires further experimentation. Which concepts are missing, which concepts are relevant for the constraint information, and is the ontology applicable to all land cover tasks? To address this weakness, we plan to develop and test LCSOnt in more diverse and complex land cover task scenarios, continuing to refine the concepts, relationships, and semantic constraint information of the ontology.

The attribute description information of land cover services published in the web space is usually incomplete, and there is no uniform standard for their semantic description. This means that in the approach of this paper, the semantic information of the land cover task, data, and service needs to be labeled manually by uniform terminology. Adding semantic information to the land cover service using ontology is tedious. In the future, we plan to introduce semantic matching techniques to measure similar land cover concepts. Users do not need to know the terminology of the land cover domain, and the algorithm understands the task objective in just one sentence.

Moreover, the algorithm does not use semantic matching techniques to measure similar concepts defined by different terms. In the future, we plan to introduce semantic matching techniques to measure similar land cover concepts. The user does not need to know the terminology of the land cover domain; the algorithm understands the task objective in just one sentence.

6. Conclusions

This paper proposed a domain constraints-driven automatic service composition approach for online land cover geoprocessing. A land cover service ontology, consisting of task ontology, service ontology, service chain ontology, and data ontology, is proposed to add the required semantic information to the land cover service composition. A constraint-aware GraphPlan algorithm is then proposed. It incorporates logical process constraints

to optimize the search space in the forward search to generate a service planning graph. The A* algorithm is used in the backward search process, and its calculation and search strategy are improved to improve the accuracy of the generated service chain. The service composition approach that considers domain constraints is suitable for complex land cover application scenarios.

The algorithm was integrated into the service composition module of a prototype web-based system to verify its accuracy. The module was used to solve automatic service composition problems in case studies related to land cover. Users do not need sufficient prior knowledge but only need to provide input data information and target demand to obtain land cover change data products. The solution is, therefore, also suitable for implementation and uses in realistic scenarios.

In future work, we will test our ontology and algorithm in multiple land cover application scenarios, verifying the integrity of the ontology and the applicability of the algorithm. In addition, we will consider creating a broader and more complex collection of land cover semantic web services on which to test our method and discuss its performance. Next, we will consider QoS's impact on service discovery and selection during service composition.

Author Contributions: Conceptualization, Huaqiao Xing, Rui Li and Huayi Wu; methodology, Chang Liu; software, Haihang Wang and Jinhua Zhang; validation, Chang Liu, Haihang Wang and Jinhua Zhang; formal analysis, Huaqiao Xing, Rui Li and Huayi Wu; investigation, Huaqiao Xing; resources, Huayi Wu; data curation, Rui Li; writing—original draft preparation, Chang Liu; writing—review and editing, Chang Liu and Huaqiao Xing; supervision, Huaqiao Xing. All authors have read and agreed to the published version of the manuscript.

Funding: This paper is jointly funded by the National Natural Science Foundation of China (No. 41930107), Shandong Provincial Natural Science Foundation (No. ZR2022YQ36), Open Fund of State Laboratory of Information Engineering in Surveying, Mapping and Remote Sensing, Wuhan University (No. 20S01).

Data Availability Statement: Not applicable.

Acknowledgments: The authors would like to thank the editor and the anonymous reviewer whose constructive comments will help to improve the presentation of this paper.

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

References

1. Atay Kaya, İ.; Kut Görgün, E. Land use and land cover change monitoring in Bandırma (Turkey) using remote sensing and geographic information systems. *Environ. Monit. Assess.* **2020**, *192*, 430. [CrossRef] [PubMed]
2. Kharazmi, R.; Tavili, A.; Rahdari, M.R.; Chaban, L.; Panidi, E.; Rodrigo-Comino, J. Monitoring and assessment of seasonal land cover changes using remote sensing: A 30-year (1987–2016) case study of Hamoun Wetland, Iran. *Environ. Monit. Assess.* **2018**, *190*, 356. [CrossRef] [PubMed]
3. Mollenhauer, H.; Kasner, M.; Haase, P.; Peterseil, J.; Wohner, C.; Frenzel, M.; Mirtl, M.; Schima, R.; Bumberger, J.; Zacharias, S. Long-term environmental monitoring infrastructures in Europe: Observations, measurements, scales, and socio-ecological representativeness. *Sci. Total Environ.* **2018**, *624*, 968–978. [CrossRef]
4. Li, S.; Yang, H.; Lacayo, M.; Liu, J.; Lei, G. Impacts of land-use and land-cover changes on water yield: A case study in Jing-Jin-Ji, China. *Sustainability* **2018**, *10*, 960. [CrossRef]
5. Dissanayake, D.; Morimoto, T.; Ranagalage, M.; Murayama, Y. Land-use/land-cover changes and their impact on surface urban heat islands: Case study of Kandy City, Sri Lanka. *Climate* **2019**, *7*, 99. [CrossRef]
6. Jin, X.; Jin, Y.; Mao, X. Ecological risk assessment of cities on the Tibetan Plateau based on land use/land cover changes–Case study of Delingha City. *Ecol. Indic.* **2019**, *101*, 185–191. [CrossRef]
7. Zhu, L.; Xing, H.; Hou, D. Analysis of carbon emissions from land cover change during 2000 to 2020 in Shandong Province, China. *Sci. Rep.* **2022**, *12*, 8021. [CrossRef]
8. Mohanty, P.; Padhy, H.M.; Mishra, P. Geoweb Application for Web based geoprocessing. *Asian J. Converg. Technol. (AJCT)* **2018**, *4*, 1–5.
9. Xing, H.; Chen, J.; Wu, H.; Zhang, J.; Liu, B. An online land cover change detection system with web service composition. In Proceedings of the 4th IEEE International Workshop on Earth Observation and Remote Sensing Applications (EORSA), Bandung, Indonesia, 25–27 May 2016; pp. 275–279.

10. Xing, H.; Hou, D.; Wang, S.; Yu, M.; Meng, F. O-LCMapping: A Google Earth Engine-based web toolkit for supporting online land cover classification. *Earth Sci. Inform.* **2021**, *14*, 529–541. [CrossRef]
11. Hofer, B. Uses of online geoprocessing technology in analyses and case studies: A systematic analysis of literature. *Int. J. Digit. Earth* **2015**, *8*, 901–917. [CrossRef]
12. de Melo, M.V.N.; de Oliveira, M.E.G.; de Almeida, G.L.P.; Gomes, N.F.; Morales, K.R.M.; Santana, T.C.; Silva, P.C.; Moraes, A.S.; Pandorfi, H.; da Silva, M.V. Spatiotemporal characterization of land cover and degradation in the agreste region of Pernambuco, Brazil, using cloud geoprocessing on Google Earth Engine. *Remote Sens. Appl. Soc. Environ.* **2022**, *26*, 100756.
13. Li, R.; Liu, W.; Peng, Y.; Zhu, X.; Zhao, T.; Che, L. Glc-statistics: A web-based spatial statistics system for global land cover data. *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci.* **2022**, *4*, 57–65. [CrossRef]
14. Xing, H.; Chen, J.; Wu, H.; Hou, D. A web service-oriented geoprocessing system for supporting intelligent land cover change detection. *ISPRS Int. J. Geo-Inf.* **2019**, *8*, 50. [CrossRef]
15. Fonte, C.C.; Minghini, M.; Patriarca, J.; Antoniou, V.; See, L.; Skopeliti, A. Generating up-to-date and detailed land use and land cover maps using OpenStreetMap and GlobeLand30. *ISPRS Int. J. Geo-Inf.* **2017**, *6*, 125. [CrossRef]
16. Baryannis, G.; Plexousakis, D. Automated web service composition: State of the art and research challenges. *ICS-FORTH Tech. Rep.* **2010**, *409*. Available online: https://www.academia.edu/16602430/Automated_Web_Service_Composition_State_of_the_Art_and_Research_Challenges (accessed on 31 October 2022).
17. Blum, A.L.; Furst, M.L. Fast planning through planning graph analysis. *Artif. Intell.* **1997**, *90*, 281–300. [CrossRef]
18. Mena, F.M.; Ucan, R.H.; Cetina, V.U.; Ramirez, F.M. Web service composition using the bidirectional Dijkstra algorithm. *IEEE Lat. Am. Trans.* **2016**, *14*, 2522–2528. [CrossRef]
19. Meyer, H.; Weske, M. Automated service composition using heuristic search. In Proceedings of the International Conference on Business Process Management, Vienna, Austria, 5–7 September 2006; pp. 81–96.
20. Yan, Y.; Chen, M.; Yang, Y. Anytime QoS optimization over the PlanGraph for web service composition. In Proceedings of the 27th Annual ACM Symposium on Applied Computing, Trento, Italy, 26–30 March 2012; pp. 1968–1975.
21. Brogi, A.; Corfini, S. Behaviour-aware discovery of Web service compositions. *Int. J. Web Serv. Res. (IJWSR)* **2007**, *4*, 1–25. [CrossRef]
22. Brogi, A.; Corfini, S.; Montes, J.F.A.; Delgado, I.N. A Prototype for Discovering Compositions of Semantic Web Services. In Proceedings of the SWAP, Pisa, Italy, 2016; Available online: https://www.yumpu.com/en/document/view/4779564/a-prototype-for-discovering-compositions-of-semantic-web-services (accessed on 31 October 2022).
23. Lutz, M. Ontology-based descriptions for semantic discovery and composition of geoprocessing services. *Geoinformatica* **2007**, *11*, 1–36. [CrossRef]
24. Thakkar, S.; Knoblock, C.A.; Ambite, J.L.; Shahabi, C. Dynamically composing web services from on-line sources. In Proceedings of the AAAI-2002 Workshop on Intelligent Service Integration, Edmonton, AB, Canada, 29 July 2002; pp. 1–7.
25. Yue, P.; Di, L.; Yang, W.; Yu, G.; Zhao, P. Semantics-based automatic composition of geospatial Web service chains. *Comput. Geosci.* **2007**, *33*, 649–665. [CrossRef]
26. Cruz, S.A.; Monteiro, A.M.; Santos, R. Automated geospatial web services composition based on geodata quality requirements. *Comput. Geosci.* **2012**, *47*, 60–74. [CrossRef]
27. Huang, W.; Harrie, L. Towards knowledge-based geovisualisation using Semantic Web technologies: A knowledge representation approach coupling ontologies and rules. *Int. J. Digit. Earth* **2020**, *13*, 976–997. [CrossRef]
28. Sun, Z.; Yue, P.; Lu, X.; Zhai, X.; Hu, L. A task ontology driven approach for live geoprocessing in a service-oriented environment. *Trans. GIS* **2012**, *16*, 867–884. [CrossRef]
29. Zhuang, C.; Xie, Z.; Ma, K.; Guo, M.; Wu, L. A task-oriented knowledge base for geospatial problem-solving. *ISPRS Int. J. Geo-Inf.* **2018**, *7*, 423. [CrossRef]
30. Li, W.; Song, M.; Tian, Y. An ontology-driven cyberinfrastructure for intelligent spatiotemporal question answering and open knowledge discovery. *ISPRS Int. J. Geo-Inf.* **2019**, *8*, 496. [CrossRef]
31. Ulutaş Karakol, D.; Cömert, Ç. Architecture for semantic web service composition in spatial data infrastructures. *Surv. Rev.* **2022**, *54*, 1–16. [CrossRef]
32. Scheider, S.; Meerlo, R.; Kasalica, V.; Lamprecht, A.-L. Ontology of core concept data types for answering geo-analytical questions. *J. Spat. Inf. Sci.* **2020**, 167–201. [CrossRef]
33. Scheider, S.; Nyamsuren, E.; Kruiger, H.; Xu, H. Geo-analytical question-answering with GIS. *Int. J. Digit. Earth* **2021**, *14*, 1–14. [CrossRef]
34. Farnaghi, M.; Mansourian, A. Multi-agent planning for automatic geospatial web service composition in geoportals. *ISPRS Int. J. Geo-Inf.* **2018**, *7*, 404. [CrossRef]
35. Kasalica, V.; Lamprecht, A.-L. APE: A command-line tool and API for automated workflow composition. In *Security and Trust Management*; Springer Science and Business Media LLC: Cham, Switzerland, 2020; pp. 464–476.
36. Kruiger, J.F.; Kasalica, V.; Meerlo, R.; Lamprecht, A.L.; Nyamsuren, E.; Scheider, S. Loose programming of GIS workflows with geo-analytical concepts. *Trans. GIS* **2021**, *25*, 424–449. [CrossRef]
37. Feoktistov, A.; Gorsky, S.; Kostromin, R.; Fedorov, R.; Bychkov, I. Integration of Web Processing Services with Workflow-Based Scientific Applications for Solving Environmental Monitoring Problems. *ISPRS Int. J. Geo-Inf.* **2021**, *11*, 8. [CrossRef]

38. Miao, L.; Liu, C.; Fan, L.; Kwan, M.-P. An OGC web service geospatial data semantic similarity model for improving geospatial service discovery. *Open Geosci.* **2021**, *13*, 245–261. [CrossRef]
39. Wei, Z.; Gui, Z.; Zhang, M.; Yang, Z.; Mei, Y.; Wu, H.; Liu, H.; Yu, J. Text GCN-SW-KNN: A novel collaborative training multi-label classification method for WMS application themes by considering geographic semantics. *Big Earth Data* **2021**, *5*, 66–89. [CrossRef]
40. Wang, Z.; Cheng, B.; Zhang, W.; Chen, J. Q-graphplan: QoS-aware automatic service composition with the extended planning graph. *IEEE Access* **2020**, *8*, 8314–8323. [CrossRef]
41. Li, X.; Madnick, S.; Zhu, H.; Fan, Y. An approach to composing web services with context heterogeneity. In Proceedings of the 2009 IEEE International Conference on Web Services, Los Angeles, CA, USA, 6–10 July 2009; pp. 695–702.
42. Zhu, M.; Fan, G.; Li, J.; Kuang, H. An approach for QoS-aware service composition with graphplan and fuzzy logic. *Procedia Comput. Sci.* **2018**, *141*, 56–63. [CrossRef]
43. Wiemann, S.; Karrasch, P.; Bernard, L. Ad-hoc combination and analysis of heterogeneous and distributed spatial data for environmental monitoring–design and prototype of a web-based solution. *Int. J. Digit. Earth* **2018**, *11*, 79–94. [CrossRef]
44. Hou, Z.-W.; Qin, C.-Z.; Zhu, A.-X.; Wang, Y.-J.; Liang, P.; Wang, Y.-J.; Zhu, Y.-Q. Formalizing Parameter Constraints to Support Intelligent Geoprocessing: A SHACL-Based Method. *ISPRS Int. J. Geo-Inf.* **2021**, *10*, 605. [CrossRef]
45. Peer, J. *Web Service Composition as AI Planning: A Survey*; University of St. Gallen Switzerland: St. Gallen, Switzerland, 2005.
46. Xing, H.; Chen, J.; Wu, H.; Zhang, J.; Li, S.; Liu, B. A service relation model for web-based land cover change detection. *ISPRS J. Photogramm. Remote Sens.* **2017**, *132*, 20–32. [CrossRef]
47. Horrocks, I.; Patel-Schneider, P.F.; Boley, H.; Tabet, S.; Grosof, B.; Dean, M. SWRL: A semantic web rule language combining OWL and RuleML. *W3C Memb. Submiss.* **2004**, *21*, 1–31.
48. Bechhofer, S.; Volz, R.; Lord, P. Cooking the Semantic Web with the OWL API. In Proceedings of the International Semantic Web Conference, Sanibel Island, FL, USA, 20–23 October 2003; pp. 659–675.
49. O'Connor, M.J.; Shankar, R.D.; Musen, M.A.; Das, A.K.; Nyulas, C. The SWRLAPI: A Development Environment for Working with SWRL Rules. In Proceedings of the 5th OWLED Workshop on OWL: Experience and Directions, Karlsruhe, Germany, 26–27 October 2008.
50. O'Connor, M.J.; Das, A.K. SQWRL: A query language for OWL. In Proceedings of the 6th International Workshop on OWL: Experiences and Directions (OWLED 2009), Chantilly, VA, USA, 23–24 October 2009; Volume 529, pp. 42–44.
51. Hou, D.; Wang, S.; Tian, X.; Xing, H. An Attention-Enhanced End-to-End Discriminative Network With Multiscale Feature Learning for Remote Sensing Image Retrieval. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2022**, *15*, 8245–8255. [CrossRef]
52. Xing, H.; Zhu, L.; Feng, Y.; Wang, W.; Hou, D.; Meng, F.; Ni, Y. An Adaptive Change Threshold Selection Method Based on Land Cover Posterior Probability and Spatial Neighborhood Information. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2021**, *14*, 11608–11621. [CrossRef]
53. Xing, H.; Zhu, L.; Hou, D.; Zhang, T. Integrating change magnitude maps of spectrally enhanced multi-features for land cover change detection. *Int. J. Remote Sens.* **2021**, *42*, 4284–4308. [CrossRef]
54. Hou, D.; Miao, Z.; Xing, H.; Wu, H. Two novel benchmark datasets from ArcGIS and bing world imagery for remote sensing image retrieval. *Int. J. Remote Sens.* **2021**, *42*, 240–258. [CrossRef]
55. Wei, D.; Hou, D.; Zhou, X.; Chen, J. Change Detection Using a Texture Feature Space Outlier Index from Mono-Temporal Remote Sensing Images and Vector Data. *Remote Sens.* **2021**, *13*, 3857. [CrossRef]